



The δ Foundation

Distinction, Arithmetic, and the Limits of Abstraction

Jonathan Washburn

Recognition Science Institute

How to read this book

This book follows a construction. We begin with a distinction that can be repeated and ask what is required to preserve its finite records. We build arithmetic and show that each stage of the number tower, from the tallies through the ratios to the line, is the unique solution of its task; for the line this uses countable choice. We examine the rules by which its statements can be proved, identify the additional principles needed for stronger decisions, and compare the resulting rules with Peano arithmetic and Heyting arithmetic. The final part connects finite descriptions with finite physical records.

Elementary algebra and experience following a proof are enough to begin. Parts I and II introduce the logical language they use; prior study of category theory is unnecessary. Chapter 9 is a more demanding introduction to a small dependent type theory. Its examples give the main argument, while its final soundness proof requires a slower reading of substitution and reduction. The chapter provides a route through those details rather than assuming they are already familiar.

Part IV requires finite-dimensional complex linear algebra: orthonormal bases, eigenvalues, positive matrices and tensor products. Its control arguments also use differentiation, integration and limits of matrix exponentials. Appendix C explains the notation and the specific moves used here; it is a companion to that background, not a replacement for a first course. The central proofs are part of the text, and each physical construction states the operations it assumes.

Read the examples before the theorem statements. In each theorem, notice the hypotheses: they identify the circumstances in which its conclusion is forced. In each proof, ask where a hypothesis is used. That habit is especially useful when the book compares a construction with a stronger logical principle. An additional principle matters only if it permits a step that was previously unavailable, and that has to be proved. Chapter 6 shows that, with induction on every formula, one of the book's three named rules, in the form the arithmetic language admits, permits nothing new, while another is exactly as strong as excluded middle.

The δ foundation concerns actual mathematics: the structures that distinction and its consistent development require. Making that claim precise requires us to state the finite constructions with which we begin, the reasoning available for studying them, and the conditions attached to each extension. Those details are the argument.

The companion volume, *The Structure of Reality*, develops the physical forcing chain. The present volume gives the mathematical construction room to be taught on its own. Its physical chapters identify the finite bridge and the resources its constructions use.

The four parts can be read at different speeds. Part I develops the arithmetic carrier and its extensions. Part II makes proof rules and their logical resources explicit. Part III compares forms of construction and examines the difference between arbitrarily precise approximation and exact decision, including what completeness of the line costs. Part IV asks what finite records and specified physical apparatus can preserve. Read these parts in order on a first pass; each uses distinctions made by the preceding one.

Appendix C explains the proof moves and matrix notation used along the way. Appendix D collects the arithmetic rules, and Appendix E is a notation guide and glossary. Selected exercise solutions are in Appendix F. Appendix G provides routes back to the main arguments. The two technical appendices supply the finite-control construction and the count identity used in the physical chapters.

The book brings several kinds of argument together. The arithmetic laws, signed differences and ratios reconstruct standard mathematics. The initiality and categoricity arguments have their classical antecedent in Dedekind [2], and the limits of first-order characterization go back to Skolem [3]. Their role here is to show, step by step, what the starting construction determines.

The δ -calculus [1] supplies the arithmetic and recognition framework. The book then makes the strength of particular conclusions visible through exact rule systems, separating models and two-direction comparisons with logical principles. Those comparisons price the book's own three rules, one classification under six combinations of supplied information and requested answer, and the exact decisions and completeness properties of the continuum. The six prices of that classification are pairwise different: for each pair, a model of intuitionistic second-order arithmetic satisfies one and refutes the other. With excluded middle, the rules prove exactly the theorems of Peano arithmetic, and with no added principle exactly those of Heyting arithmetic. The final chapter compares a richer language of functions with the rules and with Peano arithmetic. Its proofs compile into derivations by the arithmetic rules, so for sentences built from $=$, $\perp$, $\rightarrow$ and $\forall$ it proves nothing that those rules cannot derive without an added principle. These claims belong to the specified languages and representations; the surrounding constructive tradition is described in [4, 5, 6, 7].

The physical chapters use established matrix and dilation theory [9, 10, 11]. Their particular task is to connect complete finite descriptions to the stated register, controls and error accounting, and to derive a recorder bound for the specified conserved charges. The conservation argument belongs to the line of work begun by Wigner and Araki and Yanase [13, 14]. The point of including these different arguments is to make their connections checkable, with each result's own premises kept in view.

Contents

How to read this book	ii
I Distinction becomes arithmetic	1
1 A distinction that can be repeated	2
1.1 Start, then take one more step	2
1.2 Why a proof can follow the construction	3
1.3 A familiar notation, with a proved translation	4
1.4 From an orbit to a realization	5
Questions to work through	5
2 Why a faithful tally has one structure	6
2.1 The data a translation must preserve	6
2.2 There is exactly one reading	7
2.3 What makes the reading faithful?	7
2.4 The exact uniqueness result	9
2.5 Where the force of the argument lies	10
Questions to work through	10
3 Addition, multiplication, and order	11
3.1 Addition follows the second record	11
3.2 Multiplication repeats an addition	12
3.3 Why multiplication distributes	13
3.4 The step determines both operations	14
3.5 Comparing two finite records	16
3.6 Subtraction stops at the start	17
4 Differences and ratios	18
4.1 Keep both sides of a difference	18
4.2 Operations must survive a change of representative	19

4.3	The integer laws follow on the classes	20
4.4	Reading the signed construction	21
4.5	A ratio needs a nonzero denominator	22
4.6	Constructing arithmetic on ratio classes	23
4.7	Why a reciprocal keeps the sign in the numerator	23
4.8	Checking the field laws	24
4.9	Each extension is determined by its task	25
II	What a proof is allowed to use	28
5	The language and the rules	29
5.1	Expressions are finite constructions	29
5.2	One complete argument	30
5.3	The permitted inferences	30
5.4	Reading the expressions as arithmetic	32
5.5	Why every permitted step preserves truth	34
5.6	The same induction, now inside the language	35
5.7	What each family of rules supplies	35
	Questions to work through	39
6	The price of an additional principle	40
6.1	A witness, a decision, and a double negation	40
6.2	A ledger for a finite argument	42
6.3	No added principle is not the same as an empty ledger	42
6.4	The record agrees with the derivation	43
6.5	Soundness with the prices attached	44
6.6	A route's price and a theorem's exact strength	45
6.7	What the three rules cost	45
	Questions to work through	53
7	What can recognize the same record?	54
7.1	Identifications that respect addition	54
7.2	How a period reaches an earlier collision	55

7.3	The constructive classification	56
7.4	What changes when the collision is not supplied	57
7.5	What a decision procedure alone can buy	58
7.6	The unrestricted catalogue and excluded middle	61
7.7	What the classification tells us about a reader	62
7.8	Signed records keep only the pure periods	63
7.9	One classification, six prices	65
7.10	Why the six prices are different	66
	Questions to work through	70
8	Changing the representation	72
8.1	An interpretation is fixed by its operations	72
8.2	Arithmetic on lists of one mark	73
8.3	Terms agree under corresponding assignments	74
8.4	Truth is preserved in both directions	75
8.5	What survives a one-way translation	76
8.6	What happens to the proof and its ledger	78
8.7	A second proof language: Peano arithmetic	78
8.8	A quotient represents what a reading retains	80
8.9	The limits are part of the invariance theorem	81
	Questions to work through	81
III	Construction, iteration and the continuum	83
9	Iteration, recursion and induction	84
9.1	Three ways to proceed from a start	84
9.2	The language in which the question is exact	86
9.3	What indexed recursion immediately gives	87
9.4	An iterator can merge two roots	88
9.5	Recursion can leave a whole chain unreached	89
9.6	Induction does not invent a discriminator	91
9.7	The strict sequence of resources	93
9.8	Why the countermodels cover every rule	93

9.8.1	A total interpretation and its limits	94
9.8.2	Substitution and safe expressions	95
9.8.3	Joining conversions without changing the language	96
9.8.4	Soundness of the complete typing derivations	98
	Exercises	99
10	Finite construction and effective description	100
10.1	An explicit budget for a tally	100
10.2	Arithmetic carries its own finite bounds	101
10.3	A record with a decoder	102
10.4	A finite rule can describe an unbounded calculation	102
10.5	Storing the program preserves its execution	104
10.6	One program for every requested precision	105
10.7	A finite prefix is a different quantifier	106
11	The continuum through rational enclosures	108
11.1	Why every later interval constrains every earlier one	108
11.2	A unique value lies in every enclosure	109
11.3	When two protocols describe the same point	110
11.4	Rationals and arithmetic act on the intervals	111
11.5	Every classical real has such a display	112
11.6	Why finite prefixes do not provide a uniform name	113
12	Exact comparison has a cost	115
12.1	Finite stages and exact predicates	115
12.2	Questions about a binary sequence	117
12.3	Encoding an event as a small jump	117
12.4	Exact zero decisions are equivalent to WLPO	119
12.5	Sign trichotomy is equivalent to LPO	120
12.6	One-sided order is equivalent to LLPO	120
12.7	The difference between denying and finding	121
12.8	One line built from ratios	124
12.9	What completeness costs	128

12.10 Exercises	135
IV Mathematics and finite physical records	137
13 What a finite record can establish	138
13.1 The target, the record and the reader	138
13.2 Several worlds can remain compatible with one record	139
13.3 A finite capacity bound	140
13.4 Adaptive tests still produce a bounded transcript	140
13.5 An error tolerance gives a separation bound	142
13.6 Finite length without a common budget	142
13.7 The physical task left by the mathematics	143
14 The construction of finite operations	145
14.1 An operation with a retained outcome	145
14.2 An explicit reversible extension	147
14.3 A fixed register and a complete reader	148
14.4 An error bound that keeps a reference system	149
14.5 Two descriptions of finite operations	150
14.6 Searching with a margin	151
14.7 Recovering the matrix description	152
14.8 Enacting a sequence with positive energy	153
14.8.1 Retaining the phase	154
14.9 The finite bridge and its physical interfaces	155
15 Conservation and the cost of keeping a record	157
15.1 The event, the interaction and the pointer	157
15.2 Accuracy controls the noise of a sharp event	158
15.3 What exact conservation does to the pulled-back event	159
15.4 The native source and its two charges	160
15.5 A four-entry witness inside the actual source	161
15.6 How much charge spread can the recorder supply?	162
15.7 The two-charge recorder theorem	163

15.8	What the resource is, and what the result establishes	164
16	What has been forced	165
16.1	The arithmetic structure is unique	165
16.2	Identifying records and extending logic are visible choices	166
16.3	Peano arithmetic and a richer language	168
16.4	Finite access and a uniform construction differ	173
16.5	The physical bridge has a construction and a constraint	174
16.6	What a further claim must show	176
A	Finite controls and approximation	178
A.1	The matrix directions that suffice	178
A.2	From finite operations to continuous directions	179
A.3	Two native cells supply all matrix directions	180
A.4	Overlapping pairs control a whole register	183
A.5	A finite alphabet has an infinite-order seed	184
A.6	Sign masks isolate the seed's edges	185
A.7	The finite alphabet connects the cells	186
A.8	A complete record-and-field assembly	187
B	Deriving the conserved native count	191
B.1	Separate the spectral gaps of the clock	191
B.2	The entries needed for the certificate	192
B.3	A finite expression for the odd-mode projection	193
B.4	Transport to a conserving recorder	194
C	A toolkit for reading the proofs	195
C.1	Read a theorem as a task	195
C.2	Induction follows a finite construction	195
C.3	Maps, round trips and quotients	196
C.4	Quantifiers say what information must be supplied	197
C.5	Equivalences and countermodels answer different questions	197
C.6	Reading the finite matrices	198
C.7	Eigenvalues, positivity and evolution	199

C.8	Keeping both systems in the calculation	199
D	The rules at a glance	201
D.1	Reading the notation	201
D.2	The arithmetic base	201
D.3	The logical rules	202
D.4	Additional schemas and their records	202
D.5	The three orbit constructions	203
E	Notation and glossary	205
E.1	Arithmetic and maps	205
E.2	Formulas, judgments and resources	206
E.3	Typed constructions and finite descriptions	207
E.4	Physical records and finite matrices	208
E.5	Glossary	209
F	Selected solutions	213
F.1	Distinction and rigidity	213
F.2	Arithmetic and its extensions	214
F.3	Rules and logical resources	215
F.4	Recognition and representation	217
F.5	Iteration and effective description	220
F.6	Enclosures and exact comparison	221
F.7	Records and physical construction	222
F.8	Conservation and the closing argument	223
G	Finding an argument	225
G.1	What determines arithmetic?	225
G.2	What does a proof use?	225
G.3	Which constructions give stronger eliminations?	226
G.4	Does a richer language prove more about numbers?	226
G.5	When does finite information give one uniform procedure?	226
G.6	What can a physical record preserve?	227

Bibliography

228

Part I

Distinction becomes arithmetic

Chapter 1

A distinction that can be repeated

Suppose we record arrivals by making a mark. Before anyone arrives, there is no mark. The first arrival adds one. The next arrival adds another. We can change the pen, the shape of the mark, or the name we give the record. What must survive those changes if the record is still to count the same arrivals?

The question is small enough to answer completely. It is also the beginning of a much larger question: how much mathematics is already determined when we insist that distinctions can be made, retained, and repeated faithfully?

We begin with the record itself. We need a starting record and an operation that adds one step. We then ask what their finite repetition constructs. The arithmetic operations will come later. At this point, even the familiar decimal numerals would conceal more than they explain.

1.1 Start, then take one more step

Write 0_δ for the starting record. Write S for the operation that takes one further step. The first few expressions are

$$0_\delta, \quad S(0_\delta), \quad S(S(0_\delta)), \quad S(S(S(0_\delta))).$$

Each expression says how it was built. The last expression, for example, starts at 0_δ and applies the step three times. Nothing in this notation specifies the material of a recorder or the duration of an event. Those are questions about a realization. Here we are constructing the finite records that a faithful realization must distinguish.

Definition 1.1 (The finite distinction orbit). The orbit $\mathbb{N}_\delta$ consists of the finite expressions generated by these two rules:

1. 0_δ is an expression.
2. If a is an expression, then $S(a)$ is an expression.

There are no further ways to construct an expression. Equality is equality of the constructed expressions: the starting expression is different from every step expression, and two step expressions are equal exactly when their inner expressions are equal.

The final sentence matters. A device might return to an earlier state after several updates. Such a device can read the orbit, but its states do not retain every distinction made in the orbit. We will study identifications of records later. First we need the object before any such identification.

Two facts can now be read directly from the construction.

$$0_\delta \xrightarrow{S} S(0_\delta) \xrightarrow{S} S(S(0_\delta)) \longrightarrow \dots$$

Figure 1.1. The orbit is generated from a start by finite repetition of a step. The arrows describe construction. No period or identification has been imposed.

Proposition 1.2 (A start is not a step; steps preserve distinction). *For all $a, b \in \mathbb{N}_\delta$,*

$$0_\delta \neq S(a), \quad S(a) = S(b) \implies a = b.$$

Proof. The expression 0_δ is built by the first rule. An expression $S(a)$ is built by the second. The equality of expressions does not identify these different outer forms, so $0_\delta \neq S(a)$.

If $S(a) = S(b)$, both expressions have the same outer constructor. Equality of these expressions therefore requires equality of the expressions to which that constructor was applied. Hence $a = b$. $\square$

The second property is called *injectivity* of the step: different inputs cannot become the same output after one step. It is the elementary form of preserving a distinction. The first property says that stepping never produces the starting record. Together they tell us how to compare constructed records without assuming arithmetic in advance.

1.2 Why a proof can follow the construction

A finite expression has a finite construction. To establish a property of all expressions, it is enough to establish it at the start and show that one further step preserves it. The proof follows the only two ways an expression can be formed.

Proposition 1.3 (Induction on the orbit). *Let $P(a)$ be a property of orbit expressions. Suppose that*

1. $P(0_\delta)$ holds;
2. for every $a \in \mathbb{N}_\delta$, $P(a)$ implies $P(S(a))$.

Then $P(a)$ holds for every $a \in \mathbb{N}_\delta$.

Proof. Take an expression together with its finite construction. If it was built by the starting rule, the first premise supplies the property. If it was built by applying the step to an earlier expression, the property of that earlier expression and the second premise supply the property of the new one. Applying this reasoning along the finite construction establishes the property at its final expression. These are all the construction rules, so they cover every expression in the orbit. $\square$

This proof makes a background commitment visible. We are reasoning about finite inductively generated expressions and are allowed to follow their construction. That is the setting in which the orbit has been defined. The resulting induction principle belongs to that setting.

We have not produced a language, a notion of finite construction, or the rules for reasoning about it from an absence of all rules.

Keeping this point explicit helps us ask a sharper question. Given this construction, which further structures are determined? And when a stronger conclusion needs a further principle, where does that principle enter? Much of this book is devoted to making those dependencies visible.

1.3 A familiar notation, with a proved translation

The expressions become cumbersome quickly. We would like to use ordinary natural-number notation without losing the construction it abbreviates. For that purpose, construct two maps.

The first map, $r : \mathbb{N}_\delta \rightarrow \mathbb{N}$, reads an orbit expression as its ordinary natural-number index:

$$r(0_\delta) = 0, \quad r(S(a)) = r(a) + 1. \quad (1.1)$$

The second map, $e : \mathbb{N} \rightarrow \mathbb{N}_\delta$, expands an index into an orbit expression:

$$e(0) = 0_\delta, \quad e(n + 1) = S(e(n)). \quad (1.2)$$

Here $\mathbb{N}$ is the usual natural-number system used in the surrounding mathematics. Introducing it here provides a comparison with familiar notation. The orbit itself was already specified by its finite construction.

Theorem 1.4 (Reading and expansion are inverse). *For every $n \in \mathbb{N}$ and $a \in \mathbb{N}_\delta$,*

$$r(e(n)) = n, \quad e(r(a)) = a.$$

Consequently, reading and expansion give mutually inverse identifications between the two carriers.

Proof. First prove $r(e(n)) = n$ by induction on n . At zero,

$$r(e(0)) = r(0_\delta) = 0.$$

Suppose $r(e(n)) = n$. Then the recursive rules give

$$r(e(n + 1)) = r(S(e(n))) = r(e(n)) + 1 = n + 1.$$

This proves the first identity.

For the second identity, use induction on the orbit. At the starting expression,

$$e(r(0_\delta)) = e(0) = 0_\delta.$$

If $e(r(a)) = a$, then

$$e(r(S(a))) = e(r(a) + 1) = S(e(r(a))) = S(a).$$

The orbit induction principle therefore proves $e(r(a)) = a$ for every a . The two identities are exactly the assertion that the maps are inverse. $\square$

The result allows notation to become lighter without becoming ambiguous. When it is useful, we can write 1_δ for $S(0_\delta)$ and 2_δ for $S(S(0_\delta))$. These names refer to particular expressions. They do not add operations or impose equations on the orbit.

The proof also shows what a successful change of representation should provide. A proposed correspondence must do more than assign labels. It must tell us how to go in both directions and establish that no information is lost on either return journey. Here both directions are explicit.

1.4 From an orbit to a realization

The constructed orbit and a physical tally answer different questions. The orbit specifies the finite records. A tally implements a way of representing them. To examine that representation, we need to say what the tally does at the start and what it does after one more step.

This leads to the next chapter's central construction. Let another system have a starting point b and an update s . There is an evident way to read the orbit into it: send 0_δ to b , then send each additional orbit step to one application of s . What is less obvious, until proved, is that these two requirements leave no other reading.

That uniqueness does not yet say that the target retains every distinction. We will prove the unique reading first, then identify the additional conditions under which it is a one-to-one correspondence. This order keeps the force of each premise visible.

Questions to work through

1. In the proof of Proposition 1.2, which part uses the fact that the two constructors have different outer forms, and which part uses equality of their inner expressions?
2. Reproduce the two induction steps in Theorem 1.4 without looking at the proof. Explain why the first induction runs over $\mathbb{N}$ and the second over $\mathbb{N}_\delta$.
3. Which background rule is used to pass from the two construction cases to a statement about every finite orbit expression?

Chapter 2

Why a faithful tally has one structure

A tally need not look like a string of marks. It can use words, positions, or some other family of states. To compare it with the orbit, the relevant data are a starting state and a rule for the next state. We can ask for a translation that respects those data before asking whether the translation is faithful.

The distinction between those two questions is essential. Every pointed step system receives a unique reading of the orbit. Only a system that preserves distinctions and contains no unreachable extra elements is necessarily another presentation of the whole orbit.

These mapping arguments have a classical antecedent in Dedekind's treatment of arithmetic [2]. We give the construction and its proof so that the conditions behind uniqueness remain visible.

2.1 The data a translation must preserve

Definition 2.1 (A pointed step system). A pointed step system is a carrier M , a designated element $b \in M$ and a function $s : M \rightarrow M$. We write it as (M, b, s) .

At this stage there are no further conditions. In particular, we have not said that s is injective, that b is different from every output of s , or that every element of M is generated from b .

Definition 2.2 (A structure-preserving map). A map $f : \mathbb{N}_\delta \rightarrow M$ preserves the starting point and the step when

$$f(0_\delta) = b, \quad f(S(a)) = s(f(a)) \quad \text{for every } a \in \mathbb{N}_\delta. \quad (2.1)$$

The second equation says that the order of translation and stepping does not matter. We may take a step in the orbit and translate the result, or translate first and take a step in the target. Both routes must agree.

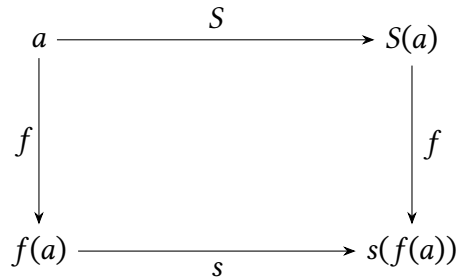


Figure 2.1. Preserving the step means that both routes have the same result. The equation $f(0_\delta) = b$ separately fixes the starting point.

2.2 There is exactly one reading

The required map can be constructed directly. Define $R_M : \mathbb{N}_\delta \rightarrow M$ by recursion:

$$R_M(0_\delta) = b, \quad R_M(S(a)) = s(R_M(a)). \quad (2.2)$$

To evaluate R_M on an expression, follow the construction of the expression. Start with b and apply s once for each outer step. The expression is finite, so this instruction specifies its value through a finite recursive construction.

Theorem 2.3 (The unique reading of the orbit). *For every pointed step system (M, b, s) , there is exactly one map $f : \mathbb{N}_\delta \rightarrow M$ preserving the starting point and the step.*

Proof. The recursively constructed R_M satisfies both equations in (2.1), so such a map exists.

For uniqueness, let f be any map satisfying those equations. We prove $f(a) = R_M(a)$ by induction on a . At the starting expression,

$$f(0_\delta) = b = R_M(0_\delta).$$

Suppose $f(a) = R_M(a)$. Preservation of the step and the recursive rule give

$$f(S(a)) = s(f(a)) = s(R_M(a)) = R_M(S(a)).$$

Induction proves equality at every orbit expression. Thus every permitted map agrees with R_M , and there is exactly one. $\square$

This property is called *initiality*. The orbit is initial among pointed step systems because every such system receives exactly one structure-preserving map from it. The name abbreviates the theorem; the content is the existence and uniqueness just proved.

Notice what the proof used. It followed the construction of the source orbit. It required no injectivity, separation or induction property in the target M . The map therefore exists even when the target identifies some of the records. Initiality tells us that the reading is determined. It does not tell us that the reading loses nothing.

2.3 What makes the reading faithful?

There are two possible failures to exclude. First, two different orbit expressions might receive the same target value. Second, the target might contain elements that no orbit expression reaches. The first is a failure of injectivity; the second is a failure of surjectivity.

The hypotheses below address both.

Definition 2.4 (A faithful generated step system). A pointed step system (M, b, s) is faithful and generated when:

1. *The step is injective:* $s(x) = s(y)$ implies $x = y$.
2. *The start is not a step:* $s(x) \neq b$ for every $x \in M$.

3. *Induction holds on the target:* for every property P of elements of M , if $P(b)$ holds and $P(x)$ implies $P(s(x))$ for every x , then $P(y)$ holds for every $y \in M$.

The third condition quantifies over properties in the surrounding mathematics. It is the full generation principle used in the argument below, and it cannot be weakened to the properties expressible by formulas of an arithmetic language. Skolem constructed systems with a start, a step, addition and multiplication that satisfy every first-order arithmetic sentence true of the natural numbers, every instance of induction for such formulas included, and that nevertheless contain elements no tally reaches [3]. In such a system the property used in Lemma 2.6, “is reached from the orbit,” cannot be written as a formula of that language.

Lemma 2.5 (No two orbit records are merged). *If (M, b, s) is a faithful generated step system, then R_M is injective.*

Proof. We prove that $R_M(a) = R_M(c)$ implies $a = c$, using induction on a .

First let $a = 0_\delta$. If $c = 0_\delta$, the conclusion is immediate. If $c = S(d)$, the assumed equality becomes

$$b = s(R_M(d)),$$

contrary to separation of the start from every step. Thus c must also be 0_δ .

Now suppose the result has been proved for a , and consider an equality $R_M(S(a)) = R_M(c)$. If $c = 0_\delta$, this equality says $s(R_M(a)) = b$, again contradicting separation. The remaining case is $c = S(d)$. The equality then reads

$$s(R_M(a)) = s(R_M(d)).$$

Injectivity of s gives $R_M(a) = R_M(d)$. The induction hypothesis gives $a = d$, and hence $S(a) = S(d) = c$.

These cases exhaust the possible forms of c at each induction step. The reading is therefore injective. $\square$

The argument tells us exactly where each condition works. Separation rules out a comparison between the start and a step. Injectivity removes a common outer step so that the comparison can be reduced to earlier expressions. No assumption about reaching every target element was needed.

Lemma 2.6 (Every target element is reached). *If (M, b, s) is a faithful generated step system, then R_M is surjective.*

Proof. Apply target induction to the property

$$P(y) \iff \text{there is an } a \in \mathbb{N}_\delta \text{ with } R_M(a) = y.$$

The property holds at b , witnessed by 0_δ , since $R_M(0_\delta) = b$. Suppose it holds at x , witnessed by an expression a . Then

$$R_M(S(a)) = s(R_M(a)) = s(x),$$

so $S(a)$ witnesses the property at $s(x)$. Target induction now gives the property at every $y \in M$. This is surjectivity. $\square$

Here the target's generation principle does all the work. We chose the property “is reached from the orbit,” proved it at the start, and showed that it survives a step. The principle then excludes any extra target element outside the reading's image.

2.4 The exact uniqueness result

A bijective structure-preserving map is an isomorphism of pointed step systems. It identifies their elements while preserving the specified start and step. The preceding results now fit together.

Theorem 2.7 (Uniqueness of a faithful generated tally). *For every faithful generated step system (M, b, s) , the canonical reading $R_M : \mathbb{N}_\delta \rightarrow M$ is bijective. It is the unique bijective map preserving the starting point and the step.*

Proof. By Lemma 2.5, the first two hypotheses make R_M injective. By Lemma 2.6, the third makes it surjective. It is therefore bijective and already preserves the structure by its construction.

Any other bijective structure-preserving map is, in particular, a structure-preserving map. The uniqueness established in Theorem 2.3 forces it to equal R_M . Thus the isomorphism is unique. $\square$

The result is stronger than saying that some convenient labels can be matched. There is only one matching compatible with the starting point and the step. Changing the names or the carrier does not introduce a choice into that matching.

It is also more precise than the statement that every system with a step is arithmetic. A raw step system always receives a reading. To identify its whole carrier with the orbit, we used injectivity, separation and generation. The conclusion carries those conditions with it.

The conditions are not only sufficient. They are exactly what a bijective reading requires.

Proposition 2.8 (The faithfulness conditions are exact). *Let (M, b, s) be a pointed step system.*

1. *Target induction holds exactly when R_M is surjective.*
2. *The system is faithful and generated exactly when R_M is bijective.*

Proof. The proof of Lemma 2.6 used only target induction, so target induction gives surjectivity. Conversely, suppose R_M is surjective, and let P satisfy $P(b)$ and $P(x) \Rightarrow P(s(x))$ for every x . Induction on the orbit proves $P(R_M(a))$ for every a , since $R_M(0_\delta) = b$ and $R_M(S(a)) = s(R_M(a))$. Every $y \in M$ is some $R_M(a)$, so $P(y)$ holds.

Theorem 2.7 proved the forward half of the second assertion. Conversely, let R_M be bijective. The first assertion gives target induction. If $s(x) = s(y)$, write $x = R_M(a)$ and $y = R_M(c)$; then $R_M(S(a)) = R_M(S(c))$, so $S(a) = S(c)$ by injectivity of R_M , and Proposition 1.2 gives $a = c$ and hence $x = y$. If $s(x) = b$ with $x = R_M(a)$, then $R_M(S(a)) = R_M(0_\delta)$, so $S(a) = 0_\delta$, contrary to Proposition 1.2. $\square$

Injectivity of the reading needs less than the first two conditions. The proof of Lemma 2.5 used them only at elements the reading reaches, and there it needs them exactly: the reading is injective precisely when the step is injective on reached elements and never takes a reached element to the start. The converse is the argument just given for the second assertion, applied to reached elements only, so that no surjectivity is needed to write them as $R_M(a)$ and $R_M(c)$. Add to the orbit one extra element $*$ with $s(*) = 0_\delta$. The start is now a step, yet the reading is still injective. What fails is surjectivity: nothing reaches $*$.

2.5 Where the force of the argument lies

There are three distinct achievements here. First, we constructed a map from the orbit into an arbitrary pointed step system. Second, we proved that no other map preserves the specified data. Third, for faithful generated targets, we proved that this map reaches every element and merges none.

The first two give initiality. The third gives categoricity within the stated class of targets. Their combination gives a unique isomorphism. Keeping these achievements separate makes the theorem easier to use: a later construction may need only the unique reading, while a change of arithmetic carrier needs the full correspondence.

This is a familiar kind of mathematical uniqueness theorem. Its role here is foundational: it shows that the arithmetic carrier obtained from finite distinction is fixed up to its unique structure-preserving correspondence. The further work is to construct operations on that carrier and to identify what additional reasoning principles are needed for stronger statements. The number of symbols used to describe the carrier does not answer either question. The constructions and their proofs do.

Questions to work through

1. In Theorem 2.3, identify the exact induction hypothesis and the equation that uses preservation of the target step. Which of the three target conditions in Definition 2.4 appears in this proof?
2. In Lemma 2.5, mark each use of separation and each use of injectivity. Explain why the target induction principle is absent.
3. State the property to which target induction is applied in Lemma 2.6. Give the witness at the start and the witness after a step.
4. Why does uniqueness among all structure-preserving maps imply uniqueness among bijective structure-preserving maps?
5. In the system obtained by adding $*$ with $s(*) = 0_\delta$, decide which conditions of Definition 2.4 hold. Exhibit a property for which target induction fails, and explain which assertion of Proposition 2.8 this illustrates.

Chapter 3

Addition, multiplication, and order

A tally tells us how many steps have been taken. Arithmetic begins when we use one tally to act on another. To add, start at one orbit position and take the steps recorded by the other. To multiply, repeat an addition. To compare, remove matching steps until one of the records reaches its start.

These instructions concern the expressions already constructed. We do not enlarge their carrier or postulate a table of arithmetic laws. We define the operations, then prove what the definitions require. The order of the proofs matters: a familiar identity cannot justify an earlier step merely because we recognize it from ordinary arithmetic.

For the rest of this chapter, write 0 for 0_δ and 1 for $S(0)$. The letters a, b, c denote elements of $\mathbb{N}_\delta$. Products such as ab mean the multiplication defined below.

3.1 Addition follows the second record

Definition 3.1 (Addition). Define addition by the two recursive clauses

$$a + 0 = a, \quad a + S(b) = S(a + b). \quad (3.1)$$

The second input controls the recursion. If it is the starting record, there is nothing to do. If it has one more step, perform the earlier addition and then take one more step. Each recursive call removes an outer step from that input, so evaluation ends on every constructed expression.

The definition is deliberately asymmetric. It gives $a + 0 = a$ immediately, but it does not give $0 + a = a$ immediately. It tells us how to move a step out of the second input, but not out of the first. Both missing facts have proofs.

Lemma 3.2 (Moving a step through addition). For all $a, b \in \mathbb{N}_\delta$,

$$0 + a = a, \quad S(a) + b = S(a + b).$$

Proof. For the first identity, induct on a . At 0 the identity is $0 + 0 = 0$, which is a defining clause. If $0 + a = a$, then

$$0 + S(a) = S(0 + a) = S(a).$$

For the second identity, fix a and induct on b . At 0, both sides are $S(a)$. If $S(a) + b = S(a + b)$, then

$$S(a) + S(b) = S(S(a) + b) = S(S(a + b)) = S(a + S(b)).$$

This is the required identity at $S(b)$. □

The second proof illustrates a recurring technique. An equation is easy to use when it matches the argument on which the definition recurses. If it does not, first prove a lemma that moves the construction into that position.

Theorem 3.3 (Order and grouping of addition). *Addition is commutative and associative:*

$$a + b = b + a, \quad (a + b) + c = a + (b + c).$$

Proof. For commutativity, induct on a , keeping b arbitrary. At 0, the two sides agree by the defining zero clause and Lemma 3.2. If $a + b = b + a$, then

$$S(a) + b = S(a + b) = S(b + a) = b + S(a).$$

For associativity, induct on c . At 0, both sides reduce to $a + b$. If $(a + b) + c = a + (b + c)$, then

$$\begin{aligned} (a + b) + S(c) &= S((a + b) + c) \\ &= S(a + (b + c)) \\ &= a + S(b + c) \\ &= a + (b + S(c)). \end{aligned}$$

The first, third and fourth equalities use the recursive addition clause; the second uses the induction hypothesis. $\square$

We may now rearrange and regroup finite sums. The permission has been earned by these proofs. It was not part of the definition of addition.

Proposition 3.4 (Cancellation). *If $a + b = a + c$, then $b = c$. If $a + c = b + c$, then $a = b$.*

Proof. For the first assertion, induct on a . At 0, remove $0 +$ from both sides by Lemma 3.2. At $S(a)$, the equality becomes $S(a + b) = S(a + c)$. Injectivity of S gives $a + b = a + c$, and the induction hypothesis gives $b = c$. For the second assertion, commute the common right addend to the left and apply the first assertion. $\square$

Cancellation will do substantial work in the next chapter. When two signed descriptions contain the same contribution on both sides of a comparison, it lets us remove that contribution without changing the comparison.

3.2 Multiplication repeats an addition

Definition 3.5 (Multiplication). Define multiplication by

$$a \cdot 0 = 0, \quad a \cdot S(b) = a \cdot b + a. \tag{3.2}$$

The second input again controls the recursion. The starting case performs no additions. Each further step adds one more copy of a to the result. Thus the operation is built from the earlier addition, just as addition was built from the step.

Lemma 3.6 (The other argument of multiplication). *For all $a, b \in \mathbb{N}_\delta$,*

$$0 \cdot b = 0, \quad S(a) \cdot b = a \cdot b + b.$$

Proof. Induction on b proves the first assertion. Its base case is a defining clause. In the step, $0 \cdot S(b) = 0 \cdot b + 0 = 0$.

For the second assertion, again induct on b . At 0, both sides are 0. At the next step, the induction hypothesis gives

$$\begin{aligned} S(a) \cdot S(b) &= S(a) \cdot b + S(a) \\ &= (a \cdot b + b) + S(a) \\ &= S((a \cdot b + b) + a) \\ &= S((a \cdot b + a) + b) \\ &= (a \cdot b + a) + S(b) \\ &= a \cdot S(b) + S(b). \end{aligned}$$

The middle rearrangement uses only the addition laws already established. □

Proposition 3.7 (Commutativity and the unit). *For all $a, b \in \mathbb{N}_\delta$,*

$$ab = ba, \quad a1 = 1a = a.$$

Proof. Induct on a for commutativity. The zero case follows from the two zero laws. In the step,

$$S(a)b = ab + b = ba + b = bS(a).$$

For the unit, $a1 = aS(0) = a0 + a = 0 + a = a$. Commutativity gives $1a = a$. □

3.3 Why multiplication distributes

To multiply by a sum, the second input asks for two successive groups of additions. We should therefore obtain the sum of the two products. The following induction makes that explanation exact.

Theorem 3.8 (Distributivity). *For all $a, b, c \in \mathbb{N}_\delta$,*

$$a(b + c) = ab + ac, \quad (a + b)c = ac + bc.$$

Proof. For the first identity, fix a, b and induct on c . At 0, both sides reduce to ab . If $a(b + c) = ab + ac$, then

$$\begin{aligned} a(b + S(c)) &= aS(b + c) \\ &= a(b + c) + a \\ &= (ab + ac) + a \\ &= ab + (ac + a) \\ &= ab + aS(c). \end{aligned}$$

The fourth equality is associativity of addition. For the second identity, commute the product, apply the first identity and commute the two resulting products:

$$(a + b)c = c(a + b) = ca + cb = ac + bc.$$

□

Proposition 3.9 (Grouping of multiplication). *For all $a, b, c \in \mathbb{N}_\delta$, $(ab)c = a(bc)$.*

Proof. Induct on c . Both sides are 0 at the start. At a successor,

$$(ab)S(c) = (ab)c + ab = a(bc) + ab = a(bc + b) = a(bS(c)).$$

The equalities use, in order, the multiplication clause, the induction hypothesis, distributivity and the multiplication clause again. □

This completes the dependency chain for the elementary algebra: recursive addition, addition laws, recursive multiplication, commutativity, distributivity and associativity. Later expansions of products of sums depend on this chain.

3.4 The step determines both operations

We defined addition and multiplication by recursion, then proved their laws. The question can be turned around. Suppose someone proposes other operations $\oplus$ and $\otimes$ on $\mathbb{N}_\delta$ and claims only a few familiar laws for them. Could they differ from ours? Not once adding one is required to be the step.

Theorem 3.10 (A few laws force the operations). *Let $\oplus$ and $\otimes$ be binary operations on $\mathbb{N}_\delta$.*

1. *If $\oplus$ is associative, and $a \oplus 0 = a$ and $a \oplus 1 = S(a)$ for every a , then $a \oplus b = a + b$ for all a, b .*
2. *If $a \otimes 0 = 0$, $a \otimes 1 = a$ and $a \otimes (b + c) = a \otimes b + a \otimes c$ for all a, b, c , then $a \otimes b = ab$ for all a, b .*

Proof. For the first assertion, fix a and induct on b . At 0, the hypothesis gives $a \oplus 0 = a = a + 0$. Suppose $a \oplus b = a + b$. Since $S(b) = b \oplus 1$ by hypothesis,

$$a \oplus S(b) = a \oplus (b \oplus 1) = (a \oplus b) \oplus 1 = S(a \oplus b) = S(a + b) = a + S(b).$$

The equalities use the step hypothesis, associativity, the step hypothesis again, the induction hypothesis and the addition clause.

For the second assertion, again fix a and induct on b . At 0 both sides are 0. The addition clauses give $b + 1 = S(b + 0) = S(b)$, so if $a \otimes b = ab$, then

$$a \otimes S(b) = a \otimes (b + 1) = a \otimes b + a \otimes 1 = ab + a = aS(b).$$

□

The hypotheses do separate work. Associativity cannot be omitted from the first assertion. The operation that agrees with $+$ when its second input is 0 or 1, and returns its first input otherwise, satisfies the other two conditions. It gives $0 \oplus 2 = 0$, and it is not associative: $(0 \oplus 1) \oplus 1 = 2$, while $0 \oplus (1 \oplus 1) = 0$. The second assertion uses no commutativity.

These laws belong to a familiar list. A *commutative semiring* is a set with two associative and commutative operations, an element neutral for the first and an element neutral for the second, such that multiplying by the first neutral element gives that element and multiplication distributes over addition. The theorems of this chapter show that $\mathbb{N}_\delta$, with 0, 1, $+$ and $\cdot$, is a commutative semiring.

Proposition 3.11 (The orbit maps uniquely into every commutative semiring). *For every commutative semiring A , with neutral elements 0_A and 1_A , there is exactly one map $f : \mathbb{N}_\delta \rightarrow A$ with*

$$f(0) = 0_A, \quad f(1) = 1_A, \quad f(a + b) = f(a) + f(b), \quad f(ab) = f(a)f(b).$$

Proof. Let f be the canonical reading of Chapter 2 into the pointed step system $(A, 0_A, x \mapsto x + 1_A)$. Then $f(1) = 0_A + 1_A = 1_A$. For additivity, induct on b ; the base case uses the neutral element, and the step is

$$f(a + S(b)) = f(S(a + b)) = f(a + b) + 1_A = f(a) + f(b) + 1_A = f(a) + f(S(b)).$$

Multiplicativity follows in the same way: the base case uses $f(a) \cdot 0_A = 0_A$, and

$$f(aS(b)) = f(ab + a) = f(a)f(b) + f(a) = f(a)(f(b) + 1_A) = f(a)f(S(b)).$$

For uniqueness, any map g with these properties satisfies $g(0) = 0_A$ and $g(S(n)) = g(n + 1) = g(n) + 1_A$. It therefore preserves the start and the step of the same pointed system, and Theorem 2.3 gives $g = f$. $\square$

Corollary 3.12 (The only semiring whose unit step is the tally step). *Let $\oplus, \otimes, e$ and u make $\mathbb{N}_\delta$ a commutative semiring, with e neutral for $\oplus$ and u neutral for $\otimes$. If $a \oplus u = S(a)$ for every a , then $e = 0$, $u = 1$, $\oplus = +$ and $\otimes = \cdot$.*

Proof. The element e is 0 or a step. Suppose $e = S(w)$. Then $w \oplus u = S(w) = e$, and

$$0 = 0 \oplus e = 0 \oplus (w \oplus u) = (0 \oplus w) \oplus u = S(0 \oplus w),$$

contrary to Proposition 1.2. Hence $e = 0$, and $u = e \oplus u = S(e) = 1$. The first part of Theorem 3.10 now gives $\oplus = +$. The semiring laws give $a \otimes 0 = 0$, $a \otimes 1 = a$ and distributivity over $+$, so the second part gives $\otimes = \cdot$. $\square$

The arithmetic of the tally is therefore not one convenient choice among several. Ask for a commutative semiring on the orbit in which adding the unit takes one step, and exactly one structure remains. The step carries that content: other commutative semiring structures on $\mathbb{N}_\delta$ keep its start and add differently, as the last question at the end of the chapter shows.

3.5 Comparing two finite records

Comparison need not inspect a decimal numeral. The expressions themselves carry enough information. Compare their outer forms. If both are steps, remove one step from each and compare what remains.

Definition 3.13 (Finite comparison). Let $L(a, b)$ have the values true or false, with recursive clauses

$$\begin{aligned} L(0, b) &= \text{true}, \\ L(S(a), 0) &= \text{false}, \\ L(S(a), S(b)) &= L(a, b). \end{aligned}$$

Write $a \leq b$ when $L(a, b) = \text{true}$.

The cases are exhaustive. The first input reaches 0, or the second does, or both have a step to remove. On each recursive call, an outer step is removed from both inputs. Every finite comparison therefore ends with an answer. This fact will later distinguish finite arithmetic from comparison of arbitrary continuum descriptions.

Theorem 3.14 (Agreement with the natural-number reading). *Let $r : \mathbb{N}_\delta \rightarrow \mathbb{N}$ be the reading of Chapter 1. Then*

$$r(a + b) = r(a) + r(b), \quad r(ab) = r(a)r(b),$$

and

$$L(a, b) = \text{true} \iff r(a) \leq r(b).$$

Proof. For addition, induct on b . At 0 the assertion reduces to $r(a) = r(a)$. For the step, use $r(S(x)) = r(x) + 1$ and the induction hypothesis:

$$r(a + S(b)) = r(S(a + b)) = r(a) + r(b) + 1 = r(a) + r(S(b)).$$

For multiplication, induct on b again. The zero case is immediate. In the step, the addition identity just proved gives

$$r(aS(b)) = r(ab + a) = r(a)r(b) + r(a) = r(a)r(S(b)).$$

For comparison, follow its three recursive cases. At $(0, b)$, the left side is true and $0 \leq r(b)$. At $(S(a), 0)$, the left side is false and $r(a) + 1 \leq 0$ is false. At $(S(a), S(b))$, remove the common successor: $r(a) + 1 \leq r(b) + 1$ is equivalent to $r(a) \leq r(b)$. The induction hypothesis identifies this last comparison with $L(a, b) = \text{true}$. $\square$

The theorem connects the constructed arithmetic with its familiar notation. It does not supply the definitions retrospectively: the recursive operations and comparison were specified before their readings were proved.

3.6 Subtraction stops at the start

Removing matching steps also supplies a finite subtraction procedure. Define the *truncated difference* $a \dot{-} b$ by

$$\begin{aligned} a \dot{-} 0 &= a, \\ 0 \dot{-} S(b) &= 0, \\ S(a) \dot{-} S(b) &= a \dot{-} b. \end{aligned} \tag{3.3}$$

There is no expression before 0 in this carrier. If the first input runs out before the second, the procedure returns 0. This is a useful operation, but it does not retain a negative difference.

Proposition 3.15 (Reading a truncated difference). *The natural-number reading of $a \dot{-} b$ is the ordinary truncated natural difference $r(a) \dot{-} r(b)$.*

Proof. Induct on a , treating b as arbitrary. If $a = 0$, the result is 0 whether b is 0 or a successor. If $a = S(c)$ and $b = 0$, both sides give $r(S(c))$. If $b = S(d)$, both truncated subtractions remove their common successor, reducing the assertion to the induction hypothesis at (c, d) . $\square$

The next chapter keeps both sides of a difference instead of stopping when one side runs out. That change in representation is enough to make signed arithmetic possible.

Questions to work through

1. In the proof of commutativity of addition, identify the exact point where the lemma for a step in the first argument is needed.
2. Reconstruct the cancellation proof without using the map r . Which property of the step permits the induction to continue?
3. In the distributivity proof, mark each equality by its justification: definition, induction hypothesis or an earlier theorem.
4. Explain why the comparison procedure terminates on the finite expressions. Which part of that explanation would need replacement for an input given by an unending approximation procedure?
5. Verify the operation described after Theorem 3.10: check $a \oplus 0 = a$ and $a \oplus 1 = S(a)$, then locate the first step of the theorem's proof that fails for it.
6. Exchange 1 and 2 in the natural-number reading and transport the ordinary operations along the resulting bijection. Show that the result is a commutative semiring on $\mathbb{N}_\delta$ whose neutral element for addition is 0, compute $1 \oplus 1$, and identify the hypothesis of Corollary 3.12 that fails.

Chapter 4

Differences and ratios

Suppose one record counts contributions and another counts removals. Keeping only their truncated difference loses information when removals exceed contributions. Keeping the two records separately avoids that loss. It also creates a new problem: different pairs can describe the same difference.

The same problem appears with ratios. Changing the unit of comparison can change both parts of a fraction while leaving its value unchanged. Arithmetic must respect these changes of description. Otherwise the result of a calculation would depend on how we happened to write its inputs.

This chapter constructs the two extensions in that order. First we identify balanced pairs of orbit records. Then we identify ratios by cross multiplication. At each stage we must prove that the identification is consistent and that the operations preserve it.

4.1 Keep both sides of a difference

Definition 4.1 (A signed record and its balance). A signed record is a pair (a, b) of elements of $\mathbb{N}_\delta$. Its first entry records the positive side and its second entry the negative side. Define

$$(a, b) \sim (c, d) \iff a + d = c + b. \quad (4.1)$$

The right side uses only orbit addition. It does not subtract anything. It compares the first positive side together with the second negative side against the second positive side together with the first negative side. Equality of those totals is the condition for balance.

The notation (a, b) still names a particular pair. The notation $[a, b]$ will name its *class*: all pairs balanced with it. Before using classes, we must show that balance is an equivalence relation. This means that a pair balances with itself, balance can be reversed, and two successive balances compose.

Proposition 4.2 (Balance is an equivalence relation). *The relation $\sim$ is reflexive, symmetric and transitive.*

Proof. Reflexivity is the identity $a + b = a + b$. Symmetry reverses the equality $a + d = c + b$.

For transitivity, suppose

$$a + d = c + b, \quad c + f = e + d.$$

We must prove $a + f = e + b$. Add f to the first equality, then use the second equality and the addition laws:

$$(a + f) + d = (a + d) + f = (c + b) + f = (c + f) + b = (e + d) + b = (e + b) + d.$$

Cancel the common addend d by Proposition 3.4. The result is the required balance. $\square$

Definition 4.3 (Constructed integers). The carrier $\mathbb{Z}_\delta$ consists of signed records identified by balance. Thus $[a, b] = [c, d]$ exactly when $a + d = c + b$.

This operation of forming classes is called taking a *quotient*. Here the word has no connection with division. It means that an explicitly specified equivalence relation determines when descriptions count as the same object. The equivalence proof is what makes the classes coherent.

4.2 Operations must survive a change of representative

The natural candidate operations on pairs are

$$\begin{aligned} (a, b) + (c, d) &= (a + c, b + d), \\ -(a, b) &= (b, a), \\ (a, b)(c, d) &= (ac + bd, ad + bc). \end{aligned} \tag{4.2}$$

The product places equal-sign contributions on the positive side and opposite-sign contributions on the negative side. Every entry is an orbit expression, so all three formulas construct signed records.

To turn these formulas into operations on $\mathbb{Z}_\delta$, we must prove more: replacing an input by a balanced pair must produce a balanced output. This requirement is called *representative independence*.

Theorem 4.4 (Signed arithmetic is representative independent). *Addition, negation and multiplication in (4.2) preserve balance. They therefore define operations on $\mathbb{Z}_\delta$.*

Proof. Suppose $(a, b) \sim (a', b')$, so $a + b' = a' + b$.

For addition of a fixed pair (c, d) , rearranging terms gives

$$(a + c) + (b' + d) = (a + b') + (c + d) = (a' + b) + (c + d) = (a' + c) + (b + d).$$

This is exactly the balance of the two sums. The same argument with the two summands exchanged handles replacement of the other input.

For negation, the required equation is $b + a' = b' + a$. It is the assumed balance with its sides reversed and its summands commuted.

For multiplication by (c, d) , the two sides of the required balance are

$$(ac + bd) + (a'd + b'c), \quad (a'c + b'd) + (ad + bc).$$

Group the terms in the first expression by their factors c and d :

$$\begin{aligned} (ac + bd) + (a'd + b'c) &= (a + b')c + (b + a')d \\ &= (a' + b)c + (b' + a)d \\ &= (a'c + b'd) + (ad + bc). \end{aligned}$$

The middle equality uses the given balance and its commuted form. The other equalities use distributivity and the addition laws from Chapter 3. Exchanging the two factors handles replacement of the second input. Replacing both inputs is now justified in two successive steps, using transitivity of balance. $\square$

We may consequently write the same operation symbols on classes. In particular, set

$$0 = [0, 0], \quad 1 = [1, 0], \quad -[a, b] = [b, a].$$

The role of the brackets matters. A pair and its negative usually differ as records, but their sum has class 0 because its two entries balance.

4.3 The integer laws follow on the classes

Theorem 4.5 (The laws of signed arithmetic). *Addition on $\mathbb{Z}_\delta$ is commutative and associative, has identity 0, and satisfies $x + (-x) = 0$. Multiplication is commutative and associative, has identity 1, and distributes over addition.*

Proof. Choose representatives $x = [a, b]$, $y = [c, d]$, $z = [e, f]$.

Commutativity and associativity of addition hold in each entry by the corresponding orbit laws. Adding $[0, 0]$ leaves both entries unchanged. Also

$$x + (-x) = [a + b, b + a] = [0, 0],$$

because $a + b = b + a$ is the balance condition with $[0, 0]$.

For multiplication, commutativity of the orbit product makes the two entries of xy equal to the two entries of yx . Multiplying by $[1, 0]$ gives $[a, b]$ using the unit and zero laws.

To check associativity, expand both $(xy)z$ and $x(yz)$. After regrouping, each has positive entry

$$ace + bde + adf + bcf$$

and negative entry

$$acf + bdf + ade + bce.$$

Every regrouping is justified by the associative and distributive orbit laws. Thus the representatives themselves agree.

For left distributivity, $x(y + z)$ has entries

$$(a(c + e) + b(d + f), a(d + f) + b(c + e)).$$

Expansion gives

$$(ac + ae + bd + bf, ad + af + bc + be),$$

which, after regrouping, are the entries of $xy + xz$. Right distributivity follows by commuting the product and applying left distributivity. $\square$

These are the laws of a commutative ring. The name packages the identities we have just proved. Subtraction is now defined by $x - y = x + (-y)$; unlike truncated subtraction, it retains a negative result.

4.4 Reading the signed construction

We can compare the constructed integers with familiar integer notation. Let r be the natural-number reading from Chapter 1, and define the reading of a pair by

$$I(a, b) = r(a) - r(b) \quad \text{in } \mathbb{Z}. \quad (4.3)$$

Here the right side is a comparison with ordinary signed arithmetic. It was not used to define the pairs, balance, or their operations.

Proposition 4.6 (The signed reading is faithful). *Two signed records balance exactly when their readings under I agree. The induced map $I : \mathbb{Z}_\delta \rightarrow \mathbb{Z}$ is bijective and preserves zero, one, addition, negation and multiplication.*

Proof. The addition reading of Theorem 3.14 gives

$$\begin{aligned} (a, b) \sim (c, d) &\iff r(a) + r(d) = r(c) + r(b) \\ &\iff r(a) - r(b) = r(c) - r(d). \end{aligned}$$

The first equivalence also uses faithfulness of r . The second is ordinary integer rearrangement. Thus I is well defined on classes and injective.

Every nonnegative integer n is the reading of $[e(n), 0]$, where e is the expansion map from Chapter 1. Every negative integer $-n$, for $n > 0$, is the reading of $[0, e(n)]$. This proves surjectivity.

Preservation of zero and one follows by reading their representatives. For addition and negation, expand the definitions:

$$\begin{aligned} I([a + c, b + d]) &= (r(a) - r(b)) + (r(c) - r(d)), \\ I([b, a]) &= -(r(a) - r(b)). \end{aligned}$$

For multiplication, the two orbit reading identities give

$$\begin{aligned} I([ac + bd, ad + bc]) &= r(a)r(c) + r(b)r(d) - r(a)r(d) - r(b)r(c) \\ &= (r(a) - r(b))(r(c) - r(d)). \end{aligned}$$

Thus the operations agree under the reading. □

The proof distinguishes construction from comparison. The carrier and its laws were built from orbit records; the bijection then identifies that construction with the usual integers. We can use this faithful reading to check an integer identity without changing the constructed objects.

4.5 A ratio needs a nonzero denominator

A ratio record consists of a signed pair and a nonzero orbit denominator. Keeping the denominator in $\mathbb{N}_\delta$ places its sign in the numerator. We will write u, v, w for the integer classes of the chosen numerator pairs, and abbreviate a ratio record by (u, d) . The record retains its actual pair; the abbreviation saves us from writing both of its entries in every formula. The letters d, e, f denote nonzero orbit denominators, also read as the corresponding positive integers when they occur in signed products.

Representative independence of signed arithmetic ensures that replacing a numerator pair by a balanced pair does not change any cross comparison below. This is why the abbreviated notation is legitimate.

Definition 4.7 (Cross equality). Two ratio records (u, d) and (v, e) are cross equal when

$$ue = vd. \quad (4.4)$$

The equation is equality of constructed integers. Equivalently, before passing to signed classes, the two scaled signed records balance.

The denominator restriction does real work. It ensures that a common denominator can be cancelled in the proof that cross equality is transitive.

Proposition 4.8 (Cross equality is an equivalence relation). *Cross equality on ratio records is reflexive, symmetric and transitive.*

Proof. Reflexivity is $ud = ud$; symmetry reverses a cross equality. For transitivity, suppose $ue = vd$ and $vf = we$. Multiply the first equality by f and the second by d . The integer laws give

$$(uf)e = (ue)f = (vd)f = (vf)d = (we)d = (wd)e.$$

The common factor e is nonzero. Apply the faithful integer reading of Proposition 4.6, cancel that nonzero integer factor, and use faithfulness to return to constructed integers. The result is $uf = wd$, as required. $\square$

The cancellation used here is ordinary integer cancellation. Its use is explicit: the preceding proposition supplied an operation-preserving bijection, so an integer equality proved after reading is also an equality of the constructed classes. No rational arithmetic has entered this proof.

Definition 4.9 (Constructed rationals). The carrier $\mathbb{Q}_\delta$ consists of ratio records identified by cross equality. Write $\langle u; d \rangle$ for the class of (u, d) .

The angle brackets keep this second quotient visually separate from the signed class $[a, b]$. Once the construction is understood, the familiar notation u/d can be used for $\langle u; d \rangle$.

4.6 Constructing arithmetic on ratio classes

Define

$$\begin{aligned}\langle u; d \rangle + \langle v; e \rangle &= \langle ue + vd; de \rangle, \\ -\langle u; d \rangle &= \langle -u; d \rangle, \\ \langle u; d \rangle \langle v; e \rangle &= \langle uv; de \rangle.\end{aligned}\tag{4.5}$$

The new denominator is allowed: a product of two nonzero orbit positions is nonzero. This follows either from their faithful natural-number readings or directly from repeated addition of a nonzero position.

Theorem 4.10 (Rational arithmetic is representative independent). *The operations in (4.5) preserve cross equality.*

Proof. Suppose $ud' = u'd$ and $ve' = v'e$. These are the hypotheses for replacing (u, d) by (u', d') and (v, e) by (v', e') .

For addition, cross multiplication of the proposed outputs gives

$$\begin{aligned}(ue + vd)(d'e') &= (ud')ee' + (ve')dd' \\ &= (u'd)ee' + (v'e)dd' \\ &= (u'e' + v'd')(de).\end{aligned}$$

Thus the two sums are cross equal.

For multiplication,

$$(uv)(d'e') = (ud')(ve') = (u'd)(v'e) = (u'v')(de).$$

For negation, $ud' = u'd$ gives $(-u)d' = (-u')d$ by negating both sides. All three operations are therefore well defined on ratio classes. $\square$

The formulas were specified as constructions. The theorem proves that they respect the identification, which is the exact condition needed to lift them to classes. It does not by itself show that no other construction could serve. Section 4.9 proves that uniqueness.

4.7 Why a reciprocal keeps the sign in the numerator

For a nonzero numerator u , let $|u|$ denote its positive magnitude. This magnitude has a finite construction. For a signed representative (a, b) , remove matching steps and add the two remaining truncated differences:

$$|(a, b)| = (a \div b) + (b \div a).\tag{4.6}$$

Its natural-number reading is $|r(a) - r(b)|$. To verify this, compare $r(a)$ and $r(b)$. In the first order, the second truncated difference is zero and the first is $r(a) - r(b)$; in the other order their roles are reversed. The truncated-difference reading from the previous chapter therefore

gives the stated absolute value. In particular, the magnitude is nonzero exactly when the signed class is nonzero.

The sign is also decided by finite comparison: (a, b) is nonnegative exactly when $L(b, a)$ is true. This agrees with $0 \leq r(a) - r(b)$ by Theorem 3.14. No search through an infinite description is involved.

Define the reciprocal on a ratio with nonzero numerator by

$$\langle u; d \rangle^{-1} = \begin{cases} \langle d; |u| \rangle, & u > 0, \\ \langle -d; |u| \rangle, & u < 0. \end{cases} \quad (4.7)$$

Both cases have a nonzero orbit denominator. To make the operation defined on every class, set $0^{-1} = 0$. This last clause is a convention for the value of a total operation; the inverse cancellation theorem retains its nonzero hypothesis.

Proposition 4.11 (The reciprocal is well defined and cancels). *Cross-equal ratio records have cross-equal reciprocals. For every nonzero $q \in \mathbb{Q}_\delta$,*

$$qq^{-1} = q^{-1}q = 1, \quad 1 = \langle 1; 1 \rangle.$$

Proof. Suppose $ue = vd$ with d, e positive. If $u = 0$, cancellation of the nonzero factor d gives $v = 0$; both reciprocal records have zero class. The same argument in reverse shows that nonzero numerators remain nonzero.

In the nonzero case, u and v have the same sign. Indeed, multiplication by a positive integer preserves sign, so opposite signs would contradict $ue = vd$. If both are positive, the reciprocal cross equation is $dv = eu$, the original equality reversed. If both are negative, it is $(-d)(-v) = (-e)(-u)$, again the same equality. This proves representative independence in all cases.

Now let $q = \langle u; d \rangle$ be nonzero. For $u > 0$, its product with the reciprocal is $\langle ud; du \rangle$, cross equal to $\langle 1; 1 \rangle$. For $u < 0$, the product is $\langle u(-d); d(-u) \rangle$, whose numerator and denominator also agree. Commutativity gives cancellation in the opposite order. $\square$

4.8 Checking the field laws

Theorem 4.12 (The rational field laws). *The constructed rationals have commutative, associative addition with zero and additive inverses; commutative, associative multiplication with one; distributivity; distinct zero and one; and multiplicative inverses for nonzero elements.*

Proof. Write three representatives as $(u, d), (v, e), (w, f)$. For addition, commutativity exchanges $ue + vd$ with $vd + ue$ and de with ed . The two groupings of a three-term sum, after expansion, have the common denominator def and numerator $uef + vdf + wde$. The two numerators agree by the integer laws, and the denominators agree by the orbit laws. This proves associativity.

Adding $\langle 0; 1 \rangle$ leaves $\langle u; d \rangle$ unchanged. Adding its negative produces $\langle 0; d^2 \rangle$, which is cross equal to $\langle 0; 1 \rangle$.

For multiplication, the two orders give uv over de , and the two groupings of three factors give uvw over def . The integer and orbit laws justify all rearrangements. Multiplying by $\langle 1; 1 \rangle$ leaves both parts unchanged.

For left distributivity, the representative of the left side is

$$\langle u(vf + we); def \rangle.$$

The representative obtained by adding the two products on the right is

$$\langle uvd f + uwde; d^2 e f \rangle.$$

The second numerator and denominator are each the first multiplied by d , so cross multiplication proves equality. Commutativity supplies right distributivity.

Zero and one are distinct because their cross equality would assert $0 = 1$ in the integer construction, contradicting its faithful reading. Proposition 4.11 supplies the inverses of nonzero elements. $\square$

The construction has now reached a field using finite records, balanced identifications, cross multiplication and explicit operations. It has not introduced a procedure for deciding every possible mathematical question. The finite comparisons used here terminate because they act on finite constructed data.

4.9 Each extension is determined by its task

The constructions of this chapter specified pairs, relations and formulas. A reader may ask whether different choices would have served as well. Each extension answers a definite question, and each is the unique answer.

Signed records make subtraction possible while keeping the orbit's addition. Write $\iota(n) = [n, 0]$. A *group* is a set with an associative operation, written $+$, a neutral element and inverses; it is a *commutative group* when the operation is also commutative. A map f from $\mathbb{N}_\delta$ into a group is *additive* when $f(a + b) = f(a) + f(b)$ for all a, b .

Theorem 4.13 (Signed records complete the orbit's addition). *The map ι is additive. For every group G , commutative or not, and every additive map $f : \mathbb{N}_\delta \rightarrow G$, there is exactly one additive map $g : \mathbb{Z}_\delta \rightarrow G$ with $g(\iota(n)) = f(n)$ for every n . It is*

$$g([a, b]) = f(a) - f(b).$$

Proof. Additivity of ι is the addition formula $[a, 0] + [b, 0] = [a+b, 0+0] = [a+b, 0]$. An additive map sends 0 to the neutral element, since $f(0) = f(0 + 0) = f(0) + f(0)$, and then $f(a + 1) = f(a) + f(1)$ shows, by induction on a , that $f(a)$ is the sum of a copies of $f(1)$. So every value of f , and every negative of a value, is a sum of copies of $f(1)$ or of $-f(1)$, and any two such sums commute. The computations below involve only these elements, so they may be rearranged as in a commutative group, whether or not G is one. The formula for g is representative independent: if $a + d = c + b$, then $f(a) + f(d) = f(c) + f(b)$, so $f(a) - f(b) = f(c) - f(d)$. It is additive, because $g([a + c, b + d]) = f(a + c) - f(b + d) = (f(a) - f(b)) + (f(c) - f(d))$.

Since $f(0)$ is the neutral element, $g(\iota(n)) = f(n) - f(0) = f(n)$. For uniqueness, every class satisfies $[a, b] = \iota(a) + (-\iota(b))$, and an additive map h on $\mathbb{Z}_\delta$ preserves negatives, because $h(x) + h(-x) = h(0)$ is the neutral element. So any such map takes $[a, b]$ to $f(a) - f(b)$. $\square$

The pair $(\mathbb{Z}_\delta, \iota)$ is determined by this property up to exactly one isomorphism compatible with the maps from the orbit: two pairs with the property map into each other, and uniqueness forces both composites to be identities. The map matters, not only the carrier. The rationals with $n \mapsto \langle \iota(n); 1 \rangle$, and the integers with $n \mapsto 2n$, both fail. In each, the element assigned to 1 has a half, and an additive map into the integers extending the natural-number reading would have to send that half to an integer h with $h + h = 1$.

Corollary 4.14 (The constructed integers map uniquely into every commutative ring). *For every commutative ring R there is exactly one map $\mathbb{Z}_\delta \rightarrow R$ preserving 1, addition and multiplication.*

Proof. Let $f : \mathbb{N}_\delta \rightarrow R$ be the map of Proposition 3.11, and let g be its extension given by Theorem 4.13. It sends $1 = [1, 0]$ to 1_R , and it preserves products:

$$g([ac + bd, ad + bc]) = f(a)f(c) + f(b)f(d) - f(a)f(d) - f(b)f(c) = (f(a) - f(b))(f(c) - f(d)).$$

Conversely, if h preserves 1, addition and multiplication, then so does $h \circ \iota$, because ι preserves them, and $h \circ \iota$ sends 0 to 0_R because it is additive. Proposition 3.11 gives $h \circ \iota = f$, and the uniqueness in Theorem 4.13 gives $h = g$. $\square$

A field K has *characteristic zero* when $1_K + \dots + 1_K$, with one or more terms, is never 0_K . Equivalently, the map $f : \mathbb{N}_\delta \rightarrow K$ of Proposition 3.11 sends every nonzero tally to a nonzero element.

Theorem 4.15 (The constructed rationals come first among fields of characteristic zero). *For every field K of characteristic zero there is exactly one map $\mathbb{Q}_\delta \rightarrow K$ preserving 1, addition and multiplication. It sends $\langle u; d \rangle$ to $g(u)/f(d)$, where f and g are the maps of Proposition 3.11 and Corollary 4.14.*

Proof. The denominator $f(d)$ is nonzero because d is nonzero. If $ue = vd$, the ring map g gives $g(u)f(e) = g(v)f(d)$, so $g(u)/f(d) = g(v)/f(e)$: the formula is representative independent. It preserves sums, since

$$\frac{g(ue + vd)}{f(de)} = \frac{g(u)f(e) + g(v)f(d)}{f(d)f(e)} = \frac{g(u)}{f(d)} + \frac{g(v)}{f(e)},$$

it preserves products in the same way, and it sends $\langle 1; 1 \rangle$ to 1_K . Conversely, let k preserve 1, addition and multiplication. On the classes $\langle u; 1 \rangle$ it preserves these operations as a function of u , so Corollary 4.14 gives $k(\langle u; 1 \rangle) = g(u)$. The cross equality $\langle u; d \rangle \langle \iota(d); 1 \rangle = \langle u; 1 \rangle$ then gives $k(\langle u; d \rangle) f(d) = g(u)$. $\square$

Two fields with this property are isomorphic by exactly one isomorphism of fields. The ordinary rationals are one of them, so exactly one isomorphism of fields connects $\mathbb{Q}_\delta$ with

them. The real numbers have characteristic zero but do not come first: a map from them into the rationals preserving the operations would send $\sqrt{2}$ to a rational whose square is 2.

The order used in this chapter's sign test is also forced. Call a subset P of a commutative ring a *positive cone* when it is closed under addition and multiplication, every element x satisfies $x \in P$, $x = 0$ or $-x \in P$, and no element has both $x \in P$ and $-x \in P$.

Proposition 4.16 (Only one order fits the arithmetic). *The classes of $\mathbb{Z}_\delta$ read as positive integers form a positive cone, and they form the only one. Likewise, the classes $\langle u; d \rangle$ with positive numerator form the only positive cone of $\mathbb{Q}_\delta$.*

Proof. The displayed sets are cones: their readings are closed under addition and multiplication, and the sign of an integer satisfies the other two conditions. For ratios, the sign of the numerator is representative independent by the argument in Proposition 4.11.

Let P be any positive cone. The last condition with $x = 0$ shows $0 \notin P$. If $x \neq 0$, then x or $-x$ lies in P , and in either case $x \cdot x = (-x)(-x) \in P$. Hence $1 = 1 \cdot 1 \in P$, and by addition every $\iota(n)$ with $n \neq 0$ lies in P . In $\mathbb{Z}_\delta$ these are exactly the classes read as positive integers. A class read as negative has its negative in P , so it is not in P . Thus P is the displayed cone. In $\mathbb{Q}_\delta$, also $\langle 1; d \rangle = \iota(d); 1 \rangle \langle 1; d \rangle^2 \in P$. A ratio with positive numerator, $\langle u; d \rangle = \langle u; 1 \rangle \langle 1; d \rangle$, therefore lies in P , and a ratio with negative numerator does not, because its negative does. $\square$

The order is therefore part of the arithmetic rather than an additional choice. Any ordering compatible with addition and multiplication in this sense is the one read from the finite comparison. The line built from these ratios in Chapter 12 is determined by its task from the other side: every Archimedean ordered field maps into it in exactly one way that preserves the arithmetic and the order, once countable choice is granted (Theorem 12.24). Which further conclusions require additional principles is the question of the next part.

Questions to work through

1. In the transitivity proof for balance, explain why equality of $a + d$ and $c + b$ is more useful than trying to subtract in $\mathbb{N}_\delta$.
2. Identify the two different tasks in defining an operation on classes: constructing its output and proving representative independence. Locate both tasks for signed multiplication.
3. Explain precisely where a nonzero denominator is used in the transitivity proof for cross equality.
4. Reproduce the two sign cases of the reciprocal proof. Explain why assigning $0^{-1} = 0$ does not establish inverse cancellation at zero.
5. Show directly that the integers with $n \mapsto 2n$ fail Theorem 4.13: take f to be the natural-number reading and find the equation that has no integer solution.
6. Using only the three cone conditions, show that every positive cone of $\mathbb{Q}_\delta$ contains $\langle 1; d \rangle$ for every nonzero d . Which condition excludes a cone containing -1 ?

Part II

What a proof is allowed to use

Chapter 5

The language and the rules

The equation $1 + 1 = 2$ can be read at two levels. We can calculate with the constructed numbers and establish the equality. We can also write a finite argument whose permitted steps are completely specified. The first activity concerns numbers. The second makes the argument itself an object we can study.

This distinction matters when we ask what a result requires. A calculation may be valid, yet conceal the use of a stronger principle in one of its steps. To expose that use, we need to say which expressions are allowed, which inferences count as steps, and how those expressions are interpreted.

We begin with arithmetic. This choice gives us a domain already constructed in Part I and a small language whose rules can be stated in full. It also keeps two questions separate: what the arithmetic structure is, and what a particular formal argument can establish about it.

5.1 Expressions are finite constructions

An arithmetic expression such as $S(0)+S(0)$ names a number. A statement such as $S(0)+S(0) = S(S(0))$ says something that can be proved. We call the first kind of expression a *term* and the second a *formula*.

Definition 5.1 (The arithmetic language). Terms are built from variables $x_0, x_1, \dots$, the symbol 0 , and the operations S , $+$ and $\cdot$. Formulas are built from equalities of terms, the false formula $\perp$, conjunction $\wedge$, disjunction $\vee$, implication $\rightarrow$, and the quantifiers $\forall$ and $\exists$. Negation abbreviates implication to falsehood:

$$\neg\varphi := \varphi \rightarrow \perp.$$

Each term and formula is a finite expression generated by these rules.

The symbols 1 and 2 remain abbreviations for $S(0)$ and $S(S(0))$. An equality between two terms is a formula; it does not assert that the two written expressions have the same shape. Thus $1 + 1$ and 2 are different terms, and $1 + 1 = 2$ is a statement about their values.

We use positions for variables so that the treatment of quantifiers is exact. Immediately inside a quantifier, x_0 denotes its newly bound variable. The earlier variables move up one position. Inside two nested quantifiers, x_0 refers to the inner one and x_1 to the outer one. Ordinary notation such as $\forall x \varphi(x)$ will be used when this binding convention is unambiguous.

Two operations on expressions are needed. *Lifting* inserts space for a new variable. *Substitution* replaces a variable by a term. They must respect the distinction between a free variable and one bound by a quantifier.

Definition 5.2 (Lifting and substitution). To lift an expression by d at cutoff c , leave a variable x_n unchanged when $n < c$, and replace it by x_{n+d} otherwise. Apply the operation to each constituent expression; on entering a quantifier, increase c by one. Write $\varphi \uparrow$ for lifting by one at cutoff zero.

To substitute a term t at position k , replace x_k by t , lower an index $n > k$ to $n - 1$, and leave an index $n < k$ unchanged. Again act on each constituent expression. Under a quantifier, substitute at $k + 1$ and first lift the free variables of t by one. Write $\varphi(t)$ for substitution at position zero.

The lowering in substitution has a purpose: instantiating a quantifier removes its variable position. The lifting under a quantifier prevents a free variable in the substituted term from becoming bound accidentally. These operations are determined by the finite expression. Neither asks whether a formula is true.

5.2 One complete argument

A *context* Γ is a finite list of formulas admitted temporarily as hypotheses. A judgment

$$\Gamma \vdash \varphi$$

says that a finite derivation of φ from those hypotheses has been given. A blank context is written simply $\vdash \varphi$.

Here is the promised proof of $1 + 1 = 2$. Two arithmetic rules supply

$$1 + 0 = 1, \quad 1 + S(0) = S(1 + 0).$$

Equality substitution replaces $1 + 0$ by 1 in the second formula. The whole derivation is therefore

$$\frac{\vdash 1 + 0 = 1 \quad \vdash 1 + S(0) = S(1 + 0)}{\vdash 1 + S(0) = S(1)} \quad \text{equality substitution.} \quad (5.1)$$

The concluding formula is $1 + 1 = 2$ after expanding the numeral abbreviations. There is no induction in this argument, and there is no decision about an arbitrary formula. Its two leaves are defining equations of addition.

For complete precision, the formula with a substitution position is $1 + S(0) = S(x_0)$. Filling that position with $1 + 0$ produces the second premise; filling it with 1 produces the conclusion. This is the general equality rule in a small case.

5.3 The permitted inferences

We now give the rules behind judgments. Each displayed premise must itself have a finite derivation. Temporary hypotheses appear to the left of $\vdash$. A rule with no premises starts a branch of the derivation tree.

Hypotheses and equality

Any formula listed in Γ may be used. Equality has reflexivity and substitution:

$$\Gamma \vdash t = t, \quad \frac{\Gamma \vdash t = s \quad \Gamma \vdash \varphi(t)}{\Gamma \vdash \varphi(s)}.$$

The terms and the formula with its substitution position are specified as part of the application of the rule. A resemblance between formulas is not enough; the premise must be the indicated instance.

The arithmetic structure

The start and the step satisfy

$$\Gamma \vdash \neg(S(t) = 0), \quad \frac{\Gamma \vdash S(t) = S(u)}{\Gamma \vdash t = u}.$$

The recursive operations supply four further formulas:

$$\begin{aligned} \Gamma \vdash t + 0 &= t, & \Gamma \vdash t + S(u) &= S(t + u), \\ \Gamma \vdash t \cdot 0 &= 0, & \Gamma \vdash t \cdot S(u) &= t \cdot u + t. \end{aligned}$$

These are rules of the arithmetic language. Their validity in the constructed carrier was established in Part I. Writing them as rules does not erase the construction that licenses their interpretation.

Induction has the form

$$\frac{\Gamma \vdash \varphi(0) \quad \Gamma \vdash \forall x(\varphi(x) \rightarrow \varphi(S(x)))}{\Gamma \vdash \forall x \varphi(x)}. \quad (5.2)$$

Here the bound induction variable is fresh relative to the hypotheses. Other parameters of φ may remain free. The language permits this rule for every formula; the next chapter records separately whether the formula itself contains a quantifier.

Combining and using propositions

Implication is introduced by discharging a temporary hypothesis and used by applying it:

$$\frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \rightarrow \psi}, \quad \frac{\Gamma \vdash \varphi \rightarrow \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi}.$$

The comma means that the indicated hypothesis has been added to the context. For conjunction, proofs of both components give a proof of their conjunction; either component can then be recovered:

$$\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi}, \quad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \varphi}, \quad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \psi}.$$

For disjunction, a proof of either component gives the disjunction. To use a disjunction, both cases must lead to the same conclusion:

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \varphi \vee \psi}, \quad \frac{\Gamma \vdash \psi}{\Gamma \vdash \varphi \vee \psi},$$

$$\frac{\Gamma \vdash \varphi \vee \psi \quad \Gamma, \varphi \vdash \chi \quad \Gamma, \psi \vdash \chi}{\Gamma \vdash \chi}.$$

Finally, from a derivation of $\perp$ one may infer any specified formula. This is the rule for an impossible case. None of these rules supplies a proof of $\varphi \vee \neg\varphi$ for an arbitrary φ .

Quantifiers and a genuinely arbitrary value

Universal introduction proves a formula at a fresh variable; universal elimination substitutes a term:

$$\frac{\Gamma \uparrow \vdash \varphi}{\Gamma \vdash \forall \varphi}, \quad \frac{\Gamma \vdash \forall \varphi}{\Gamma \vdash \varphi(t)}.$$

Every formula in $\Gamma \uparrow$ has been lifted over the fresh variable. The hypotheses therefore cannot constrain that variable. This is the exact force of calling it arbitrary.

Existential introduction exhibits a witness. Existential elimination uses a fresh witness without letting it escape into the conclusion:

$$\frac{\Gamma \vdash \varphi(t)}{\Gamma \vdash \exists \varphi}, \quad \frac{\Gamma \vdash \exists \varphi \quad \Gamma \uparrow, \varphi \vdash \psi \uparrow}{\Gamma \vdash \psi}.$$

The last condition matters. The conclusion must be the lifted form of a formula from the original context, so it cannot depend on the particular witness chosen inside the branch.

Together these are the base arithmetic rules. A derivation is a finite tree assembled from them, with matching premises at every junction. Its validity as a derivation is a question about expressions and rules. To connect it to arithmetic truth, we still need an interpretation theorem.

5.4 Reading the expressions as arithmetic

We use the ordinary natural-number reading proved faithful in Chapter 1. An *environment* ρ assigns a natural number to each free variable position. Write $[n, \rho]$ for the extended environment with

$$[n, \rho](0) = n, \quad [n, \rho](k+1) = \rho(k).$$

The new variable receives n and the earlier assignments are retained.

Definition 5.3 (Evaluation and satisfaction). The value $E_\rho(t)$ of a term is defined recursively:

$$\begin{aligned} E_\rho(x_k) &= \rho(k), & E_\rho(0) &= 0, \\ E_\rho(S(t)) &= E_\rho(t) + 1, & E_\rho(t + u) &= E_\rho(t) + E_\rho(u), \\ E_\rho(t \cdot u) &= E_\rho(t)E_\rho(u). \end{aligned}$$

Write $\rho \models \varphi$ when the formula is satisfied. Equality is interpreted as equality of term values and $\perp$ as falsehood. Conjunction, disjunction and implication are interpreted by their corresponding constructive logical operations. Quantifiers are read by

$$\begin{aligned} \rho \models \forall \varphi &\iff \text{for every } n \in \mathbb{N}, [n, \rho] \models \varphi, \\ \rho \models \exists \varphi &\iff \text{there is } n \in \mathbb{N} \text{ with } [n, \rho] \models \varphi. \end{aligned}$$

An environment satisfies a context when it satisfies every formula in it.

The formulas of the language are finite objects. The assertions that they are satisfied are made in the surrounding mathematics. That surrounding reasoning is called the *metatheory*. It supplies the finite inductive constructions, equality and logical reasoning being used to study the language. We have made this background visible; we have not eliminated it.

Three identities ensure that manipulating an expression corresponds to manipulating its interpretation.

Lemma 5.4 (Binding respects interpretation). *For every formula φ , term t , environment ρ and natural number n ,*

$$[n, \rho] \models \varphi \uparrow \iff \rho \models \varphi, \quad (5.3)$$

$$\rho \models \varphi(t) \iff [E_\rho(t), \rho] \models \varphi. \quad (5.4)$$

Replacing the induction variable by its successor correspondingly replaces its assigned value n by $n + 1$.

Proof. First check terms. Lifting a variable moves its position past the newly inserted entry, so its value remains its previous assignment. Substitution at position k has three cases: the variable is x_k , is below k , or is above k . Its value is respectively the substituted term's value, the unchanged earlier assignment, or the assignment shifted down one position. These are exactly the environment changes specified by the operations. Zero is unchanged; the successor, sum and product cases follow by applying their evaluation clauses to the already established identities for their constituent terms. This proves the term identities by structural induction.

Now induct on formulas. An equality uses the term identities on both sides. Falsehood is unchanged. Each propositional connective applies to the corresponding already established interpretations of its components. Under a quantifier, the new bound position receives its own value and the cutoff increases by one. For substitution, lifting the inserted term keeps its value independent of this newly bound position. Thus the extended environments agree at position zero and at every later position. For a universal formula this argument applies to every assigned value; for an existential formula it applies to its witness, in both directions. These are all the formula constructors, proving the two equivalences.

For the final assertion, the inserted term $S(x_0)$ has value $n + 1$ when x_0 has value n . The other free positions remain fixed by the lifting and substitution convention. Apply the substitution identity. $\square$

This is the bookkeeping behind the freshness conditions. Its purpose is to make the soundness proof about the same assignments before and after a rule is applied.

5.5 Why every permitted step preserves truth

Theorem 5.5 (Soundness of the base arithmetic rules). *Suppose a finite derivation of $\Gamma \vdash \varphi$ uses the base rules listed in this chapter. If ρ satisfies every hypothesis in Γ , then $\rho \models \varphi$.*

Proof. Induct on the derivation tree, keeping the context and environment arbitrary. For each final rule, the induction hypotheses say that its premise derivations preserve satisfaction in their respective contexts. We check every kind of rule.

Hypothesis and equality. A selected hypothesis is true because the whole context is satisfied. Reflexivity evaluates to equality of a value with itself. For substitution, the first premise gives $E_\rho(t) = E_\rho(s)$. The second, by Lemma 5.4, gives satisfaction of the hole formula with $E_\rho(t)$ in its first position. Replace that equal value by $E_\rho(s)$ and use the same lemma in reverse.

The arithmetic rules. A successor value is not zero, and equality of successor values implies equality of the preceding values. The four operation equations evaluate to the recursive arithmetic identities. Thus these rule conclusions are satisfied. In the injectivity rule the premise supplies the required equality of successors.

Implication. To introduce an implication, assume the antecedent is satisfied. The augmented context is then satisfied, so the induction hypothesis for the branch gives the consequent. This proves the interpreted implication. To eliminate one, the premise interpretations give an implication and its antecedent; applying the former to the latter gives the consequent.

Conjunction, disjunction and falsehood. Conjunction introduction combines the two satisfied components; either elimination selects the required component. A disjunction introduction supplies its proved side. In disjunction elimination, the interpreted disjunction gives one of its two alternatives. Add that alternative to the satisfied context and apply the induction hypothesis for the corresponding branch. Both branches have the same concluding formula. If the premise is falsehood, its interpreted truth is impossible, and the falsehood rule is sound.

Universal quantification. For introduction, choose an arbitrary natural number n and extend ρ to $[n, \rho]$. Every lifted hypothesis is satisfied by (5.3). The induction hypothesis therefore gives the body of the universal formula at n . Since n was arbitrary, the universal statement is satisfied. For elimination, apply the interpreted universal statement to $E_\rho(t)$, then use (5.4) to obtain the desired instance.

Existential quantification. Introduction uses $E_\rho(t)$ as its witness; the substitution identity supplies the body at that witness. For elimination, the first premise supplies a witness n . The extended environment satisfies the lifted original hypotheses and the existential body. The branch induction hypothesis gives $\psi \uparrow$ in this environment. Equation (5.3) then returns ψ under the original ρ . This is where the restriction on the conclusion of existential elimination is used.

Induction. The first premise gives the body at zero. The second gives, at every n , an implication from the body at n to the body at $n + 1$; Lemma 5.4 identifies the successor instance. Induction in the surrounding natural-number mathematics now gives the body at every n . This is precisely satisfaction of the universal conclusion. The context remains satisfied throughout because its formulas do not acquire the fresh induction variable.

Every base rule has been covered. Since the derivation tree is finite, the induction reaches its conclusion. $\square$

The last case shows the scope of the result plainly. Soundness of the object-language induction rule was proved using induction in the metatheory. This is a proof about the stated language in its stated arithmetic interpretation. It is not an internal proof, written in that language, of its own unrestricted soundness.

The proof also shows which facts about the natural numbers it used. The arithmetic rules needed the recursive identities, separation of the start from every step, and injectivity of the step. The induction rule needed induction. The remaining cases used only that quantifiers range over the numbers. Section 5.7 uses this observation to show what each family of rules supplies.

5.6 The same induction, now inside the language

Chapter 3 proved $0 + a = a$ about the constructed operation. We can now display an object-language proof of its universal formula. Set $\varphi(x)$ to $0 + x = x$. The base is the rule $0 + 0 = 0$. For the step, temporarily assume $0 + x = x$. The recursive rule gives

$$0 + S(x) = S(0 + x).$$

Equality substitution changes the right side to $S(x)$. Discharge the temporary hypothesis and generalize the fresh variable to obtain

$$\forall x(0 + x = x \rightarrow 0 + S(x) = S(x)).$$

Apply (5.2). The conclusion is

$$\vdash \forall x(0 + x = x). \quad (5.5)$$

Soundness now reads this finite derivation as a true arithmetic statement. The formula on which we inducted was the equality $0 + x = x$, with no quantifier inside it. The quantifier in the conclusion does not change that fact. This distinction will matter for the induction record in the next chapter.

5.7 What each family of rules supplies

Besides the four recursive equations, which define the operations, the arithmetic rules fall into three families. *Separation* is the rule $\neg(S(t) = 0)$: a step never returns to the start. *Injectivity* is the rule from $S(t) = S(u)$ to $t = u$: equal successors have equal predecessors. *Induction* is rule (5.2). They are the object-language counterparts of the three conditions of Definition 2.4. We can now ask what each family contributes that the others cannot.

Begin with what needs no induction at all. The defining equations already settle every closed calculation.

Proposition 5.6 (Closed calculations need no induction). *Let φ be a quantifier-free formula with no free variables. If φ is true, then $\vdash \varphi$ has a derivation that uses no induction rule. If φ is false, then $\vdash \neg\varphi$ has such a derivation. Neither derivation uses an additional principle.*

Proof. First, every closed term is provably equal to the numeral of its value, with no induction. Proceed by recursion on the term in the metatheory. A numeral needs nothing; for $S(t)$, substitution replaces t by its numeral $\bar{n}$, and $S(\bar{n})$ is the numeral $\overline{n+1}$. For a sum, equality substitution replaces the summands by their numerals $\bar{m}$ and $\bar{n}$; then n uses of the successor equation for addition and one use of its zero equation turn $\bar{m} + \bar{n}$ into $\overline{m+n}$. Products are treated the same way with the multiplication equations, using the case of sums. Symmetry and transitivity of equality, each one use of equality substitution together with reflexivity, combine these steps.

The values of closed terms can be computed and compared, so each closed quantifier-free formula is either true or false. We prove both claims together by induction on φ . Take an equation $t = s$ with values m and n . If $m = n$, both sides are provably equal to the same numeral, so $t = s$ follows. If $m < n$, assume $t = s$, hence $\bar{m} = \bar{n}$. Injectivity, applied m times, gives $0 = \bar{n} - \bar{m}$. The right side is a successor, so separation refutes it; discharging the assumption proves $\neg(t = s)$. The case $m > n$ is symmetric. Falseness is refuted by $\perp \rightarrow \perp$, which discharges the hypothesis $\perp$. A true conjunction combines two derivations, and a false one refutes the false component. A true disjunction introduces its true side, and a false one refutes both sides by disjunction elimination. A true implication has a true consequent or a false antecedent, and either gives the implication; a false one has a true antecedent and a false consequent, which together refute it. No step introduces an induction rule. $\square$

So every closed instance of a familiar law, such as $0 + \bar{n} = \bar{n}$, $\bar{m} + \bar{n} = \bar{n} + \bar{m}$ or $\neg(S(\bar{n}) = \bar{n})$, has a derivation with no induction. The universal statements are a different matter. To show that something cannot be derived, we need an interpretation in which the permitted rules still preserve truth while the statement fails. The proof of Theorem 5.5 supplies one almost at once, because the particular facts about the natural numbers entered only at the arithmetic rules and at induction.

Definition 5.7 (A structure for the arithmetic language). A *structure* M consists of a carrier, an element 0_M , an operation S_M of one argument, and operations $+_M$ and $\cdot_M$ of two. Environments assign elements of the carrier to variable positions. Evaluation and satisfaction are given by the clauses of Definition 5.3 with M in place of $\mathbb{N}$, and quantifiers range over the carrier. Write $M, \rho \models \varphi$.

Theorem 5.8 (Soundness in other structures). *Let a finite derivation of $\Gamma \vdash \varphi$ use the base rules, and let ρ be an environment in a structure M that satisfies every formula of Γ . Then $M, \rho \models \varphi$ in each of the following cases.*

1. *The derivation contains no induction rule, and M satisfies the six arithmetic rules: $S_M(a) \neq 0_M$, S_M is injective, and the four recursive equations hold for all elements.*
2. *The derivation contains no separation rule, and M has exactly one element.*
3. *The derivation contains no injectivity rule, and M is the Boolean structure on $\{0, 1\}$ with $S(a) = 1$, $a + b = \max(a, b)$ and $a \cdot b = \min(a, b)$.*

Proof. Lemma 5.4 holds in every structure, with the same proof, because that proof used only the evaluation clauses. Now repeat the proof of Theorem 5.5. The hypothesis, equality,

propositional and quantifier cases go through unchanged, with quantifiers ranging over the carrier of M . Only the arithmetic rules and induction need attention.

In the first case each arithmetic rule is true in M by hypothesis, and the induction case does not arise.

In the second case every equation is satisfied, so reflexivity, substitution, injectivity and the four recursive equations all hold. Induction holds as well, since its base already gives the body at the only element. Separation is the one rule that fails, and it does not occur.

In the third case check the equations directly:

$$\begin{aligned} a + 0 &= a, & a + S(b) &= \max(a, 1) = 1 = S(a + b), \\ a \cdot 0 &= 0, & a \cdot S(b) &= \min(a, 1) = a = \max(\min(a, b), a) = a \cdot b + a. \end{aligned}$$

Separation holds because $S(a) = 1 \neq 0$. Induction holds because the base gives the body at 0, and one use of the step gives it at $1 = S(0)$, the only other element. Injectivity fails, since $S(0) = S(1)$, and it does not occur. $\square$

The first case is the interesting one. Let W consist of the natural numbers together with two further elements a and b . On natural numbers the operations are the usual ones. In addition,

$$S(a) = a, \quad S(b) = b, \quad a + n = a, \quad b + n = b, \quad x + a = x + b = b$$

for every natural number n and every element x . A product is 0 when either factor is 0, the usual product when both factors are natural numbers, and b in every other case.

The six arithmetic rules hold in W . No element steps to 0, and S is injective because a and b are fixed points that no natural number reaches. For $x + 0 = x$ there is nothing new to check. For $x + S(y) = S(x + y)$, the new cases are of two kinds: if x is a or b and y is a number, both sides are x ; if y is a or b , then $S(y) = y$ and both sides are b . The rule $x \cdot 0 = 0$ is built into the table. For $x \cdot S(y) = x \cdot y + x$, both sides are 0 when $x = 0$; when $x \neq 0$ and x or y is a or b , both sides are b . In W , three familiar laws fail:

$$S(a) = a, \quad 0 + a = b \neq a, \quad a + 0 = a \neq b = 0 + a.$$

Corollary 5.9 (What only induction supplies). *None of the formulas*

$$\forall x \neg(S(x) = x), \quad \forall x(0 + x = x), \quad \forall x \forall y(x + y = y + x)$$

has a derivation from the base rules without an induction rule. Each has a derivation whose induction formulas are all quantifier-free, and each of their closed instances has a derivation with no induction.

Proof. An induction-free derivation of any of them would make it true in W , by the first case of Theorem 5.8; the displayed values refute each one. The derivation of $\forall x(0 + x = x)$ is (5.5). For $\forall x \neg(S(x) = x)$, induct on the quantifier-free formula $\neg(S(x) = x)$: separation gives the base, and in the step injectivity turns $S(S(x)) = S(x)$ into $S(x) = x$, contradicting the hypothesis. Commutativity follows the proofs of Lemma 3.2 and Theorem 3.3, with each induction on an equation in which the other variable is a parameter. The closed instances are covered by Proposition 5.6. $\square$

The other two cases of Theorem 5.8 complete the picture. Every equation holds in the one-point structure, so no derivation from no hypotheses that avoids separation proves any formula $\neg(t = s)$, not even $\neg(S(0) = 0)$. In the Boolean structure, $S(S(0))$ and $S(0)$ both have value 1, so no derivation from no hypotheses that avoids injectivity proves $\neg(S(S(0)) = S(0))$. With all three families present, each of these statements has a derivation with no additional principle and only quantifier-free induction. Each family therefore supplies a statement that the other two cannot:

Family removed	A structure satisfying the rest	A statement lost
separation	one element	$\neg(S(0) = 0)$
injectivity	the Boolean structure	$\neg(S(S(0)) = S(0))$
induction	W	$\forall x \neg(S(x) = x)$

The same method shows that the statement “every number is zero or a successor”, $\forall x(x = 0 \vee \exists y(x = S(y)))$, needs induction. Let C consist of the natural numbers together with a second chain $c_0, c_1, c_2, \dots$, with $S(c_k) = c_{k+1}$. Set $x + c_0 = c_0$ and $x \cdot c_0 = c_0$ for every element x , and let the recursive equations $x + 0 = x$, $x + S(y) = S(x + y)$, $x \cdot 0 = 0$ and $x \cdot S(y) = x \cdot y + x$ determine every other sum and product. Every element is reached from 0 or from c_0 by steps, so this defines both operations, and the four recursive equations hold by construction. Separation and injectivity hold because S moves one place along each chain. The element c_0 is neither zero nor a successor, so the statement fails in C , and by the first case of Theorem 5.8 it has no derivation without induction. It has one by induction on the formula $x = 0 \vee \exists y(x = S(y))$, which contains a quantifier. The structure W , by contrast, satisfies the statement, since $a = S(a)$ and $b = S(b)$. Corollary 5.9 therefore survives when the statement is added as a hypothesis. In this language, then, induction supplies the passage from all the instances of a law to the universal law itself.

Remark 5.10. Chapter 6 adds rules for excluded middle and two related principles. Every formula has a decided truth value in the one-point and Boolean structures, so those rules are true there, and the second and third cases of Theorem 5.8 extend to derivations that use them. Corollary 5.9 also extends, with or without the hypothesis that every number is zero or a successor. With excluded middle in the metatheory the added rules are true in W . Without it, one reads disjunction and existence in W through double negation. Induction-free derivations remain sound under that reading, with or without the added rules, and because equality in W is decidable the reading agrees with ordinary truth on the three universal formulas of the corollary. With that hypothesis and the excluded-middle rule, the induction-free rules present Robinson’s arithmetic, so Robinson’s arithmetic proves none of the three formulas.

We now have both sides of the connection. A derivation gives a finite record of permitted reasoning; an interpretation gives arithmetic content to its expressions; soundness proves that the first preserves the second. The next task is to retain this connection when the rules are enlarged.

Questions to work through

1. In (5.1), identify the hole formula and the two terms substituted into it. Which rule supplies each premise?
2. Explain why a fresh variable may receive an arbitrary value in the universal-introduction case of the soundness proof. Where is the same fact used for existential elimination?
3. Locate the two inductions in the soundness argument: induction on the derivation and the induction used to interpret the induction rule. What is the object of each?
4. In (5.5), distinguish the formula used for induction from the formula finally proved. Does the first contain a quantifier?
5. Check the six arithmetic rules in the structure W . Once $S(a) = a$, which values can $x + a$ take, given the rule $x + S(y) = S(x + y)$ at $y = a$? Why would a structure with only the one new element a fail to refute $\forall x(0 + x = x)$?
6. Give an induction-free derivation of $0 + S(S(0)) = S(S(0))$. Explain why no induction-free derivation of $\forall x(0 + x = x)$ exists, although every instance has one.
7. Which rule fails in the Boolean structure? Explain why no derivation from no hypotheses that avoids that rule can prove $\neg(S(S(0)) = S(0))$, and give a derivation that uses it.

Chapter 6

The price of an additional principle

Consider the formula

$$(0 = 0) \vee \neg(0 = 0).$$

One way to obtain it is to invoke excluded middle: every formula holds or its negation holds. Another way is to prove $0 = 0$ by reflexivity and then introduce the left side of the disjunction. The second argument never asks for a decision about an arbitrary formula.

The conclusion is identical. The resources used by the two arguments are different. If we record the first argument faithfully, excluded middle appears in its record. It would be a mistake to infer that the conclusion therefore requires excluded middle. The second argument has already answered that question.

This chapter makes resource accounting precise. We will distinguish the principles a derivation uses, the conditions under which its conclusion is true, and the separate work needed to prove that a principle is unavoidable for a whole mathematical assertion. We then carry out that work for the three additional rules themselves. With induction on every formula they generate exactly two theories, which we name; without induction, or with induction on quantifier-free formulas only, they mark four strengths.

6.1 A witness, a decision, and a double negation

To establish $\exists n P(n)$ constructively, one supplies a value of n together with the required evidence for $P(n)$. To establish $\neg\neg\exists n P(n)$, one gives a contradiction from the assumption that there is no such value. The second form does not itself display a witness. A rule that converts it to the first is therefore a substantive addition to the permitted steps.

Similarly, deciding each individual instance $P(n)$ does not include a specified bound on where a successful instance might occur. Finite inspection of initial values and a decision about the entire unbounded predicate are different tasks. This difference motivates the following principles.

Definition 6.1 (Three named additional principles). *Excluded middle*, abbreviated EM, states

$$P \vee \neg P \quad \text{for every proposition } P.$$

The *limited principle of omniscience*, in the predicate form used here and abbreviated LPO, states

$$[\forall n (P(n) \vee \neg P(n))] \longrightarrow [(\exists n P(n)) \vee (\forall n \neg P(n))]. \quad (6.1)$$

Markov's principle, abbreviated MP, states

$$[\forall n (P(n) \vee \neg P(n))] \longrightarrow [\neg\neg\exists n P(n) \longrightarrow \exists n P(n)]. \quad (6.2)$$

Here n ranges over the natural numbers and P is an arbitrary predicate in the surrounding mathematics.

EM concerns every proposition. LPO concerns a family with an individual decision for every member and returns a global alternative. MP uses such individual decisions together with a double-negated existence assertion to obtain existence. Stating these principles separately lets a theorem say which one it actually uses.

There is also an object-language version of each principle. To the arithmetic rules of Chapter 5, we may add the schemas

$$\Gamma \vdash \varphi \vee \neg\varphi, \quad \text{EM}, \quad (6.3)$$

$$\begin{aligned} \Gamma \vdash (\forall x(\varphi(x) \vee \neg\varphi(x))) \\ \rightarrow ((\exists x\varphi(x)) \vee (\forall x\neg\varphi(x))), \end{aligned} \quad \text{LPO}, \quad (6.4)$$

$$\Gamma \vdash \neg\neg\exists x\varphi(x) \rightarrow \exists x\varphi(x), \quad \text{MP}, \quad (6.5)$$

where the MP rule is permitted only when φ is quantifier-free. These are schemas over the finite formulas of the arithmetic language. Definition 6.1 concerns propositions and predicates in the metatheory. The latter principles suffice to interpret the former rules; the two levels should not be silently identified. Section 6.7 shows how far apart they are. With induction on every formula, the MP schema permits nothing new and the LPO schema proves every instance of the EM schema. As principles about arbitrary predicates in the metatheory, the three keep the separate strengths they have in constructive mathematics [4].

Why does the object-language MP rule omit an explicit premise giving individual decisions? Its restriction supplies that premise.

Lemma 6.2 (Quantifier-free formulas can be decided). *If φ contains no quantifier, then for every environment ρ ,*

$$(\rho \models \varphi) \vee \neg(\rho \models \varphi).$$

Proof. Induct on the formula. For equality, evaluate the two terms and compare their natural-number values. Falsehood has the negative answer. For a conjunction, decide its two components: both positive answers give the conjunction, and a negative answer to either refutes it. For a disjunction, a positive answer to either supplies a side; two negative answers refute both possible sides.

For an implication $A \rightarrow B$, first decide B . If B holds, the implication follows. If B does not hold, decide A . A proof of A then refutes the implication, since applying it would prove B ; a refutation of A proves the implication by eliminating the impossible antecedent. These are all cases because the formula has no quantifier. $\square$

The lemma decides a fixed finite formula at a fixed environment. By itself it says nothing about an existential formula, which has infinitely many instances. The restriction on the MP schema is part of its mathematical statement, and replacing it by an unrestricted schema would change the rule being studied. The restriction also has a consequence that is not visible here. In this language an existential formula over a quantifier-free body can be decided by a finite search, and Section 6.7 shows that the restricted rule therefore permits nothing new once induction on every formula is allowed.

6.2 A ledger for a finite argument

The word *cost* in this part of the book means a recorded logical resource. It is neither elapsed time nor an energy quantity. The later physical chapters introduce their own physical resources with their units and hypotheses.

Definition 6.3 (The four-entry ledger). A ledger is a tuple

$$O = (e, \ell, m; i), \quad e, \ell, m, i \in \{\text{false}, \text{true}\}.$$

The first three entries record use of EM, LPO and MP. The final entry records whether an induction formula with a quantifier was used. The empty ledger has all four entries false. Ledgers combine by taking the Boolean inclusive-or separately in each entry; write this combination as $O_1 \sqcup O_2$.

A base rule contributes no entry of its own, with one stated exception: an induction rule contributes the final entry when its induction formula contains a quantifier. An EM, LPO or MP rule contributes its corresponding entry. At a rule with premises, combine all their ledgers and the rule's own contribution.

This defines the ledger of a finite derivation by following its tree. An entry says that the resource occurred at least once. It does not count how often it occurred. In particular, using the same resource twice does not change its Boolean entry after the first use.

Proposition 6.4 (Resources cannot disappear in combination). *A named principle present in either input ledger is present in their combination. Moreover,*

$$O_1 \sqcup O_2 = \emptyset \iff O_1 = \emptyset \text{ and } O_2 = \emptyset,$$

where $\emptyset$ denotes the empty four-entry ledger.

Proof. Each output entry is the inclusive-or of the corresponding input entries. It is true whenever either input is true, and false exactly when both inputs are false. Apply this observation to all four entries. $\square$

No rearrangement of the surrounding base rules erases an occurrence already recorded in a premise. Removing a resource requires changing the derivation itself, for example by replacing a branch with a proof that avoids the additional rule.

6.3 No added principle is not the same as an empty ledger

We need two descriptions. A derivation is *posit-free* when it uses none of the three additional principles. Its first three entries are then false. A derivation has an *empty ledger* only when its fourth entry is also false. Thus empty-ledger proofs restrict the induction formulas to the quantifier-free tier, while posit-free proofs may use induction on quantified formulas.

In this accounting convention, the verdict “forced” tests the first three entries. It is accompanied by the induction tier. The verdict is relative to the stated base rules and to this alphabet of

additional principles. Reading it as “no background assumptions of any kind” would discard the language, the base rules and the metatheory that made the argument possible.

The distinction can be seen directly in examples already available. The proof of $1 + 1 = 2$ has no induction and an empty ledger. The proof of $\forall x(0 + x = x)$ uses induction, but its induction formula is the quantifier-free equality $0 + x = x$. Its ledger is also empty.

For a contrasting induction, let φ be $\forall y(y = y)$, independent of the induction variable x . Its base follows by reflexivity and universal introduction. Its step is the identity implication $\varphi \rightarrow \varphi$, generalized over x . Induction gives $\forall x\forall y(y = y)$. This derivation posts the final entry because φ contains a quantifier, even though no EM, LPO or MP rule occurs. The record describes this particular use of induction; it is not a claim that this trivial conclusion needs that use.

6.4 The record agrees with the derivation

There are two ways to obtain the resource record. One is to carry it through each valid inference while verifying the premises and conclusion. The other is to ignore those formulas and simply inspect the rule labels and induction formulas in the whole tree. The latter inspection by itself does not establish that the tree is a valid derivation. On a valid tree, the two records agree exactly.

Theorem 6.5 (Agreement of resource accounting). *For every accepted finite derivation, its threaded ledger equals the ledger obtained by inspecting its rule occurrences. Consequently its first three entries are false if and only if the tree contains no EM, LPO or MP rule. Its induction entry is true if and only if it contains induction on a formula with a quantifier.*

Proof. Induct on the tree. At a hypothesis, equality-reflexivity or arithmetic leaf, both records are empty. At an additional-principle leaf, both contain exactly its named entry. A one-premise base rule carries its premise ledger unchanged in both accounts. A two-premise rule combines the two records by $\sqcup$; disjunction elimination combines its three records. The induction hypotheses identify every premise record, so the combined records agree.

At an induction node, both accounts combine the base and step records. They then inspect the same finite formula: if it is quantifier-free, neither adds an entry; if it has a quantifier, both add the final entry. This covers all rules and proves equality of the two accounts.

For the consequences, an inclusive-or over finitely many occurrences is false exactly when all its inputs are false, by repeated application of Proposition 6.4. Therefore absence of the three posit entries is equivalent to absence of their three rule kinds. The same reasoning, with the induction formula test, gives the characterization of the final entry. $\square$

The acceptance hypothesis cannot be dropped when interpreting the record as a proof’s resources. A tree with mismatched premise formulas can have a perfectly clear list of rule labels. That list does not repair the invalid inference.

6.5 Soundness with the prices attached

A ledger has a semantic use as well as a syntactic one. It tells us which additional metatheoretic principles suffice to justify a derivation.

Theorem 6.6 (Conditional soundness). *Let a derivation of $\Gamma \vdash \varphi$ have ledger O . Assume the metatheoretic EM principle if its EM entry is true, LPO if its LPO entry is true, and MP if its MP entry is true. Every environment satisfying Γ then satisfies φ .*

Proof. Use the derivation induction of Theorem 5.5. All base-rule cases are unchanged. The principle assumptions required for a premise are available because a true premise entry remains true in the conclusion ledger, by Proposition 6.4. It remains to check the three new leaves.

At an EM leaf, apply the supplied EM principle to the proposition $\rho \models \varphi$. This gives precisely the interpretation of $\varphi \vee \neg\varphi$.

At an LPO leaf, suppose the interpreted premise says that each instance of the body is decidable. Apply the supplied LPO principle to

$$P(n) \quad \text{defined by} \quad [n, \rho] \models \varphi.$$

Its existential alternative is the interpretation of $\exists\varphi$; its universal negative alternative is the interpretation of $\forall\neg\varphi$. This is the required implication.

At an MP leaf, the rule requires the body to be quantifier-free. Lemma 6.2 supplies a decision for each instance of the same predicate $P(n)$. The supplied MP principle then converts the double-negated existential premise into the existential conclusion.

The induction entry requires no further principle in this semantic theorem: the surrounding natural-number mathematics already supplies induction on the interpreted formula, as explicitly used in the base soundness proof. Its use inside the derivation remains recorded. $\square$

Corollary 6.7 (Soundness without the three additional principles). *For a posit-free derivation, conditional soundness requires none of EM, LPO or MP. This holds at either recorded induction tier.*

Proof. All three principle entries are false, so none of the three conditional assumptions in Theorem 6.6 is required. $\square$

This establishes sufficiency, with the source of each permission visible. It does not say that every conclusion with an EM-labelled proof is unprovable without EM. The opening example proves the contrary. The assumptions in Theorem 6.6 can also be weakened: Corollary 6.15 shows that an MP entry requires no principle at all, and that an EM or LPO entry requires only the metatheoretic LPO for the predicates that formulas define, which is the least assumption under which every derivation with such an entry is sound.

6.6 A route's price and a theorem's exact strength

The two proofs of $(0 = 0) \vee \neg(0 = 0)$ give a useful complete comparison. The EM-rule proof has ledger

(true, false, false; false).

The reflexivity-and-disjunction proof has the empty ledger. Both have the same conclusion. The record is exact about each tree, but the first record overstates what is necessary to prove that conclusion.

The rule occurrence also retains more information than its Boolean entry: it specifies the particular formula to which the principle was applied. In this case that formula is $0 = 0$, whose positive side is already provable. Forgetting the formula and retaining only the name EM loses the reason that this occurrence can be discharged.

There is a second limitation. The named alphabet in this chapter contains three principles. It is a useful way to report their use, but three names need not mark three strengths, and no fixed alphabet can distinguish every intermediate principle someone might formulate. Over the base rules with induction on every formula, the three names mark exactly two strengths (Theorem 6.11). A conclusion's intrinsic logical strength is a question about derivability and reductions over a specified base, not merely about which names a chosen record can print.

To prove an *exact* logical cost, two directions are needed. First, derive the mathematical assertion using the proposed principle. Second, show that the assertion itself yields that principle over the same retained base. The second direction is a *reversal*. It supplies the lower bound that inspection of one proof cannot supply.

6.7 What the three rules cost

We now apply the two-direction test to the three rules themselves. Say that a rule *reduces* to a set of rules when every instance of its schema has a derivation, from no hypotheses, whose additional rules all belong to that set. A rule that reduces to a set can be removed from any derivation. Replace each of its leaves by the derivation of the same instance, with the leaf's hypotheses added to its context. A derivation stays valid when unused hypotheses are added, each lifted wherever a quantifier rule lifts the context; an induction on the derivation proves this, because no rule requires a hypothesis to be absent. The conclusion and the context of the whole derivation do not change. Its ledger can change in the induction entry, since some replacement derivations below use induction on formulas with quantifiers.

The answers depend on which induction rules the replacement derivations may use. We first allow induction on every formula.

The restricted Markov rule permits nothing new

The MP rule applies only to a quantifier-free body. In this language such a body cannot describe a search whose outcome is unknown. The reason is a fact about polynomials.

Lemma 6.8 (Quantifier-free formulas are eventually constant). *Let φ be quantifier-free and let ρ be an environment. There is a number B , computed from ρ by evaluating terms, such that*

$$[n, \rho] \models \varphi \iff [B, \rho] \models \varphi \quad \text{for every } n \geq B.$$

Proof. Fix ρ . Each term t has the value

$$p_t(n) = c_0 + c_1n + \dots + c_kn^k$$

under $[n, \rho]$, for natural-number coefficients c_i that do not depend on n . Induct on t : the variable x_0 gives the polynomial n ; another variable and 0 give constants; S adds one to c_0 ; sums and products of such polynomials are again such polynomials. Each coefficient is the value of a term in the other variables, built by the same recursion.

Let B be one more than the sum of all coefficients of both sides of every equation in φ . If every coefficient is 0, every equation holds at every n and there is nothing to prove. Otherwise $B \geq 2$. For $n \geq B$ every coefficient is less than n , so $c_0 + c_1n + \dots + c_kn^k$ is the base- n expansion with digits $c_0, \dots, c_k$. Base- n expansions are unique. Hence an equation $t = s$ holds at such an n exactly when t and s have the same coefficients, after the shorter list is padded with zeros. That condition does not mention n . Each equation in φ therefore has one truth value for all $n \geq B$, and so does every combination of equations by $\wedge$, $\vee$, $\rightarrow$ and $\perp$. $\square$

Take, for example, $\varphi(x)$ to be $x + x = y$, with x the searched variable and y a parameter. The left side has coefficients $(0, 2)$ and the right side has (y) , so $B = y + 3$. When $n \geq y + 3$ the digit lists $(0, 2)$ and $(y, 0)$ differ, and the equation fails. Whether some n satisfies it is settled by checking $n = 0, 1, \dots, y + 3$.

Theorem 6.9 (The restricted Markov rule costs nothing). *Let φ be quantifier-free, let ρ be an environment, and let B be the bound of Lemma 6.8.*

1. $\rho \models \exists\varphi$ holds exactly when $[n, \rho] \models \varphi$ for some $n \leq B$. It is therefore decided by finitely many applications of Lemma 6.2.
2. Consequently $\rho \models \neg\neg\exists\varphi \rightarrow \exists\varphi$ holds with none of EM, LPO or MP in the metatheory.
3. Every instance of the MP schema has a derivation from the base rules, with induction on every formula permitted, and with no EM, LPO or MP rule. The derivation constructed below uses induction on formulas that contain quantifiers.

Proof. For the first assertion, a witness $n > B$ can be replaced by B , by the lemma; then examine $n = 0, 1, \dots, B$, deciding each instance by Lemma 6.2. For the second, run that search. If it finds an instance, the existential holds. If it finds none, the first assertion gives $\neg\exists\varphi$, which contradicts the double-negated premise, and the impossible case gives the conclusion.

The third assertion carries out the same argument inside the language; we give its outline. Write $x < m$ for $\exists z(x + S(z) = m)$, and let T be the term whose value is B . The derivation proves, by inductions, that every term equals its polynomial form in x_0 ; that two polynomial forms whose coefficients lie below the base are equal exactly when their coefficients are; that $\varphi(T + z)$ and $\varphi(T)$ are equivalent for every z ; and that for every bound m , either some x satisfies φ or no $x < m$ does. The last of these is an induction on a formula with quantifiers. Apply it at $m = T$. In the second case decide $\varphi(T)$. If it holds, T is a witness. If it fails, it fails at every $T + z$, and every x is either below T or of the form $T + z$, a fact also proved by induction. Together these give $(\exists x \varphi(x)) \vee (\forall x \neg \varphi(x))$. The second alternative contradicts the premise $\neg \neg \exists x \varphi(x)$, so the MP instance follows. $\square$

The theorem explains why, once induction on every formula is allowed, the restriction leaves nothing for Markov's principle to do in this language. Markov's principle concerns a search that is decidable at each step but whose stopping point is unknown, such as the search for the step at which a computation halts. A quantifier-free body comes with a bound, computed from its parameters, beyond which nothing changes, so its search is bounded in advance. No quantifier-free formula $\theta(e, n)$ can therefore say that n codes a halting computation of the program with code e on input e : a search up to the bound would then decide whether that program halts, and no program decides that [8]. Programs and their numerical codes are treated in Chapter 10. Describing a computation with only $0, S, +$ and $\cdot$ requires quantifiers. For the same reason, a version of LPO restricted to quantifier-free bodies would decide nothing new in the interpretation: its conclusion is settled by the same finite search. The principle of Definition 6.1, which applies to every decidable predicate of the metatheory, is a different matter.

The LPO rule proves excluded middle

The LPO rule has no such restriction. Its premise asks for a decision of every instance, and the rule itself supplies decisions for quantified formulas, one quantifier at a time.

Theorem 6.10 (The LPO and EM rules prove the same formulas). *Permit induction on every formula. For every formula φ and every context Γ , the judgment $\Gamma \vdash \varphi \vee \neg \varphi$ has a derivation from the base rules and the LPO rule, with no EM or MP rule. Conversely, every instance of the LPO schema has a derivation from the base rules and the EM rule, with no induction.*

Proof. Start with equality. Induction on x , with the induction formula $\forall y(x = y \vee \neg(x = y))$, proves

$$\vdash \forall x \forall y (x = y \vee \neg(x = y)). \quad (6.6)$$

The base and the step each split into cases according to whether y is 0 or a successor. An induction on y with a quantifier-free formula performs that split, since its step does not use its own hypothesis. In the base, $0 = 0$ holds by reflexivity and $\neg(0 = S(y))$ by separation. In the step, $\neg(S(x) = 0)$ holds by separation; for $S(x)$ and $S(y)$, the hypothesis decides $x = y$, and its two answers give $S(x) = S(y)$ by substitution and $\neg(S(x) = S(y))$ by injectivity.

Now prove $\Gamma \vdash \varphi \vee \neg \varphi$ for every formula φ and every context Γ together, by induction on φ . Allowing every context matters, because the quantifier cases pass to a lifted context.

Equations are instances of (6.6). *Falsehood* takes the right alternative, $\perp \rightarrow \perp$. For a *conjunction*, *disjunction* or *implication*, apply disjunction elimination to the decisions of its two components and argue by cases, exactly as in the proof of Lemma 6.2.

An existential formula $\exists x\psi$. The induction hypothesis in the lifted context, followed by universal introduction, gives $\forall x(\psi \vee \neg\psi)$. This is the premise of the LPO rule at ψ , which returns $(\exists x\psi) \vee (\forall x\neg\psi)$. In the second case existential elimination turns $\forall x\neg\psi$ into $\neg\exists x\psi$.

A universal formula $\forall x\psi$. From $\forall x(\psi \vee \neg\psi)$ obtain $\forall x(\neg\psi \vee \neg\neg\psi)$, since ψ gives $\neg\neg\psi$. The LPO rule at $\neg\psi$ returns $(\exists x\neg\psi) \vee (\forall x\neg\neg\psi)$. A witness of $\neg\psi$ refutes $\forall x\psi$. In the other case decide ψ at an arbitrary x : the negative answer contradicts $\neg\neg\psi$, so ψ holds, and universal introduction gives $\forall x\psi$.

Only the LPO rule introduces a principle in these derivations. For the converse, the EM rule at $\exists x\varphi$ gives $(\exists x\varphi) \vee \neg\exists x\varphi$, and $\neg\exists x\varphi$ gives $\forall x\neg\varphi$: assume φ at a fresh x , introduce the existential, and reach a contradiction. The premise of the LPO instance is not even used. $\square$

The only inductions in these derivations are those in the proof of (6.6), and they serve only the equations. The MP rule can do that job instead. For an equation $t = s$, let β be $t = s \vee \neg(t = s)$, placed under a quantifier whose variable does not occur in it. The formula $\neg\neg\beta$ has a derivation with no added rule: from $\neg\beta$ we obtain $\neg(t = s)$, hence β , a contradiction. So $\neg\neg\exists x\beta$ is derivable, and the MP rule at the quantifier-free β returns $\exists x\beta$, hence β . Thus excluded middle also follows from the LPO and MP rules together, with no induction at all.

Two strengths, or four

Theorem 6.11 (The price of each rule when induction is unrestricted). *Allow induction on every formula. Then:*

1. *MP reduces to the empty set of rules;*
2. *EM reduces to LPO, and LPO reduces to EM;*
3. *neither LPO nor EM reduces to MP, and in particular neither reduces to the empty set.*

Give MP level 0 and LPO and EM level 1, and give a set of rules the largest level of its members, or 0 if it is empty. A rule reduces to a set exactly when its level is at most the level of the set.

Proof. The first two assertions are Theorems 6.9 and 6.10. For the third it suffices to treat LPO: if EM reduced to MP, so would LPO, since LPO reduces to EM. We sketch the argument, which uses Kleene's realizability interpretation [4].

A realizer of a formula is a number coding a program that carries computational evidence for it. Any number realizes a true equation, and nothing realizes a false equation or $\perp$. A realizer of $\exists x\psi$ supplies a witness and a realizer of the instance; a realizer of $\psi \vee \chi$ says which side holds and realizes that side; realizers of $\psi \rightarrow \chi$ and $\forall x\psi$ are programs that convert realizers of ψ , or numbers, into realizers of χ , or of the instances. Every base rule preserves realizability, so every derivation from the base rules is realized by a program read off from the derivation. So is every MP instance: given a realizer of $\neg\neg\exists x\varphi$, search $n = 0, 1, 2, \dots$ for an n satisfying the

quantifier-free φ ; the double negation guarantees, in the classical metatheory used for this argument, that the search stops.

Now let $\theta(e, n)$ say that n codes a halting computation of the program with code e on input e . As we saw after Theorem 6.9, θ must contain quantifiers. The LPO instance at θ , closed over e , has a realizable premise: θ can be written so that checking a proposed computation record, and producing a realizer of each true instance, is a finite computation. A realizer of the instance would therefore compute, from e , which alternative holds, that is, whether program e halts on input e . No program does that [8]. So no derivation from the base rules and MP proves this instance.

For the final statement, check each set of rules against each rule using these facts, together with the observation that a rule reduces to any set that contains it. $\square$

In constructive mathematics the principles of Definition 6.1 form a chain of strictly increasing strength, MP below LPO below EM [4]. For the rules over this language, with induction on every formula, the chain collapses in two places: MP costs nothing, and LPO costs as much as EM. The theorem also leaves no freedom in assigning prices. Suppose each rule receives a degree in an ordered system with a least element and joins, so that a rule reducing to a set never has a degree above the join of the set's degrees, the empty set having the least element as its join. Then MP must receive the least degree and LPO must receive the degree of EM. The two-level chart is the finest such assignment: every other one is obtained from it by exactly one map that preserves the least element, joins and the degrees of the rules.

Remark 6.12 (Heyting arithmetic and Peano arithmetic). The two strengths have standard names, and both identifications are proved against presentations written with no ledger. The base rules of Chapter 5, with induction for every formula, have the language, the successor, addition and multiplication laws and the induction of *Heyting arithmetic*, the intuitionistic first-order theory of 0 , S , $+$ and $\cdot$ [4], and we use that name for the posit-free derivations. Chapter 8 shows that they are exactly the translations of derivations in the usual presentation of Heyting arithmetic, up to the way the injectivity of the successor is written (Corollary 8.12). In particular, from every finite context, the posit-free derivations prove exactly what Heyting arithmetic proves. With the EM rule added they correspond to *Peano arithmetic*, the classical theory. Theorem 8.11 in Chapter 8 proves this second identification: from every finite context, the derivations whose LPO and MP entries are false prove exactly what Peano arithmetic proves. Together with Theorem 6.11 this covers every combination of the three rules, with induction on every formula permitted. If MP may be used but neither EM nor LPO, removing the MP leaves of a derivation gives a posit-free derivation of the same judgment, so the derivable judgments are those of Heyting arithmetic. If EM or LPO may be used, removing the MP leaves and exchanging LPO leaves for EM derivations, or EM leaves for LPO derivations when only LPO is permitted, shows that the derivable judgments are exactly those of Peano arithmetic. The two theories differ, by the third assertion of Theorem 6.11.

Without any induction rule, the picture changes.

Proposition 6.13 (The price of each rule without induction). *Consider only derivations that contain no induction rule. Then EM reduces to LPO and MP together, and each of LPO and MP reduces to EM. LPO does not reduce to MP, MP does not reduce to LPO, and EM reduces to neither*

of them alone. The three rules therefore mark four strengths: no rule; MP; LPO; and EM, which is equivalent to LPO and MP together.

Proof sketch. The positive reductions are derivations already given. EM yields each LPO instance by the converse half of Theorem 6.10, and each MP instance by deciding $\exists x\varphi$. LPO and MP together yield EM by the remark after Theorem 6.10. None of these derivations uses induction. The realizability argument above shows that LPO does not reduce to MP, and hence that EM does not.

The remaining separations use a model with two stages, and the argument uses a classical metatheory. The later stage is $\mathbb{N}$. The earlier stage has the numbers $0, 1, 2, \dots$ and a primed copy $0', 1', 2', \dots$; erasing the prime maps it onto $\mathbb{N}$. The successor of n' is $(n+1)'$. A sum or product is computed from the erased values and carries a prime when either argument does, except that $x \cdot 0 = 0$. These operations satisfy the six arithmetic rules, and erasing primes preserves $0, S, +$ and $\cdot$. A formula holds at the earlier stage in the usual way, except that an implication or a universal statement must also hold after the passage to $\mathbb{N}$. Every induction-free derivation with no EM or MP rule holds at the earlier stage; the LPO rule holds there because every number in $\mathbb{N}$ is the image of an element. But $0' = 0 \vee \neg(0' = 0)$ fails there. The elements $0'$ and 0 differ at the earlier stage, yet $\neg(0' = 0)$ does not hold, because the two become equal when the prime is erased. So $\forall x(x = 0 \vee \neg(x = 0))$, a universal closure of an EM instance, has no induction-free derivation using only LPO. The MP instance that the remark after Theorem 6.10 uses to decide $x = 0$ fails at the earlier stage too, at $x = 0'$, so MP does not reduce to LPO. $\square$

Theorem 6.11 and Proposition 6.13 do not conflict: they concern different sets of permitted derivations. Induction on quantified formulas decides equality, the job that MP did in the remark after Theorem 6.10, and it derives every MP instance. Once MP costs nothing, the reduction of EM to LPO and MP becomes a reduction of EM to LPO, and the four strengths collapse to two. This is why the ledger records induction in an entry of its own: the price of a principle can depend on the induction available.

The empty ledger permits induction on quantifier-free formulas only. That much induction decides $x = 0$, by splitting into the zero and successor cases, but it does not decide equality in general, and the four strengths persist.

Proposition 6.14 (The price of each rule with quantifier-free induction). *Consider only derivations whose induction formulas are all quantifier-free, that is, derivations whose ledger has a false induction entry. Then the reductions are exactly those of Proposition 6.13: EM reduces to LPO and MP together, each of LPO and MP reduces to EM, LPO does not reduce to MP, MP does not reduce to LPO, and EM reduces to neither of them alone. Two explicit formulas witness the separations from LPO: the MP instance*

$$\neg\neg\exists x(x + y = z) \rightarrow \exists x(x + y = z)$$

has no derivation whose ledger has false EM, MP and induction entries, even with LPO permitted, and therefore none with an empty ledger; nor has $\forall x\forall y(x = y \vee \neg(x = y))$.

Proof sketch. The positive reductions are those of Proposition 6.13, whose derivations use no induction at all. The realizability argument in the proof of Theorem 6.11 applies to every

derivation from the base rules and MP, whatever its inductions, so LPO does not reduce to MP, and hence neither does EM. It remains to show that neither MP nor EM reduces to LPO. The two-stage model of Proposition 6.13 no longer serves, because quantifier-free induction proves $\forall x(x = 0 \vee \neg(x = 0))$. We use another model with two stages, again with a classical metatheory.

The later stage P consists of the polynomials in t with integer coefficients and nonnegative leading coefficient, with $0, x \mapsto x + 1$, sum and product of polynomials. A nonzero element of P has a positive leading coefficient, so leading terms never cancel: a sum has the larger degree of its summands, a product of nonzero elements the sum of their degrees, and P is closed under the operations. Its elements of degree 0 are the natural numbers, and subtracting 1 from an element of positive degree leaves an element of P . The six arithmetic rules hold: $x + 1 = 0$ has no solution in P , adding 1 is injective, and the recursive equations are identities of polynomials. Quantifier-free induction holds as well. Fix the other variables of a quantifier-free formula $\varphi(x)$. Each of its equations, read as an identity in x , either holds for every x or fails for all but finitely many, because a nonzero polynomial whose coefficients are integer polynomials has only finitely many roots. So φ takes one truth value at all but finitely many elements. If $\varphi(0)$ holds and each $\varphi(x)$ gives $\varphi(x + 1)$, then φ holds at every natural number, so that value is true. An element x of positive degree is reached by k steps from $x - k$, which lies in P and, for k large, outside the finite exceptional set; hence $\varphi(x)$ holds too.

The earlier stage consists of the pairs (u, v) of elements of P that are equal or both of positive degree, with the operations taken in each coordinate. Degrees never drop under sums, or under products with a nonzero factor, and a zero factor gives $(0, 0)$, so these pairs are closed under the operations. The passage to the later stage keeps the first coordinate. It preserves 0, the successor, sums and products, and it is onto. Formulas hold at the earlier stage as in Proposition 6.13: an implication or a universal statement must also hold after the passage. The six rules hold there, and the LPO rule holds because the passage is onto. Quantifier-free induction holds by the argument just given. Along the elements $(u - k, v - k)$, as along the natural numbers (n, n) , each equation of the formula is eventually true exactly when its two sides agree as polynomials in the induction variable in each coordinate, and after the passage exactly when they agree in the first. So the formula eventually takes the same value along both sequences; that value is true, and k steps carry it up to (u, v) . Hence every derivation whose ledger has false EM, MP and induction entries holds at the earlier stage.

Take $y = (t, t^2)$ and $z = (t^2, t)$. At the later stage $x = t^2 - t$ satisfies $x + t = t^2$, so $\neg\exists x(x + y = z)$ fails at both stages, and its negation holds at the earlier one. But no earlier pair (a, b) satisfies $(a, b) + (t, t^2) = (t^2, t)$, because $b = t - t^2$ has a negative leading coefficient. So the MP instance fails at the earlier stage. Likewise $x = y \vee \neg(x = y)$ fails there at $x = (t, t)$ and $y = (t, t^2)$: the two pairs differ, so $x = y$ does not hold, and they become equal after the passage, so $\neg(x = y)$ does not hold either. $\square$

The collapse to two strengths therefore needs induction on quantified formulas, the resource that the fourth entry of the ledger records.

What soundness needs

These theorems sharpen Theorem 6.6. That theorem assumes the metatheoretic MP for an MP entry and the metatheoretic EM for an EM entry. Neither assumption is needed in that form.

Corollary 6.15 (The metatheoretic price of the three rules). *Let a derivation of $\Gamma \vdash \varphi$ have ledger O . If the EM or the LPO entry of O is true, assume the metatheoretic LPO of Definition 6.1 for the predicates $n \mapsto ([n, \rho] \models \psi)$, one for each formula ψ and environment ρ ; otherwise assume nothing. Every environment satisfying Γ then satisfies φ . This assumption is the least possible. If every derivation whose additional rules are all LPO rules is sound, or every derivation whose additional rules are all EM rules is sound, then LPO holds for these predicates; and for them it is equivalent to excluded middle for the propositions $\rho \models \psi$.*

Proof. Follow the proof of Theorem 6.6. The base cases and the LPO leaves are unchanged, and only the EM and MP leaves need new arguments. At an MP leaf, the second assertion of Theorem 6.9 gives the interpreted instance with no principle. At an EM leaf we need $(\rho \models \varphi) \vee \neg(\rho \models \varphi)$. The metatheoretic LPO supplies this for every formula, by the argument of Theorem 6.10 read in the metatheory, by induction on the formula and for every environment at once. Equations are decided by comparing values, and the connectives preserve decisions. For $\exists\psi$, apply LPO to the predicate $[n, \rho] \models \psi$, which the induction hypothesis decides at every n . For $\forall\psi$, apply it to $[n, \rho] \models \neg\psi$, and use the decision of each instance to pass from $\forall n \neg\psi$ to $\forall n \psi$. Only predicates of the stated form occur, so LPO for them suffices, and the same argument shows that it gives excluded middle for every proposition $\rho \models \psi$.

For the converse, the LPO instance at ψ is a derivation from no hypotheses whose only additional rule is LPO. Its soundness at ρ is exactly LPO for the predicate $[n, \rho] \models \psi$. The EM instance at ψ gives, in the same way, $(\rho \models \psi) \vee \neg(\rho \models \psi)$; applied to $\exists\psi$, this excluded middle decides whether some n satisfies $[n, \rho] \models \psi$ or none does, which is LPO for that predicate. $\square$

For the formulas of this language, then, interpreting a derivation never needs excluded middle for arbitrary propositions. What the derivations with an EM or LPO entry need, taken together, is exactly LPO for the predicates that formulas define, which is the same as excluded middle for the propositions that formulas express. Excluded middle for arbitrary propositions is not implied by LPO [4]; formulas of the arithmetic language are not arbitrary propositions.

The next chapter carries the two-direction test to a concrete structural question: which identifications can a reader of repeated records make? Supplying an explicit identified pair, asserting only that some pair must exist, supplying only a decision procedure and asking for a yes-or-no answer, the full alternative or the parity of the index, and asking for a classification with no decision procedure lead to different theorem statements, and their costs are proved in both directions. Those costs are stated with metatheoretic principles: those of Definition 6.1, applied to arbitrary decidable predicates, and the weaker omniscience principles of Definition 7.8. At that level Markov's principle is an assumption that the classification genuinely needs: the classification implies it, and a model in Section 7.10 shows that intuitionistic second-order arithmetic proves neither. The restricted arithmetic rule of this chapter, by contrast, costs nothing once induction on every formula is allowed. The same section shows that no two of the principles priced in the next chapter are equivalent.

Questions to work through

1. Write the two derivations of $(0 = 0) \vee \neg(0 = 0)$ and annotate each node with its ledger. What does their comparison establish, and what would examining only the EM proof fail to establish?
2. Reproduce the implication case of Lemma 6.2. Explain why deciding its finite components does not decide an arbitrary existential formula.
3. Why does the quantified-induction example have a nonempty ledger while remaining posit-free? Identify the tested formula rather than looking only at the conclusion.
4. In the proof of conditional soundness, identify the use of the MP rule's quantifier-free restriction. What premise of the metatheoretic principle does it supply?
5. For $\varphi(x)$ given by $x \cdot x = y + y$, find the coefficient lists and the bound B of Lemma 6.8. Carry out the search for $y = 8$. Why does the lemma fail for the formula $\exists z(x = z + z)$, which contains a quantifier?
6. Derive an instance of the LPO schema from the EM rule. Where is existential introduction used, and what is the ledger of the result?
7. Locate the inductions in the proof of Theorem 6.10. Show how one MP instance, at a formula under a quantifier whose variable does not occur in it, replaces them for the equation $x = 0$. What is the ledger of the result?
8. Explain why Theorem 6.11 and Proposition 6.13 are consistent. What does induction on quantified formulas supply that separates them, and why does Proposition 6.14 show that induction on quantifier-free formulas does not supply it?

Chapter 7

What can recognize the same record?

A tally distinguishes successive counts. A reader of a tally may retain less. Two different counts can receive the same reading. If the reader also respects addition, that loss of distinction cannot occur arbitrarily: an identification must remain valid after the same further count is added to both sides.

This requirement is strong enough to classify every possible pattern of identification. The answer has a simple shape. Either every count remains distinct, or an initial segment is followed by a repeating part. The mathematical content lies in proving that no other shape is possible.

There is also a logical question. What information have we been given about the reader, and what do we ask the classification to return? An actual pair of identified counts provides more than the assertion that the reader cannot distinguish everything. A way to decide each pair provides more than an arbitrary relation. Asking only whether the reader is faithful asks for less than its index and period. Keeping these differences visible leads to six precise versions of the classification. One needs no additional principle, and each of the other five is equivalent to exactly one named logical principle. The six principles, counting the empty one, are pairwise different over intuitionistic second-order arithmetic (Section 7.10), so no version sits at two of them.

We use natural-number coordinates for the orbit in this chapter. The round-trip maps of Chapter 1 preserve addition, so the results describe the same constructed arithmetic. No new carrier is being substituted for it.

7.1 Identifications that respect addition

Definition 7.1 (An additive congruence). A relation $x \sim y$ on the natural numbers is an *additive congruence* if it is reflexive, symmetric and transitive, and satisfies

$$w \sim x \text{ and } y \sim z \implies w + y \sim x + z.$$

In particular, $x \sim y$ implies $x + t \sim y + t$ for every t .

The final implication follows by combining $x \sim y$ with $t \sim t$. Its direction matters. A congruence permits us to add the same count to both sides. It need not permit cancellation. A reader can lose distinctions as the count advances. Section 7.8 identifies the congruences that do permit cancellation: they are exactly the ones that survive the passage to signed records.

For an addition-preserving reading r , define

$$x \sim_r y \iff r(x) = r(y).$$

Equality of readings is an equivalence relation. Moreover, $r(w + y) = r(w) + r(y)$, so equal readings remain equal under addition. Thus the relation is an additive congruence. It is often called the *kernel* of the reading: it records exactly which inputs the reading identifies.

Definition 7.2 (The index–period relation). Let i be a natural number and let $p > 0$. Define

$$x \sim_{i,p} y \iff x = y \text{ or } (i \leq x, i \leq y, x \bmod p = y \bmod p). \quad (7.1)$$

The number i is the *index*; p is the *period*.

Before i , a count is identified only with itself. From i onward, counts with the same remainder modulo p are identified. For example, the definition gives the repeating chain

$$i \sim_{i,p} i + p \sim_{i,p} i + 2p \sim_{i,p} \dots$$

The displayed continuation describes repetitions of the same finite rule. It does not require a completed physical record of all those counts.

Proposition 7.3 (The family is well defined). *For each i and $p > 0$, relation (7.1) is an additive congruence. Furthermore,*

$$n \sim_{i,p} n + p \iff i \leq n. \quad (7.2)$$

Proof. Reflexivity uses the equality alternative. Symmetry follows by reversing equality or equality of remainders. For transitivity, an equality alternative lets us replace one endpoint directly. If both relations use the second alternative, all necessary endpoints lie at or beyond i , and equality of remainders is transitive.

For addition, if both input relations are equalities, their sums are equal. If at least one uses the second alternative, both corresponding sums are at least i . Equality of remainders of the inputs gives equality of remainders of their sums, since the remainder of a sum is determined by the remainders of its summands. This proves compatibility with addition.

Finally n and $n + p$ are distinct because $p > 0$, and they have the same remainder modulo p . Their relation therefore holds exactly when n is at least i . $\square$

7.2 How a period reaches an earlier collision

The difficult part of classification is to rule out irregular behaviour. We cannot simply cancel a large common addition in a congruence. Instead, we use an existing repetition to bring two chains together.

Say that a positive number d is a *period at x* when $x \sim x + d$. Addition compatibility and transitivity immediately give

$$x \sim x + kd \quad (k \in \mathbb{N}).$$

The same period also holds at every later point: add $n - x$ to the relation when $n \geq x$.

Lemma 7.4 (Moving a period back to a collision). *Suppose $d > 0$, $p > 0$, $x \sim x + d$, and $y \sim y + p$ with $y \geq x$. Then $x \sim x + p$.*

Proof. Choose a natural number k with $M = x + kd \geq y$. One explicit choice is $k = \lfloor (y-x)/d \rfloor + 1$. Repeating the first relation gives $x \sim M$. Translating the second relation from y to M gives $M \sim M + p$. Translating $x \sim M$ by p gives $x + p \sim M + p$. Symmetry and transitivity now combine the three relations:

$$x \sim M \sim M + p \sim x + p.$$

No cancellation of a congruence was used. □

Lemma 7.5 (A least period controls every collision). *Suppose $p > 0$, $a \sim a + p$, and p is the least positive period at a . Every positive period at any point is a multiple of p . Also, if $x < y$ and $x \sim y$, then $x \sim x + p$.*

Proof. Let $e > 0$ be a period at some x . Translate that period to $a + x$, which lies at or beyond a . Apply Lemma 7.4, using the existing period p at a . The result is $a \sim a + e$.

Divide e by p : write $e = qp + r$ with $0 \leq r < p$. Repeating the p -period gives $a \sim a + qp$. After adding r , this becomes $a + r \sim a + e$. Together with $a \sim a + e$, it gives $a \sim a + r$. If $r > 0$, this contradicts minimality of p . Therefore $r = 0$, so p divides e .

For the final assertion, $y - x > 0$ is a period at x . Translate the p -period from a to $a + x$, and apply Lemma 7.4 with the existing positive period $y - x$ at x . It yields $x \sim x + p$. □

These two lemmas explain why an apparently local collision determines the shape of every later one. Their hypotheses contain the periods they use; neither proof searches for a period without evidence that one exists.

7.3 The constructive classification

Theorem 7.6 (The classification with an explicit collision). *Let $\sim$ be an additive congruence. Suppose we have a decision procedure for each assertion $x \sim y$, and are given $a \sim b$ with $a \neq b$. Then there are unique natural numbers i and $p > 0$ such that $\sim$ is exactly $\sim_{i,p}$.*

The information in the hypothesis is concrete. The procedure decides individual pairs, and the supplied pair shows that at least one identification occurs. We will use two finite searches whose endpoints are supplied by that pair.

Construction and classification. Reverse a and b if necessary so that $a < b$. Search the integers $1, \dots, b - a$ for the least p such that $a \sim a + p$. Every test is decidable, and the last candidate succeeds because $a + (b - a) = b$. Thus p exists, is positive and is the least positive period at a .

Next search $0, \dots, a$ for the least i such that $i \sim i + p$. Again every test is decidable, and a succeeds. This i is also the least index among all natural numbers with that property: any smaller index would have occurred in the same search.

We prove equality of the two relations. Suppose first that $x \sim y$. If $x = y$, the equality alternative of (7.1) applies. If they differ, arrange $x < y$ by symmetry. Lemma 7.5 gives $x \sim x + p$, so minimality of i gives $i \leq x \leq y$. The positive gap $y - x$ is a period at x , hence a

multiple of p . Consequently x and y have the same remainder modulo p . This is the second alternative of (7.1).

Conversely, equality is always a congruence relation. For the second alternative, arrange $x \leq y$. Equal remainders mean $y - x = kp$ for some natural number k . The relation $i \sim i + p$ translates to a p -period at x because $i \leq x$. Repeating it k times gives $x \sim x + kp = y$. Symmetry handles the opposite ordering. Thus the relations agree at every pair. $\square$

Uniqueness. Suppose the same relation has two descriptions, with parameters (i, p) and (i', p') . The relation $i \sim i + p$, read in the second description, implies that p' divides p : its two unequal endpoints have equal remainders modulo p' . Reversing the descriptions shows that p divides p' . Positivity gives $p = p'$.

With the period fixed, equation (7.2) identifies the index. Since $i' \sim i' + p$, the first description gives $i \leq i'$. The second description similarly gives $i' \leq i$. Hence $i = i'$. $\square$

The classification proof separates the search from the structural argument. Once a least positive period at a and the least index for that period are supplied, Lemma 7.5 and the two relation inclusions establish the answer. The decision procedure is used to find those values from the explicit collision.

7.4 What changes when the collision is not supplied

Now replace the displayed collision by a negative statement:

$$\neg[\forall x, y (x \sim y \implies x = y)]. \quad (7.3)$$

This says that the congruence is not equality. Constructively, it does not yet hand us a pair to start the two searches.

From here on, *decidable* means that a procedure giving a positive or negative decision is supplied. We use Markov's principle in the corresponding form: for a predicate with such pointwise decisions, $\neg\neg\exists n P(n)$ implies $\exists n P(n)$. This makes explicit the decision data behind the instance decisions discussed in Chapter 6.

This is the principle of Definition 6.1, applied to an arbitrary decidable predicate. It is not the arithmetic MP rule of Chapter 6. That rule applies only to quantifier-free formulas of the arithmetic language, and Theorem 6.9 shows that each of its instances is settled by a bounded search and, with induction on every formula permitted, is derivable from the base rules. A decidable predicate here is any predicate that comes with decisions, and at this level Markov's principle is genuinely needed. The reversal below shows that the classification implies it, and Section 7.10 gives a model of intuitionistic second-order arithmetic in which Markov's principle fails and the classification of the next theorem fails with it. That system therefore proves neither.

Theorem 7.7 (The classification from negative nontriviality). *The following assertion is equivalent to Markov's principle, over the constructive mathematics retained here:*

For every decidable additive congruence satisfying (7.3), there exist i and $p > 0$ for which the congruence is $\sim_{i,p}$.

Markov's principle suffices. Let $B(k)$ assert that there are distinct related $x, y < k$. This is a finite search, so the given pairwise decisions decide $B(k)$. There cannot be no such k . Indeed, if none existed, any related pair x, y would have to be equal: if it were unequal, it would appear below $k = x + y + 1$. This would contradict (7.3). We have therefore established $\neg\neg\exists k B(k)$.

Markov's principle gives a successful k , and hence an explicit distinct related pair. Theorem 7.6 applies. $\square$

The converse is more informative. It shows how an arbitrary search question can be stored inside a congruence. Any procedure or principle that grants the stated classification must then be able to answer that question in the required sense.

The classification implies Markov's principle. Let $\theta(n)$ be any decidable predicate, and assume $\neg\neg\exists n \theta(n)$. Define

$$x \sim_{\theta} y \iff x = y \text{ or } \exists k \leq \min(x, y) \theta(k). \quad (7.4)$$

This relation is decidable by bounded search. It is reflexive and symmetric. For transitivity, equality alternatives are immediate. If $x \sim_{\theta} y$ and $y \sim_{\theta} z$ are both witnessed by successful indices, compare x and z . If $x \leq z$, the first witness lies below both x and z ; if $z < x$, the second does. Thus $x \sim_{\theta} z$. For addition compatibility, a successful witness below a related pair remains below the corresponding pair of sums. If both relations are equalities, the sums are equal. So this is an additive congruence.

It cannot be equality. If it were, any successful n would give $n \sim_{\theta} n+1$ and hence the impossible equation $n = n+1$. That would refute $\exists n \theta(n)$, contradicting its double negation.

Apply the asserted classification to obtain i, p with $p > 0$. The period relation gives $i \sim_{\theta} i+p$. Its equality alternative is impossible. Its other alternative supplies a k with $\theta(k)$. We have obtained $\exists k \theta(k)$ from its double negation, for an arbitrary decidable predicate. That is Markov's principle. $\square$

The theorem concerns the assertion for *every* congruence of the specified kind. It does not say that each particular classification needs MP. When an explicit collision is already available, the constructive proof applies directly.

7.5 What a decision procedure alone can buy

Keep the pairwise decision procedure, but supply neither a collision nor the assertion that the congruence is not equality. What may we then ask of the classification? Three questions are natural. Is the congruence equality? Which of the two alternatives holds: equality, or an index-period description? And, more modestly, if there is a description, is its index odd or even?

Each question has its own exact price. The prices are principles about binary sequences, and no two of them are equivalent over intuitionistic second-order arithmetic (Section 7.10). We state them here; Chapter 12 meets them again on the continuum.

Definition 7.8 (Omniscience principles for binary sequences). Let $\alpha : \mathbb{N} \rightarrow \{0, 1\}$ be a binary sequence; its value at each specified index is decidable. The *weak limited principle of omniscience*, WLPO, states that every binary sequence satisfies

$$(\forall n, \alpha(n) = 0) \quad \text{or} \quad \neg(\forall n, \alpha(n) = 0).$$

The *limited principle of omniscience*, LPO, states that every binary sequence satisfies

$$(\forall n, \alpha(n) = 0) \quad \text{or} \quad (\exists n, \alpha(n) = 1).$$

The *lesser limited principle of omniscience*, LLPO, concerns sequences with at most one value equal to 1. For each such sequence it states

$$(\forall k, \alpha(2k) = 0) \quad \text{or} \quad (\forall k, \alpha(2k + 1) = 0).$$

The binary form of Markov's principle states, for every binary sequence,

$$\text{MP}_{\text{bin}} : \quad \neg(\forall n, \alpha(n) = 0) \implies \exists n, \alpha(n) = 1.$$

A decision procedure for a predicate on the natural numbers is the same information as the binary sequence of its answers. So the binary LPO is the predicate form of Definition 6.1 for predicates that come with decisions, and MP_{bin} is equivalent to the form of Markov's principle used in Theorem 7.7, since $\neg(\forall n, \alpha(n) = 0)$ and $\neg\neg\exists n, \alpha(n) = 1$ imply each other. WLPO decides whether the all-zero assertion holds, without producing an index. LLPO, given at most one event, names a parity class free of events; it need not say whether an event occurs in the other class.

For a decidable congruence $\sim$, all three questions can be put to one sequence. Let

$$\beta_{\sim}(k) = 1 \iff \text{there are } x, y < k \text{ with } x \neq y \text{ and } x \sim y. \quad (7.5)$$

Each value is a finite search, decided by the supplied procedure. The sequence is all zero exactly when $\sim$ is equality. A related pair $x \neq y$ makes $\beta_{\sim}(x + y + 1) = 1$. Conversely, if the sequence is all zero and $x \sim y$, decide whether $x = y$; the answer $x \neq y$ would make $\beta_{\sim}(x + y + 1) = 1$, so $x = y$. Only the decidability of equality between two counts is used.

In the other direction we reuse the congruence $\sim_{\theta}$ of (7.4), whose construction did not use the double-negation hypothesis. Given a binary sequence α , take $\theta(n)$ to be $\alpha(n) = 1$. Then $\sim_{\theta}$ is a decidable additive congruence, and

$$\sim_{\theta} \text{ is equality} \iff \forall n, \alpha(n) = 0. \quad (7.6)$$

An event at n relates n and $n + 1$. Without events, only the equality alternative of (7.4) remains.

Theorem 7.9 (Three questions with a decision procedure). *Over the retained constructive base:*

1. *The assertion that every decidable additive congruence is equality or is not equality is equivalent to WLPO.*
2. *The assertion that every decidable additive congruence is equality or equals some $\sim_{i,p}$ with $p > 0$ is equivalent to LPO.*

3. *The assertion that, for every decidable additive congruence, either every index–period description of it has odd index or every such description has even index, is equivalent to LLPO.*

Moreover, with no added principle, the assertion in (2) is equivalent to the conjunction of the assertion in (1) with the assertion of Theorem 7.7.

Equality has no index–period description, so both alternatives in (3) hold for it vacuously. The uniqueness argument of Theorem 7.6 uses neither the decision procedure nor the collision, so a congruence has at most one description, and (3) asks for one parity bit and nothing more.

The equality test is WLPO. Assume WLPO and apply it to $\beta_{\sim}$. If the sequence is all zero, $\sim$ is equality. If it is not all zero, $\sim$ cannot be equality, since equality would make it all zero. Conversely, given a binary sequence α , apply the assertion to $\sim_{\theta}$. By (7.6) its two alternatives are exactly the two alternatives of WLPO for α . $\square$

The full alternative is LPO. Assume LPO and apply it to $\beta_{\sim}$. If the sequence is all zero, $\sim$ is equality. Otherwise LPO supplies k with $\beta_{\sim}(k) = 1$. The bounded search at k now produces an explicit pair $x \neq y$ with $x \sim y$, and Theorem 7.6 gives the description.

Conversely, apply the assertion to $\sim_{\theta}$. The equality alternative gives $\forall n, \alpha(n) = 0$ by (7.6). In the other alternative, $i \sim_{\theta} i + p$ with $p > 0$. These counts are unequal, so the definition of $\sim_{\theta}$ supplies $k \leq i$ with $\alpha(k) = 1$. Either way we have the conclusion of LPO.

For the final assertion, the equality test in (1) leaves two cases. The positive case is the first alternative of (2). In the negative case the congruence satisfies (7.3), and the assertion of Theorem 7.7 gives a description. Conversely, (2) gives (1) at once, because an index–period congruence is not equality. It also gives the assertion of Theorem 7.7, because for a congruence that is not equality its equality alternative is excluded. $\square$

The last part is the catalogue’s copy of Proposition 12.9: deciding the alternative is deciding whether there is anything to find, followed by finding it.

The parity of the index is LLPO. The forward direction needs a sequence with at most one event that records the parity. Call a pair $x < y$ the *first collision* of $\sim$ when $x \sim y$, no related pair $x' < y'$ has $y' < y$, and no $x' < x$ satisfies $x' \sim y$. Two first collisions must agree: each would otherwise violate a leastness condition of the other. Whether a given pair is the first collision is a finite check. For $b \in \{0, 1\}$ define

$$\gamma(2y + b) = 1 \iff \text{some } x < y \text{ with } x \bmod 2 = b \text{ forms the first collision } (x, y).$$

Each value is decided by finite search, and uniqueness of the first collision gives at most one event.

Suppose $\sim$ equals $\sim_{i,p}$. A related pair $x < y$ satisfies $i \leq x$ and $x + p \leq y$, because the gap $y - x$ is then a positive multiple of p . Hence the first collision is $(i, i + p)$, and γ has its event at $2(i + p) + (i \bmod 2)$. Now apply LLPO to γ . If every even entry is zero, no description has even index, so every description has odd index. If every odd entry is zero, every description has even index.

Conversely, let α have at most one event. If $\alpha(k) = 1$, the congruence $\sim_\theta$ identifies two counts exactly when they are equal or both at least k ; that is $\sim_{k,1}$. Apply the assertion to $\sim_\theta$. In its first alternative, an event at an even place $2j$ would give the description $\sim_{2j,1}$ with even index; so $\alpha(2j) \neq 1$, and since α is binary, $\alpha(2j) = 0$ for every j . The second alternative gives $\alpha(2j+1) = 0$ for every j in the same way. These are the two alternatives of LLPO. $\square$

The larger element of the first collision is the half y of the position $2y + b$, and the smaller element is then the least count related to y . That is why one parity bit is all the sequence needs to carry.

7.6 The unrestricted catalogue and excluded middle

Finally suppose no pairwise decision procedure is supplied. Ask for a catalogue that always returns one of two mathematical alternatives: either the congruence is equality, or it has an index–period description.

Theorem 7.10 (The full catalogue has exactly the strength of EM). *The assertion*

every additive congruence is equality or some $\sim_{i,p}$ with $p > 0$

is equivalent to excluded middle over the retained constructive base.

Excluded middle suffices. Use EM to decide whether the congruence is equality. If so, take that alternative. Otherwise EM gives a distinct related pair: denying the existence of any such pair would make every related pair equal. Reverse the pair if needed to obtain $a < b$.

We need least numbers for predicates that may not have supplied decision procedures. EM supplies the following least-number argument. If $P(n)$ is known, use strong induction on n . EM either gives a $k < n$ with $P(k)$, in which case the induction hypothesis gives a least witness, or says no earlier witness exists, in which case n itself is least. Thus every inhabited natural-number predicate has a least witness under EM.

Apply this argument first to positive periods at a , using $b - a$ as a witness, and then to indices carrying that least period, using a as a witness. The structural argument in Theorem 7.6 now identifies the congruence as $\sim_{i,p}$. This proves the catalogue. $\square$

The catalogue implies excluded middle. Let P be any proposition. Define

$$x \sim_P y \iff x = y \text{ or } P. \tag{7.7}$$

This is an additive congruence without first deciding P . Reflexivity uses equality; symmetry and transitivity preserve an available proof of P or reduce to equality. For addition, if both input relations are equalities, their sums are equal; otherwise a proof of P already supplies the conclusion.

Apply the catalogue. In its equality branch, P would imply $0 \sim_P 1$, and hence $0 = 1$. Therefore $\neg P$. In its index–period branch, $i \sim_P i + p$ with $p > 0$. These numbers are unequal, so definition (7.7) gives P . Either branch supplies a side of $P \vee \neg P$. Since P was arbitrary, excluded middle follows. $\square$

This proof does not turn EM into an executable decision procedure for every proposition. It establishes an equivalence between mathematical assertions in the metatheory. The forward proof keeps its least-witness claims at that level. The version with supplied pairwise procedures and an explicit collision has the more concrete bounded searches. Between the two lies the version of Section 7.5: the same alternative, asked of a congruence that comes with a decision procedure but no collision, costs exactly LPO.

7.7 What the classification tells us about a reader

Return to an addition-preserving reading r from the orbit into an additive commutative monoid V . Its kernel, $x \sim_r y$ exactly when $r(x) = r(y)$, is an additive congruence, and r is injective exactly when its kernel is equality. A question about the reader is therefore a question about a congruence, and the prices follow it.

The only new ingredient is the information carried by the target. If equality of values in V can be decided, the kernel is a decidable congruence. Conversely, every additive congruence $\sim$ is the kernel of a reading: send x to its class in the quotient $N/\sim$, whose addition $[x] + [y] = [x + y]$ is well defined because $\sim$ respects addition. Two classes are equal exactly when their representatives are related, so equality of classes is decidable when $\sim$ is.

Theorem 7.11 (The recognition dichotomy and its prices). *Consider readings that preserve zero and addition from the orbit into additive commutative monoids.*

1. *If equality in the target of r is decidable and $r(a) = r(b)$ for given $a \neq b$, then the kernel of r is $\sim_{i,p}$ for unique i and $p > 0$. No added principle is used.*
2. *For readings into targets with decidable equality, each of the following assertions about every such reading is equivalent to the principle named after it:*
 - (a) *the reading is injective or not injective: WLPO;*
 - (b) *the reading is injective or its kernel is some $\sim_{i,p}$ with $p > 0$: LPO;*
 - (c) *if the reading is not injective, its kernel is some $\sim_{i,p}$ with $p > 0$: MP;*
 - (d) *every index–period description of the kernel has odd index, or every one has even index: LLPO.*
3. *The assertion that every such reading, into an arbitrary additive commutative monoid, is injective or has an index–period kernel is equivalent to EM.*

Proof. For (1), the kernel is a decidable congruence with the exhibited collision $a \sim_r b$; apply Theorem 7.6.

For (2) and (3), compare each assertion with the catalogue assertion of the same price. A reading supplies its kernel, which is decidable when equality in the target is, and injectivity of the reading corresponds to the kernel being equality. A congruence supplies its quotient reading $x \mapsto [x]$, whose kernel is the congruence itself, and whose target has decidable equality when the congruence is decidable. These two translations carry each reading assertion to the corresponding catalogue assertion and back, with no added principle. Theorems 7.7, 7.9 and 7.10 now give the prices. $\square$

The classical statement of the dichotomy, for arbitrary targets, therefore costs exactly excluded middle. A reader whose outputs can be compared pays only LPO for the same dichotomy, and nothing at all once a single collision has been observed.

The result concerns the reading's identifications. It does not assert that every target object is reached, that every physical device has this form, or that addition preservation can be omitted. Those are different hypotheses. Chapter 8 will make the relationship between a quotient and the range of a reading precise.

7.8 Signed records keep only the pure periods

A congruence permits adding the same count to both sides; it need not permit removing one. The signed records of Chapter 4 show exactly when it does. An *additive congruence* on $\mathbb{Z}_\delta$ is defined as in Definition 7.1, with signed records in place of counts. Reading a count n as the signed record $[n, 0]$, such a congruence $\approx$ restricts to the relation $x \sim y$ defined by $[x, 0] \approx [y, 0]$. Call an additive congruence on the natural numbers *cancellative* when $x + z \sim y + z$ implies $x \sim y$.

Theorem 7.12 (Congruences of the signed records). *Restriction is a bijection from the additive congruences of $\mathbb{Z}_\delta$ onto the cancellative additive congruences of the natural numbers. A cancellative congruence $\sim$ extends by*

$$[a, b] \approx [c, d] \iff a + d \sim c + b, \quad (7.8)$$

and this is its only extension. Among the index–period congruences, $\sim_{i,p}$ is cancellative exactly when $i = 0$.

Proof. A restricted relation is an additive congruence because $[x + y, 0] = [x, 0] + [y, 0]$. It is cancellative because $\mathbb{Z}_\delta$ has negatives: adding $[0, z]$ to both sides of $[x + z, 0] \approx [y + z, 0]$ gives $[x, 0] \approx [y, 0]$.

Now let $\sim$ be cancellative. Relation (7.8) does not depend on representatives. If (a, b) is replaced by a balanced pair (a', b') , so that $a + b' = a' + b$, add b' to both sides of $a + d \sim c + b$. The left side becomes $(a' + d) + b$, so $(a' + d) + b \sim (c + b') + b$, and cancellation removes b . The same argument handles the second pair and the reverse direction. Reflexivity and symmetry are immediate. For transitivity, add f to $a + d \sim c + b$ and b to $c + f \sim e + d$; the two results chain to $(a + f) + d \sim (e + b) + d$, and cancellation removes d . Adding two instances of (7.8) gives the instance for the sums, so the extension respects addition.

The extension restricts to $\sim$, since $[x, 0] \approx [y, 0]$ means $x + 0 \sim y + 0$. Conversely, let $\approx'$ be any congruence on $\mathbb{Z}_\delta$ restricting to $\sim$. Adding $[b + d, 0]$ to both sides turns $[a, b] \approx' [c, d]$ into $[a + d, 0] \approx' [c + b, 0]$, and adding $[0, b + d]$ reverses the step. So $\approx'$ is relation (7.8). This proves the bijection and the uniqueness of the extension.

Finally, if $i > 0$ then $0 + i \sim_{i,p} p + i$, since both sums are at least i and have the same remainder modulo p , while 0 and p are different and $0 < i$, so they are not related. If $i = 0$, equal remainders of $x + z$ and $y + z$ give equal remainders of x and y . $\square$

Passing to signed records therefore erases the index. A congruence with index $i > 0$, which keeps the counts below i apart, has no counterpart on the signed records; only the pure periods

survive. The prices do not change, provided the parity question is asked of the period: every description now has index zero, so the parity of the index costs nothing, while the parity of the period costs exactly LLPO.

Corollary 7.13 (The signed catalogue). *Let $\approx$ be a decidable additive congruence on $\mathbb{Z}_\delta$ with an exhibited pair of distinct related records. Then there is exactly one $p > 0$ such that*

$$[a, b] \approx [c, d] \iff (a + d) \bmod p = (c + b) \bmod p.$$

No added principle is used. For congruences on $\mathbb{Z}_\delta$, each of the following assertions is equivalent to the principle named after it:

- (a) *every decidable congruence that is only known not to be equality has this form: MP;*
- (b) *for every decidable congruence, either every period p describing it in this form is odd, or every such period is even: LLPO;*
- (c) *every decidable congruence is equality or is not equality: WLPO;*
- (d) *every decidable congruence is equality or has this form: LPO;*
- (e) *every congruence, with no decision procedure, is equality or has this form: EM.*

The parity of the index, asked as in Theorem 7.9, costs nothing here: every index–period description of the restriction of a congruence on $\mathbb{Z}_\delta$ has index 0.

Proof. Restriction carries a decision procedure for $\approx$ to one for $\sim$, and (7.8) carries one back. Distinct related records $[a, b] \approx [c, d]$ give the collision $a + d \sim c + b$, with $a + d \neq c + b$ because the records are distinct. Theorem 7.6 describes $\sim$ as $\sim_{i,p}$, cancellation forces $i = 0$, and (7.8) gives the displayed relation. The period is unique because the description is.

For assertions (a), (c), (d) and (e), the forward directions follow in the same way from Theorems 7.7, 7.9 and 7.10 applied to the restriction. By (7.8), $\approx$ is equality exactly when $\sim$ is: if $\sim$ is equality, $[a, b] \approx [c, d]$ says $a + d = c + b$, which is equality of the two records, and if $\approx$ is equality, so is its restriction.

The reversals cannot simply reuse $\sim_\theta$: when the first witness of θ is positive, $\sim_\theta$ has a positive index and is not cancellative. Read a signed record through the faithful integer reading I of Proposition 4.6, and for a decidable θ put

$$u \approx_\theta v \iff u = v, \text{ or } \theta \text{ has a least witness } n \text{ and } (n + 1)! \text{ divides } I(v) - I(u).$$

This relation is decidable: when $u \neq v$ the difference $I(v) - I(u)$ is nonzero, and a witness n with $(n + 1)!$ dividing it satisfies $n < (n + 1)! \leq |I(v) - I(u)|$, so the search is bounded. A least witness is unique, so any two uses of the second alternative involve the same factorial; this makes $\approx_\theta$ reflexive, symmetric and transitive, and it respects addition. If $\approx_\theta$ is equality, θ has no witness, since a witness gives a least one n by bounded search and then relates $[0, 0]$ to the distinct record $[(n + 1)!, 0]$. If $\approx_\theta$ has a period $p > 0$, it relates $[0, 0]$ to $[p, 0]$, which needs a witness. These two facts give MP from (a) and LPO from (d), exactly as in the proofs of Theorems 7.7 and 7.9. Without a witness only the alternative $u = v$ remains, so $\approx_\theta$ is equality

exactly when θ has no witness. For $\theta(n)$ given by $\alpha(n) = 1$, the two alternatives of (c) at $\approx_\theta$ are therefore the two alternatives of WLPO for α . For EM, take any proposition P and let $u \approx_P v$ hold when $u = v$, or when P holds and 2 divides $I(v) - I(u)$. Its equality alternative refutes P , because P would relate $[0, 0]$ and $[2, 0]$; a period $p > 0$ relates $[0, 0]$ and $[p, 0]$, which requires P .

The parity question needs its own sequence, because the index of a restriction carries no information. For (b), let $\gamma(2y + b) = 1$ exactly when y is the least positive count with $[0, 0] \approx [y, 0]$ and b is the remainder of y modulo 2. Each value is a bounded search, and the least such count is unique, so γ has at most one event. If $\approx$ is described by the period p , the least such count is p , and the event sits at $2p + (p \bmod 2)$, which has the parity of p . Apply LLPO to γ . If every even entry is zero, no describing period is even; if every odd entry is zero, none is odd.

Conversely, let α have at most one event. Put $q(n) = 2n + 1$ when n is even and $q(n) = 2n + 2$ when n is odd, so that $q(n) > n$ and $q(n)$ has the parity of $n + 1$. Let $u \approx_\alpha v$ hold when $u = v$, or when $\alpha(n) = 1$ for some n such that $q(n)$ divides $I(v) - I(u)$. A witness n for a nonzero difference d satisfies $n < q(n) \leq |d|$, so each pair is decided by bounded search. Since there is at most one event, any two uses of the second alternative involve the same n ; hence $\approx_\alpha$ is an additive congruence. It is equality when α has no event, and it is described by the period $q(n)$ when $\alpha(n) = 1$. An event at an even place gives an odd period, and an event at an odd place an even one. So the first alternative of (b), every period odd, gives $\alpha(2j + 1) = 0$ for every j , and the second gives $\alpha(2j) = 0$ for every j . These are the two alternatives of LLPO.

Finally, the restriction of a congruence on $\mathbb{Z}_\delta$ is cancellative, so by Theorem 7.12 every index–period description of it has index 0. The alternative “every description has even index” therefore holds for every congruence, with no principle at all. $\square$

7.9 One classification, six prices

We have also learned how to read a logical price correctly. One classification, of one kind of object, has six exact prices. The price is fixed by what is supplied and by what is asked for:

Supplied	Asked for	Exact price
decisions and a collision	the description	none
decisions, and “not equality”	the description	MP
decisions	the parity of the index	LLPO
decisions	equality or not	WLPO
decisions	equality or the description	LPO
nothing	equality or the description	EM

Theorems 7.6, 7.7, 7.9 and 7.10 prove the rows. Theorem 7.11 gives the same table for readers, and Corollary 7.13 all six rows for signed records, the parity row read on the period. The reversals, rather than the presence of a label on one proof, establish these exact equivalences. Each reversal stores an arbitrary sequence or proposition inside a congruence, so any principle that grants the classification must answer that question.

The equivalences place each form exactly at its principle, and the principles themselves are different. No two of the six are equivalent, so no version of the classification is equivalent to two of them. Each separation is a model of intuitionistic second-order arithmetic in which one principle holds and the other fails. Three take their truth values in the open sets of a space: the one-point compactification of an uncountable discrete set (Markov's principle holds, LLPO fails), the remainder $\beta\mathbb{N} \setminus \mathbb{N}$ (LLPO holds, WLPO fails), and $\beta\mathbb{N}$ itself (WLPO holds, MP and LPO fail). The fourth takes its truth values in a chain (LPO holds, EM fails). In each of these models every version of the classification has exactly the truth value of its principle. The next section states the system and builds the models.

7.10 Why the six prices are different

An equivalence is proved by two constructions. A difference between two principles needs something of another kind: a precise system of reasoning, and an interpretation of that system in which every derivable sentence is true while one principle holds and the other fails. We supply both. Interpretations of this kind are standard in constructive mathematics [4, 5]; the work particular to this chapter is the check that the six versions of the classification behave in them exactly as their principles do.

Definition 7.14 (Intuitionistic second-order arithmetic). The language has two kinds of variable: $x, y, \dots$ for numbers and $X, Y, \dots$ for sets of numbers. Terms are built from number variables with $0, S, +$ and $\cdot$; the atomic formulas are equations $t = u$ and memberships $t \in X$; formulas are built from these with $\perp, \wedge, \vee, \rightarrow$ and quantifiers over both kinds of variable. A derivation uses the intuitionistic rules for the connectives and for both kinds of quantifier, reflexivity and substitution of equals, the six arithmetic rules of Chapter 5 as axioms, the induction axiom

$$\forall X(0 \in X \rightarrow \forall x(x \in X \rightarrow S(x) \in X) \rightarrow \forall x(x \in X)),$$

and the comprehension axiom $\exists X \forall x(x \in X \leftrightarrow \varphi)$ for every formula φ in which X is not free. This system is usually called HAS [4, 5].

Comprehension and the induction axiom together give induction for every formula. The objects of this chapter become sets. A binary sequence is a set X with $\forall n(n \in X \vee \neg n \in X)$, read as $\alpha(n) = 1$ exactly when $n \in X$. A relation on counts is a set of codes of pairs, using the injective code $(x+y)(x+y)+x$ for the pair (x, y) . A relation is decidable when $x \sim y \vee \neg(x \sim y)$ for every pair, and a description by i and p says that the relation agrees with (7.1), with the order and the remainders written by existential quantifiers. The five priced principles and the six versions of the classification are then sentences of this language. Excluded middle is $\forall X(0 \in X \vee \neg 0 \in X)$, which gives every instance $P \vee \neg P$ through comprehension.

Definition 7.15 (Models of truth values). A *system of truth values* is a set H ordered by strength, with a meet and a join for every family of elements and an implication $\Rightarrow$ satisfying

$$a \wedge c \leq b \iff c \leq (a \Rightarrow b).$$

Its least and greatest elements are $\perp$ and $\top$, and $\neg a$ is $a \Rightarrow \perp$. A formula of Definition 7.14 receives a value in H under an assignment to its free variables. Numbers range over $\mathbb{N}$, and an

equation has the value $\top$ when it holds there and $\perp$ otherwise. Sets range over all assignments $A : \mathbb{N} \rightarrow H$, and $t \in X$ has the value $A(t)$. The connectives $\wedge$, $\vee$ and $\rightarrow$ are meet, join and $\Rightarrow$; a universal quantifier is the meet of the values of its instances, and an existential quantifier their join.

The open sets of a topological space form such a system. Meet and join of two open sets are intersection and union, the join of a family is its union, the meet of a family is the interior of its intersection, and $U \Rightarrow V$ is the interior of the union of V with the complement of U . So $\neg U$ is the largest open set disjoint from U . A chain is another example. For $0 < 1 < 2 < \dots < \infty$, the implication $a \Rightarrow b$ is $\top$ when $a \leq b$ and b otherwise.

Lemma 7.16 (Derivable sentences are true in every model). *If a sentence of Definition 7.14 has a derivation, its value in every system of truth values is $\top$.*

Outline. Prove more generally, by induction on derivations, that the meet of the values of the hypotheses lies below the value of the conclusion under every assignment. The rules for $\wedge$, $\vee$ and $\rightarrow$ hold because meet, join and $\Rightarrow$ have exactly the corresponding properties, and the quantifier rules hold because meets and joins of instances do. The arithmetic axioms and the equality rules are facts about $\mathbb{N}$, whose equations have the values $\top$ and $\perp$. Comprehension holds because the assignment sending n to the value of φ at n is one of the sets over which set quantifiers range. Induction holds because, for every A , an induction on m shows $A(0) \wedge \bigwedge_n (A(n) \Rightarrow A(n+1)) \leq A(m)$. $\square$

A principle whose value is not $\top$ in some model therefore has no derivation. If one sentence has the value $\top$ and another does not, the implication from the first to the second has the value of the second and has no derivation either. One model in which P holds and Q fails shows that P and Q are not equivalent.

In the open sets of a space, a binary sequence is a sequence of sets $A_0, A_1, \dots$ that are both open and closed, since $A_n \vee \neg A_n = \top$ says exactly that the complement of A_n is open. At a point x the sequence reads $\alpha(n) = 1$ when $x \in A_n$, so it varies continuously with the point. Write O for the union of the sets A_n . The value of $\exists n \alpha(n) = 1$ is O , and the value of $\forall n \alpha(n) = 0$ is $\neg O$. At this sequence the principles say:

- LPO: O is closed;
- WLPO: the closure of O is open;
- MP: O is the interior of its closure; (7.9)
- LLPO: if the A_n are disjoint, the union of the even-indexed ones and the union of the odd-indexed ones have disjoint closures.

Excluded middle says that every open set is closed. To refute a principle it is enough to find one sequence at which its condition fails. To verify one, the condition must hold for every sequence, and also, inside each open part of the space, for the sequences decided only on that part. Each of the three spaces below has a base of sets that are both open and closed, and this reduces the second task to the first. If W is such a set inside the part, the sets $A_n \cap W$ are open and closed in the whole space, so they form a sequence decided everywhere, and a disjoint one when the A_n are disjoint on the part. Intersecting with the open set W preserves meets, joins and implication, so the condition for that sequence, which the argument for the whole

space supplies, gives the condition for the original sequence on W , and these sets W cover the part. The chain is treated directly.

The models are built in ordinary classical mathematics, with the axiom of choice supplying the ultrafilters. That is the usual setting of an independence proof. The question is what a specified system derives, and the answer is established about that system from outside it.

The one-point compactification of an uncountable set

Let S be an uncountable set, for example the set of all sets of natural numbers, and add one point ∞ . The open sets are all subsets of S , together with the sets that contain ∞ and omit only finitely many points of S . The sets both open and closed are the finite subsets of S and their complements.

Markov's principle holds. Let O be a countable union of sets both open and closed. If one of them contains ∞ , the complement of O is a finite subset of S , which is open, so O is closed and equals the interior of its closure. Otherwise O is a countable subset of S . Each point of S is isolated, so the closure of O adds at most ∞ , and ∞ is not interior to it: every neighbourhood of ∞ contains uncountably many points of S , hence points outside O and outside its closure. Again O is the interior of its closure. LLPO fails. Choose distinct points $s_0, s_1, \dots$ of S and put $A_n = \{s_n\}$. The even-indexed and the odd-indexed points form two infinite subsets of S , and ∞ lies in the closure of both.

The remainder $\beta\mathbb{N} \setminus \mathbb{N}$

The points are the non-principal ultrafilters on $\mathbb{N}$. For $B \subseteq \mathbb{N}$ let B^* be the set of those that contain B . These sets are open and closed and form a base, and by compactness every set both open and closed has this form. Moreover $B^* \cap C^* = (B \cap C)^*$, and B^* is empty exactly when B is finite.

LLPO holds. Let the disjoint sets be $A_n = B_n^*$, so that $B_m \cap B_n$ is finite when $m \neq n$. Let B be the union, over k , of B_{2k} with its finitely many points in $B_1 \cup B_3 \cup \dots \cup B_{2k+1}$ removed. Then only finitely many points of each B_{2k} lie outside B , and each B_{2m+1} meets B only in its finite overlaps with $B_0, B_2, \dots, B_{2m-2}$. Hence every even-indexed A_n lies in B^* and every odd-indexed one in $(\mathbb{N} \setminus B)^*$. These two sets are closed and disjoint, so they separate the closures of the two unions.

WLPO fails. Split $\mathbb{N}$ into infinitely many infinite sets $B_0, B_1, \dots$, put $A_n = B_n^*$, and let O be their union. If the closure of O were open, it would be open and closed, hence equal to B^* for some B , and each B_n would lie in B apart from finitely many points. Choose one point b_n of $B \cap B_n$ for each n . The set $C = \{b_0, b_1, \dots\}$ is infinite, lies in B , and meets each B_n in a single point. So C^* is a nonempty open subset of the closure of O that meets no A_n , and so misses O . That is impossible.

The space $\beta\mathbb{N}$

Now take all ultrafilters on $\mathbb{N}$. The principal ultrafilter at n is an isolated point, which we write n . The sets $\bar{B}$ of ultrafilters containing B are open and closed, they form a base, and $\bar{B}$ is the closure of B . An open set U meets $\mathbb{N}$ in a set B that is dense in U , so the closure of U is $\bar{B}$, which is open. WLPO holds. For Markov's principle and LPO, take $A_n = \{n\}$. Then $O = \mathbb{N}$, whose closure is the whole space. So O is neither closed nor the interior of its closure, and both principles fail.

The chain $0 < 1 < 2 < \dots < \infty$

In a chain, $\neg a = \perp$ for every $a \neq \perp$, so $a \vee \neg a = \top$ only when a is $\perp$ or $\top$. A sequence decided at the value $\top$ is therefore an ordinary binary sequence, and the classical mathematics in which the model is built either finds a 1 in it or shows that it is all zero. A sequence decided only below a smaller value d has, at each n , either the value $\perp$ or a value at least d , and the same dichotomy applies below d . LPO holds. Excluded middle fails at the value 1, since $1 \vee \neg 1 = 1$.

Model	Holds	Fails
one-point compactification	MP	LLPO, WLPO, LPO, EM
$\beta\mathbb{N} \setminus \mathbb{N}$	LLPO	WLPO, LPO, EM
$\beta\mathbb{N}$	WLPO, LLPO	MP, LPO, EM
chain $0 < 1 < \dots < \infty$	LPO, MP, WLPO, LLPO	EM

Table 7.1. Four models of intuitionistic second-order arithmetic. In the first three the truth values are the open sets of the space named. Each entry not checked in the text follows because in every model EM implies LPO, LPO implies MP and WLPO, and WLPO implies LLPO.

Theorem 7.17 (The six principles are different). *Over intuitionistic second-order arithmetic:*

1. *none of MP, LLPO, WLPO, LPO and EM is derivable, and no two of the six principles none, MP, LLPO, WLPO, LPO and EM are equivalent;*
2. *in every system of truth values, each of the six versions of the classification in the table of Section 7.9 has the same value as its principle;*
3. *consequently no version of the classification is equivalent to any principle other than its own.*

Outline. For (1), Table 7.1 supplies a model for each of the fifteen pairs. The empty principle holds everywhere, and the one-point compactification refutes LLPO, WLPO, LPO and EM, while $\beta\mathbb{N}$ refutes MP. The one-point compactification separates MP from LLPO, WLPO, LPO and EM; the remainder separates LLPO from WLPO, LPO and EM; $\beta\mathbb{N}$ separates WLPO from LPO and EM; and the chain separates LPO from EM. Lemma 7.16 turns each model into the stated failure of derivability. The implications in the caption hold in every model, by short arguments. Excluded middle decides whether some $\alpha(n) = 1$, and if none is, every $\alpha(n) = 0$. LPO's alternative $\exists n \alpha(n) = 1$ refutes $\forall n \alpha(n) = 0$. WLPO applied to the even entries decides LLPO, because an event at an odd place would make every even entry zero.

For (2), the arguments of this chapter run inside a model. A relation is now an assignment of a truth value to each pair. Below any value h at which it is an additive congruence, the pairs whose value is at least h form an ordinary additive congruence, so the structural lemmas apply there, and the first collision gives the description. The forward directions apply the principle to the collision search $\beta_\sim$, and, for LLPO, to the first-collision sequence γ . The reversals use the congruences $\sim_\theta$ and $\sim_P$, defined by the same formulas. Each version and its principle therefore have the same value in every model.

For (3), let a version have price P and let Q be another principle. The model that separates P from Q gives the version the value of P , so the version and Q have different values there, and by Lemma 7.16 they are not equivalent. $\square$

Two consequences follow. Markov's principle is genuinely needed for the classification from negative nontriviality: the system proves neither the principle nor that version. And the six prices do not form a single ladder. EM, LPO, WLPO and LLPO decrease strictly in strength, and LPO is WLPO and MP together, by the last assertion of Theorem 7.9; but Markov's principle is comparable with neither WLPO nor LLPO. The one-point compactification satisfies MP and refutes LLPO, and $\beta\mathbb{N}$ satisfies WLPO and refutes MP.

Questions to work through

1. In Lemma 7.4, identify each use of addition compatibility and each use of transitivity. Why would cancelling a common addition have been unjustified?
2. In the constructive classification, which supplied value bounds the search for the period? Which bounds the search for the index? At what point is the pairwise decision procedure used?
3. In (7.4), explain why an individual pair can be decided by finite search even though $\exists n \theta(n)$ may not yet have a witness.
4. Trace both branches of the EM reversal. Which branch supplies P , and which supplies $\neg P$? Why was no initial decision of P needed to define the congruence?
5. In the WLPO part of Theorem 7.9, show that $\sim_\theta$ is equality exactly when θ has no witness. In the forward direction, where is the decidability of equality between two counts used, and why does that use need no added principle?
6. Show that a decidable congruence has at most one first collision, and that the event of γ for $\sim_{i,p}$ sits at $2(i + p) + (i \bmod 2)$. Why does the sequence need to record only the parity of the smaller element?
7. Show directly that $\sim_{2,3}$ is not cancellative. Describe the congruence on $\mathbb{Z}_\delta$ whose restriction is $\sim_{0,3}$, using the integer reading of its records.
8. In the reversal for the period parity in Corollary 7.13, let α have its only event at $n = 3$. Which records does $\approx_\alpha$ relate to $[0, 0]$, and what is its period? Which alternative of LLPO does the corollary's assertion return? Why would the index parity of Theorem 7.9 be useless here?

9. In the open sets of $\beta\mathbb{N}$, take $A_n = \{n\}$. Compute the values of $\exists n \alpha(n) = 1$, of $\forall n \alpha(n) = 0$ and of its negation. Read off from (7.9) why Markov's principle and LPO fail at this sequence while WLPO holds.
10. Show that in a chain every sequence decided at the value $\top$ has only the values $\perp$ and $\top$. Why does this make LPO hold, and which value refutes excluded middle in the chain $0 < 1 < 2 < \dots < \infty$?

Chapter 8

Changing the representation

A number can be written as a numeral or as a list of identical marks. Changing between these descriptions should not change the answer to an arithmetic question. But that expectation needs a proof. The translations must preserve the operations, and quantified statements must range over corresponding objects in both descriptions.

We will construct the translations and prove what they preserve. There are three distinct objects to keep in view: the arithmetic carrier, the interpretation of a formula, and the finite derivation of that formula. A theorem about one does not automatically settle a question about the others. We therefore take them in turn. We change the carrier and follow every formula through the change, measure what survives when a translation runs in only one direction, and compare the derivations themselves with those of an independently stated proof system, Peano arithmetic, and with its intuitionistic counterpart, Heyting arithmetic.

8.1 An interpretation is fixed by its operations

Chapter 2 established a unique reading of the start-and-step orbit into any pointed step system. The terms of Chapter 5 have a larger signature: variables, zero, successor, addition and multiplication. The same structural method applies to this explicitly stated signature.

Definition 8.1 (A realization of the term signature). A term realization consists of a carrier A , an assigned element v_n for every variable position, an element 0_A , and operations

$$S_A : A \longrightarrow A, \quad +_A, \cdot_A : A \times A \longrightarrow A.$$

An interpretation preserving the signature sends each variable x_n to v_n , sends 0 to 0_A , and commutes with the three operations. No arithmetic law is assumed merely by giving these operations.

Theorem 8.2 (Unique interpretation of terms). *For every term realization, exactly one interpretation preserves the specified signature. It is the structural evaluation*

$$\begin{aligned} F_A(x_n) &= v_n, & F_A(0) &= 0_A, \\ F_A(S(t)) &= S_A(F_A(t)), \\ F_A(t + u) &= F_A(t) +_A F_A(u), \\ F_A(t \cdot u) &= F_A(t) \cdot_A F_A(u). \end{aligned}$$

Proof. The displayed recursion gives an interpretation because every term is finite and each recursive call concerns an immediate constituent term. It preserves the signature by its defining clauses.

For uniqueness, let G be any signature-preserving interpretation and induct on the term. At a variable and at zero, preservation specifies the answer. At a successor, preservation gives

$G(S(t)) = S_A(G(t))$, and the induction hypothesis replaces $G(t)$ by $F_A(t)$. At a sum or product, preservation expresses the result using the two interpreted constituents; the two induction hypotheses replace them by their F_A values. Thus $G(t) = F_A(t)$ in every constructor case. $\square$

This theorem concerns expressions. It does not make every realization a model of arithmetic. To interpret the arithmetic rules soundly, the operations must satisfy those rules. The next construction supplies a concrete realization for which the necessary agreement is proved.

8.2 Arithmetic on lists of one mark

Let a list contain repetitions of a single mark $\bullet$. The empty list is written $[]$; placing a mark at the front of u is written $\bullet :: u$. There is no additional information in a list entry: every entry is the same mark.

Definition 8.3 (The list arithmetic). Take finite lists of the single mark as the carrier. Define zero to be the empty list, successor to prepend one mark, and addition to be concatenation, written $u +_L v = u ++ v$. Define multiplication by recursion on the second list:

$$\begin{aligned} u \cdot_L [] &= [], \\ u \cdot_L (\bullet :: v) &= (u \cdot_L v) +_L u. \end{aligned}$$

These operations act directly on lists. Concatenation joins the two lists; multiplication appends one copy of the first list for every mark in the second. We can therefore compare this construction with ordinary arithmetic after both descriptions have been specified.

Write $L(u)$ for the length of a list. Write $R(n)$ for the list of n repetitions of the mark.

Theorem 8.4 (The two translations are inverse). *For every natural number n and every list u of the single mark,*

$$L(R(n)) = n, \quad R(L(u)) = u.$$

Length preserves zero, successor, addition and multiplication:

$$\begin{aligned} L([]) &= 0, & L(\bullet :: u) &= L(u) + 1, & (8.1) \\ L(u +_L v) &= L(u) + L(v), & L(u \cdot_L v) &= L(u)L(v). & (8.2) \end{aligned}$$

In particular, lists of this one mark are equal exactly when their lengths are equal.

Proof. Induction on n proves $L(R(n)) = n$: the empty repetition has length zero, and one further repetition increases length by one. Induction on u proves the other round trip. The empty list is immediate. A nonempty list has the form $\bullet :: v$, since there is only one mark. Replicating its length produces $\bullet :: R(L(v))$, which is $\bullet :: v$ by the induction hypothesis.

The start and successor equations are the definition of length. For concatenation, induct on the first list. The empty first list leaves v , while removing one mark from a nonempty first list reduces both sides by one and leaves the induction hypothesis.

For multiplication, induct on v . At the empty list, both sides are zero. At $\bullet :: v$, the definition and the addition identity give

$$\begin{aligned} L(u \cdot_L (\bullet :: v)) &= L(u \cdot_L v) + L(u) \\ &= L(u)L(v) + L(u) \\ &= L(u)(L(v) + 1). \end{aligned}$$

This is the required successor case.

Finally, equal lists have equal lengths. If their lengths are equal, their replications are equal, and the second round-trip identity gives equality of the original lists. $\square$

Both carriers also satisfy the faithful-generation conditions of Chapter 2. A nonempty list is not empty, equality of two lists with a mark prepended gives equality of their tails, and list induction reaches every list. The corresponding facts for natural numbers were already used in Part I. Thus the translation preserves the starting point and step as well as the recursive arithmetic operations.

The one-mark condition is essential to this example. Its role is visible in the round-trip proof: knowing the length reconstructs every entry because every entry has already been specified to be the same mark.

8.3 Terms agree under corresponding assignments

A formula may contain variables, so carrier translations alone are insufficient. We must translate the assignments to those variables. Let ρ_N assign natural numbers and let ρ_L assign lists. Call the assignments *related* when

$$L(\rho_L(k)) = \rho_N(k) \quad \text{for every position } k. \quad (8.3)$$

Evaluate a term on lists by the same recursive clauses as before, using the list operations just defined. Write this value as $E_{\rho_L}^L(t)$, and the natural-number value as $E_{\rho_N}^N(t)$.

Lemma 8.5 (Evaluation commutes with the translation). *For related assignments and every arithmetic term t ,*

$$L(E_{\rho_L}^L(t)) = E_{\rho_N}^N(t).$$

Proof. Induct on t . The variable case is (8.3). The zero case is $L([\]) = 0$. The successor case follows from the length identity for successor and the induction hypothesis. For a sum, length of concatenation is the sum of lengths; apply the two induction hypotheses. For a product, use the multiplication identity in (8.2) and the same two induction hypotheses. These are all the term constructors. $\square$

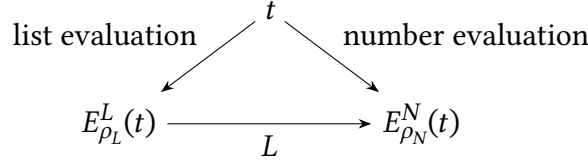


Figure 8.1. Related assignments make the two routes agree. The equation is proved for every term by following its construction.

8.4 Truth is preserved in both directions

Interpret formulas on lists directly. An equality says that its list values are equal. The propositional connectives keep their constructive readings. A universal quantifier ranges over every finite list of the mark, and an existential quantifier supplies one such list as a witness. Write this satisfaction relation as $\rho_L \models_L \varphi$.

Theorem 8.6 (Invariance of every arithmetic formula). *If ρ_N and ρ_L are related, then for every formula of the language in Chapter 5,*

$$\rho_L \models_L \varphi \iff \rho_N \models \varphi. \quad (8.4)$$

Proof. Induct on the formula, with the two assignments arbitrary. For an equality of terms, equal list values give equal lengths, and Lemma 8.5 gives equal number values. Conversely, equal number values give equal lengths, which give equal lists by Theorem 8.4. Falseness is false on both sides.

For conjunction and disjunction, transport each component by its induction hypothesis. This preserves both the pair of proofs and the chosen side of a disjunction. For an implication, both directions of each component equivalence are needed. Given the implication in one interpretation, transport an antecedent from the other, apply the implication, and transport the consequent back. The reverse direction uses the same procedure with the roles interchanged.

For a universal formula, suppose first that its list interpretation holds. Choose any natural number n and use $R(n)$ for the new list variable. The extended assignments are related because $L(R(n)) = n$. The induction hypothesis transports the body to the number interpretation at n . Since n was arbitrary, the universal number interpretation holds.

Conversely, suppose the universal number interpretation holds and choose any list u . Use $L(u)$ for the new number variable. The extended assignments are related, and the induction hypothesis transports the body at $L(u)$ to the list body at u . This proves the list universal statement.

For an existential formula, a list witness u becomes the number witness $L(u)$; a number witness n becomes the list witness $R(n)$. In each direction the extended assignments are related, and the induction hypothesis transports the witness's property. This covers every formula constructor. $\square$

The quantifier cases are why a one-way preservation result would have been insufficient. We needed every number to have a corresponding list and every list to have a corresponding

number, with the assigned values related. The round trips supply exactly that information. Section 8.5 measures what survives when only the first of them is available.

Corollary 8.7 (Validity is independent of these two presentations). *A formula holds under every number assignment if and only if it holds under every list assignment. Every closed derivation with an empty ledger has a conclusion true in both presentations.*

Proof. For a given list assignment, take the lengths at every position to obtain a related number assignment. For a given number assignment, replicate each assigned number to obtain a related list assignment. Apply Theorem 8.6 in each direction. For the last assertion, soundness gives validity in the number interpretation, and the first assertion gives validity in the list interpretation. $\square$

8.5 What survives a one-way translation

The list carrier came with translations in both directions. Many structures come with only one. Take a structure M for the arithmetic language, as in Definition 5.7: a carrier with an element 0_M and operations S_M , $+_M$ and $\cdot_M$, in which formulas are read by the clauses of Chapter 5 with quantifiers ranging over the whole carrier. Write $M, \sigma \models \varphi$ when an assignment σ of elements of M satisfies φ . Suppose that the six arithmetic rules of Chapter 5 hold in M as facts about its elements:

$$\begin{aligned} S_M(x) \neq 0_M, \quad S_M(x) = S_M(y) \Rightarrow x = y, \\ x +_M 0_M = x, \quad x +_M S_M(y) = S_M(x +_M y), \quad x \cdot_M 0_M = 0_M, \quad x \cdot_M S_M(y) = x \cdot_M y +_M x. \end{aligned}$$

Nothing is assumed about induction in M . Write $S^n(0)$ for the term with n successors, the *numeral* of n . Its value in M is $v(n) = S_M^n(0_M)$, and $v\rho$ denotes the assignment $k \mapsto v(\rho(k))$.

Theorem 8.8 (The numerals form a copy of the numbers). *Under these six conditions, v is injective and satisfies*

$$v(m + n) = v(m) +_M v(n), \quad v(mn) = v(m) \cdot_M v(n).$$

Consequently, for every assignment ρ of natural numbers and every quantifier-free formula ψ :

1. $M, v\rho \models \psi$ holds exactly when $\rho \models \psi$;
2. if $\rho \models \exists x_1 \cdots \exists x_k \psi$, then $M, v\rho \models \exists x_1 \cdots \exists x_k \psi$;
3. if $M, v\rho \models \forall x_1 \cdots \forall x_k \psi$, then $\rho \models \forall x_1 \cdots \forall x_k \psi$.

Proof. The map v is the reading of the orbit into the pointed step system $(M, 0_M, S_M)$ constructed in Chapter 2, written in natural-number coordinates. Its injectivity is the argument of Lemma 2.5. That argument used only separation and injectivity of the step, and its induction runs over the numbers, not over M . For addition, induct on n : $v(m + 0) = v(m) = v(m) +_M 0_M$, and

$$v(m + n + 1) = S_M(v(m + n)) = S_M(v(m) +_M v(n)) = v(m) +_M v(n + 1).$$

Multiplication follows the same pattern, using the addition identity at the successor step. By induction on terms, every term evaluates under $\nu\rho$ to the numeral of its value under ρ .

For (1), induct on the formula. An equation holds in M when two numerals are equal, which by injectivity happens exactly when the two values are equal. Falsehood and the connectives transport componentwise. For (2), induct on k : a witness n in $\mathbb{N}$ becomes the witness $\nu(n)$ in M , the extended assignments are again related by ν , and (1) transports the quantifier-free body. For (3), instantiate each universal quantifier of M at numerals and apply (1). $\square$

The two remaining directions fail, and one structure shows both. The structure W of Section 5.7 adds two points a and b to the natural numbers. It keeps the operations of $\mathbb{N}$ on numbers and sets, for every number n and every element x ,

$$S(a) = a, \quad S(b) = b, \quad a + n = a, \quad b + n = b, \quad x + a = x + b = b;$$

a product is 0 when either factor is 0, and b when neither factor is 0 and one of them is a or b . Section 5.7 checks that the six conditions hold in W . For example, $x + S(a) = x + a = b = S(b) = S(x + a)$, and $a \cdot S(0) = b = 0 + a = a \cdot 0 + a$. The sentence $\exists x(S(x) = x)$ is true in W , witnessed by a , and false in $\mathbb{N}$. The sentence $\forall x \neg(S(x) = x)$ is true in $\mathbb{N}$ and false in W . So existential statements need not come down, and universal statements need not go up.

The same two sentences settle the question for every pattern of quantifiers. A *prefix* is a finite string $Q_1x_1 \cdots Q_kx_k$ in which each Q_j is $\forall$ or $\exists$ and each quantifier binds a new variable; placing it before a quantifier-free formula ψ gives a formula with that prefix. Say that a prefix *goes up* when, for every structure M obeying the six conditions, every quantifier-free ψ and every assignment ρ of natural numbers, the formula with that prefix that holds at ρ in $\mathbb{N}$ also holds at $\nu\rho$ in M . Say that it *comes down* when the converse holds.

Proposition 8.9 (No other prefix transfers). *A prefix goes up exactly when all its quantifiers are existential, and comes down exactly when all its quantifiers are universal; only the empty prefix does both. If a prefix contains a universal quantifier, some quantifier-free ψ gives a formula with that prefix that is true in $\mathbb{N}$ and false in W under every assignment. If it contains an existential quantifier, some ψ gives one that is true in W and false in $\mathbb{N}$ under every assignment.*

Proof. Theorem 8.8 gives the positive directions, the empty prefix by its first assertion. Suppose the prefix contains a universal quantifier, and let x be its variable. Take ψ to be $\neg(S(x) = x)$. Every other quantifier of the prefix binds a variable that ψ does not mention, so it changes nothing, in $\mathbb{N}$ or in W , because both carriers have elements. The formula therefore holds exactly when $\forall x \neg(S(x) = x)$ does: in $\mathbb{N}$, and not in W , where $S(a) = a$. If the prefix contains an existential quantifier with variable x , take ψ to be $S(x) = x$. The formula then holds exactly when $\exists x(S(x) = x)$ does: in W , witnessed by a , and not in $\mathbb{N}$. $\square$

The proposition is about all formulas with a given prefix, not about each formula separately. A single universal sentence can go up. The sentence $\forall x(x + 0 = x)$ holds in every structure obeying the six conditions, because it is one of them.

The failure has one source: the numerals miss a and b . In the list carrier the second round trip supplied every list as the expansion of its length; here nothing supplies a numeral for a . Supplying one is the work of the generation condition of Definition 2.4, which makes the

reading reach every element by Lemma 2.6. The theorem shows how far the first round trip reaches without it.

8.6 What happens to the proof and its ledger

Throughout this comparison, the formula and its derivation remain the same finite expressions. We changed their interpretation. A rule labelled EM in a derivation is still that rule, and a quantified induction formula still contains a quantifier. The resource scan of Theorem 6.5 reads those expressions; it does not read their interpreted number or list values. It therefore retains the same record for this change of interpretation.

For example, the reflexivity-and-disjunction proof of $(0 = 0) \vee \neg(0 = 0)$ remains the same empty-ledger derivation. Its formula holds whether zero is interpreted as a number or as the empty list. The alternative proof invoking EM still records that invocation. A change of carrier does not remove an occurrence from a proof tree.

This is the exact scope of the accounting claim. Here the syntax and rules are fixed, the carrier translation is explicit, and satisfaction is preserved and reflected for every formula. Translating the tree into a different proof language is a separate task, and each such translation needs its own correspondence theorem. Section 8.7 proves such a correspondence for the standard presentations of Peano arithmetic and of Heyting arithmetic. It does not cover other proof languages, or procedures for accepting trees in general.

8.7 A second proof language: Peano arithmetic

The standard first-order presentation of arithmetic is Peano arithmetic. It was formulated with no ledger in view, so it is a fair test of what the rules of Chapters 5 and 6 describe. We state it over the same terms and formulas, with the same lifting and substitution operations.

Definition 8.10 (Peano arithmetic). The *axioms* of Peano arithmetic are all instances, for terms t and u , of

$$\neg(S(t) = 0), \quad S(t) = S(u) \rightarrow t = u, \\ t + 0 = t, \quad t + S(u) = S(t + u), \quad t \cdot 0 = 0, \quad t \cdot S(u) = t \cdot u + t.$$

A derivation of $\Gamma \vdash_{\text{PA}} \varphi$ is a finite tree built from hypotheses in Γ , axioms, reflexivity, equality substitution, the introduction and elimination rules for $\wedge$, $\vee$, $\rightarrow$, $\forall$ and $\exists$ with the freshness conditions of Chapter 5, the rule for an impossible case, induction for every formula, and excluded middle, $\Gamma \vdash_{\text{PA}} \varphi \vee \neg\varphi$ for every formula φ . No ledger appears in the definition.

The two systems differ in three visible places. Injectivity of the successor is an axiom here and a rule there. Excluded middle is an ordinary rule here and a recorded additional principle there. Induction is unrestricted in both, but only the rules of Chapter 6 record whether its formula contains a quantifier. Deleting excluded middle from the definition gives the usual presentation of Heyting arithmetic, the intuitionistic counterpart of Peano arithmetic.

Theorem 8.11 (The rules with excluded middle are Peano arithmetic). *For every finite context Γ and every formula φ , the following are equivalent:*

1. $\Gamma \vdash_{\text{PA}} \varphi$;
2. φ has a derivation from Γ by the rules of Chapters 5 and 6 whose ledger has false LPO and MP entries.

Moreover, the translation from (1) to (2) produces a derivation whose EM entry is true exactly when the Peano derivation uses excluded middle, and whose induction entry is true exactly when it uses induction on a formula containing a quantifier.

Proof. From (1) to (2), translate the Peano derivation node by node. Hypotheses, reflexivity, equality substitution, the connective and quantifier rules, the rule for an impossible case and induction have identical counterparts. Five of the axioms are rules of Chapter 5 with no premises. The injectivity axiom becomes a short derivation: assume $S(t) = S(u)$, apply the injectivity rule, and discharge the assumption. Excluded middle becomes an EM leaf. No LPO or MP leaf is ever produced. By Theorem 6.5, the ledger of the result is read from its EM leaves and its induction nodes, and these correspond exactly to the uses of excluded middle and the inductions of the original tree, with the same induction formulas.

From (2) to (1), translate the derivation node by node in the same way. The injectivity rule becomes the injectivity axiom followed by implication elimination, and an EM leaf becomes excluded middle. By Theorem 6.5, the ledger hypothesis excludes LPO and MP leaves, so no other case arises. Both translations preserve contexts, including the lifted contexts of the quantifier rules, because the two systems share their formulas and their lifting and substitution operations. $\square$

Two consequences are worth recording. A Peano derivation that never uses excluded middle becomes a posit-free derivation. By Corollary 6.7, every environment satisfying its hypotheses then satisfies its conclusion, with none of EM, LPO or MP assumed in the metatheory. If, in addition, its inductions are on quantifier-free formulas only, the translation has the empty ledger.

Conversely, a posit-free derivation translates into a Peano derivation that never uses excluded middle. The posit-free derivations are therefore exactly the translations of derivations in Heyting arithmetic, the system obtained by deleting excluded middle from Definition 8.10, up to one point of form that the next corollary states exactly.

Corollary 8.12 (The posit-free derivations are Heyting arithmetic). *For every finite context Γ and every formula φ , the following are equivalent:*

1. φ has a derivation from Γ in Heyting arithmetic, that is, a Peano derivation that never uses excluded middle;
2. φ has a posit-free derivation from Γ .

Moreover, the translation from (1) to (2) in Theorem 8.11 turns every derivation in Heyting arithmetic into a posit-free derivation, and every posit-free derivation has the judgment and the ledger of such a translation. The two trees differ only where the posit-free derivation applies the injectivity rule to a premise: the translation there derives $S(t) = S(u) \rightarrow t = u$, by the injectivity rule under a discharged assumption, and applies implication elimination.

Proof. Restrict both translations in the proof of Theorem 8.11 to trees without excluded middle. From (1) to (2), no EM leaf is produced, and no LPO or MP leaf ever is, so the result is posit-free. From (2) to (1), a posit-free derivation has no EM, LPO or MP leaf, by Theorem 6.5, so its translation never uses excluded middle. For the last assertion, translate a posit-free derivation d into Heyting arithmetic and back. Every node returns unchanged except an application of the injectivity rule, which returns as the injectivity axiom, rederived under a discharged assumption, followed by implication elimination. That short derivation has the same conclusion and the same context, and its only new leaf is a hypothesis, which adds nothing to the ledger. The induction nodes and their formulas are those of d . So the result has the judgment and the ledger of d . $\square$

The theorem and its corollary compare proof systems over one language. They are not proofs that any of these systems is consistent, and they say nothing about languages with further types or symbols; Section 16.3 takes up that comparison for the typed language of Chapter 9. What they settle are the names of the two fragments: allowing excluded middle and nothing else, the rules of Chapter 5 prove exactly the theorems of Peano arithmetic, and allowing no added principle, exactly the theorems of Heyting arithmetic. In both, the ledger shows where the quantified inductions occur, and in the first, where the classical step occurs.

8.8 A quotient represents what a reading retains

Chapter 7 studied readings that can identify different counts. Their quotients provide a second useful comparison of representations.

Let r preserve zero and addition from the orbit into an additive commutative monoid V . Its *range* consists of the values actually reached by r . Its kernel identifies x and y exactly when $r(x) = r(y)$. Write $[x]$ for the class of x under that kernel.

Theorem 8.13 (The kernel quotient and the range). *The map*

$$[x] \mapsto r(x)$$

is an addition-preserving bijection from the kernel quotient onto the range of r . If the kernel is the index-period congruence with parameters i, p , the corresponding index-period quotient represents that same range.

Proof. If x and y represent the same class, the definition of the kernel gives $r(x) = r(y)$. Hence the displayed map is well defined. It preserves zero and addition because r does. If two classes have the same image, their representatives have equal readings and therefore belong to the same kernel class; this proves injectivity. Every value in the range is $r(x)$ for a representative x , so it is the image of $[x]$; this proves surjectivity. If the kernel is the specified index-period relation, replacing the relation by that equal description gives the corresponding quotient. $\square$

The target of this bijection is the range, not automatically all of V . That distinction prevents unused target values from being counted as information carried by the reading. It also explains the role of the classification: the parameters describe precisely the retained identifications, while this theorem connects their quotient to the values the reader actually reaches.

8.9 The limits are part of the invariance theorem

The constructions in this chapter give positive reasons for independence from a notation. The arithmetic operations were defined in two ways, the translations were specified, and each connective and quantifier was checked under those translations. The resource record stayed attached to the same finite derivation.

The hypotheses exclude concrete failures. Consider a carrier with one element, using that element both as zero and as the successor of every element. It has operations for the term signature, so the theorem giving a unique interpretation of terms applies. But its zero is a successor. It therefore fails the faithful-generation conditions and does not supply the arithmetic interpretation just proved invariant. A mere assignment of operations is not enough.

The surrounding mathematics also remains present. Finite expressions, functions, equality, recursion and the interpretation of quantifiers are the tools used to state and prove the equivalence. We have proved a particular representation-independence result with those tools. The next part examines the different kinds of recursive construction themselves, including what becomes available when their rules grow stronger.

Questions to work through

1. In the list round-trip proof, identify the use of the assumption that the list has only one possible mark. Which later proof depends on the resulting injectivity of length?
2. Explain both directions of the universal-quantifier case in Theorem 8.6. Which translation supplies the new variable in each direction?
3. Why does the implication case require preservation and reflection of its components, rather than preservation alone?
4. In Theorem 8.13, locate the exact use of the kernel definition. Why is surjectivity onto the range immediate, while surjectivity onto the whole target was not claimed?
5. Verify in W that $x \cdot S(y) = x \cdot y + x$ when x is a number $n > 0$ and $y = a$, and when $x = y = a$. Why does $\forall x \neg(S(x) = x)$ fail in W while every numeral instance $\neg(S(S^n(0)) = S^n(0))$ holds there?
6. The one-element carrier at the end of this chapter has operations for the term signature. Which of the six conditions of Theorem 8.8 fails there, and which conclusion of the theorem fails with it?
7. Translate the Peano axiom $S(t) = S(u) \rightarrow t = u$ into a derivation by the rules of Chapter 5 and record its ledger. Then translate a use of the injectivity rule back into Peano arithmetic.
8. A Peano derivation of $\forall x(0 + x = x)$ uses induction on $0 + x = x$. What is the ledger of its translation? Compare it with the example in Chapter 6.

9. For the prefix $\exists y \forall x$, write the formula that Proposition 8.9 uses to show that it does not go up, and the one that shows it does not come down. Check each in $\mathbb{N}$ and in W . Why does $\forall x (x + 0 = x)$ not contradict the proposition?

Part III

Construction, iteration and the continuum

Chapter 9

Iteration, recursion and induction

Repeating an operation, using the position of a step, and proving a statement at every position are different abilities. On the finite tallies constructed in Chapter 1, we have all three. That can make their differences difficult to see. To separate them, we must ask what each rule would permit on a carrier whose structure has not already been settled.

This chapter does that with two pictures. In the first, two distinct roots lead to the same successor. An iterator works perfectly well there. In the second, two chains run alongside one another. Indexed recursion works on both, although only one begins at the designated start. These examples locate the work done by the stronger rules.

The conclusions concern a precisely specified language. We will give its rules, construct the examples, and prove that the examples respect every rule they are meant to interpret. That last step matters: an appealing picture alone cannot establish that a statement is underivable.

9.1 Three ways to proceed from a start

Write O for an orbit carrier, $0 : O$ for its designated start, and $S : O \rightarrow O$ for its successor operation. At this point, these are a type and two operations. We have not assumed that S is injective or that every element of O is reached from 0 .

An *iterator* has a fixed target type B , a starting value $b : B$, and a step $g : B \rightarrow B$. Its equations are

$$l(B, b, g, 0) \equiv b, \quad (9.1)$$

$$l(B, b, g, Sx) \equiv g(l(B, b, g, x)). \quad (9.2)$$

The symbol $\equiv$ means equality by the computation rules of the language. The step sees the previous result. It need not be told the position at which it is acting.

An *indexed recursor* still has a fixed target B , but its step receives both the position and the previous result:

$$g : \prod_{x:O} (B \rightarrow B).$$

Its equations are

$$R(B, b, g, 0) \equiv b, \quad (9.3)$$

$$R(B, b, g, Sx) \equiv g(x, R(B, b, g, x)). \quad (9.4)$$

The notation $\prod_{x:O}$ denotes a dependent function type: a function takes an $x : O$ and returns a value of the type written after the product sign. Here that return type happens to be the same $B \rightarrow B$ for every x .

A *dependent eliminator* allows the target itself to depend on the position. Given a type expression $P(x)$, a value $b : P(0)$, and

$$g : \prod_{x:O} (P(x) \rightarrow P(Sx)),$$

it provides $E(P, b, g, x) : P(x)$, with equations

$$E(P, b, g, 0) \equiv b, \tag{9.5}$$

$$E(P, b, g, Sx) \equiv g(x, E(P, b, g, x)). \tag{9.6}$$

When $P(x)$ states a property, this is induction: supply a proof at the start and a method for carrying a proof through one step, then obtain a proof at each position. When $P(x)$ describes a kind of data, the same rule constructs that data.

A family asks a different question at each position

Here is the family that will later separate indexed recursion from dependent elimination. Use recursion to rebuild a position from the designated start:

$$c(x) = R(O, 0, \lambda z : O. \lambda y : O. Sy, x).$$

Its equations say $c(0) \equiv 0$ and $c(Sx) \equiv S(c(x))$. Now ask for an identity proof between the rebuilt position and the supplied one:

$$P(x) = \text{Id}_O(c(x), x).$$

The input x occurs in the question itself. At the start, the question is the identity of 0 with itself. At a successor, an answer to the previous question can be carried through S . A dependent eliminator would put those answers together into a proof of $P(x)$ at every x . The construction of the step proof is given explicitly in Lemma 9.5.

The two-chain picture will explain why the last step matters. Recursion can rebuild both chains into the first one, so rebuilding an element of the second chain need not recover that element. Having a recursive function c is therefore different from having the requested identity proof at every input. A dependent function of type $\prod_{x:O} P(x)$ must return a proof of the question belonging to the input it receives.

These displays already suggest two inclusions. Indexed recursion can ignore the position and act as an iterator. Dependent elimination can use a constant family $P(x) = B$ and act as indexed recursion. The interesting question is whether either inclusion can be reversed.

9.2 The language in which the question is exact

We use a small typed language. Its type expressions are generated by

$$O, \quad \mathbf{0}, \quad \prod_{x:A} B(x), \quad \text{Id}_A(t, u), \quad D(t).$$

Here $\mathbf{0}$ is the empty type. An element of $\text{Id}_A(t, u)$ is an identity proof between two terms of type A . The last form, $D(t)$, is optional; we will specify exactly when it is available.

A context is a finite list of typed variables. Each type in the list may use variables declared earlier. A type family $P(x)$ is a type expression in a context containing x ; it is not an arbitrary family borrowed from the mathematics in which we discuss the language. There are no variables standing for types or type families, and no type whose elements are all types. In particular, there is no universe inside this language. There is also no dependent-pair constructor.

Terms consist of variables, 0 , successors, functions, applications, identity proofs and their eliminator, empty-type elimination, and the three orbit operations just displayed. We consider three fragments: the iterator fragment permits **I**; the indexed fragment additionally permits **R**; the dependent fragment additionally permits **E**.

The typing rules are as follows. The empty context is valid, and a valid context may be extended by a type formed in it. The types O and $\mathbf{0}$ may be formed in every valid context. A product type requires its domain A in the context and its codomain $B(x)$ in the context extended by $x : A$. An identity type requires A and two terms of that type. The type $D(t)$, when enabled, requires $t : O$.

A declared variable has its declared type. The term 0 has type O , and S sends terms of type O to terms of type O . Function introduction and application have the familiar rules

$$\frac{\Gamma, x : A \vdash t : B(x)}{\Gamma \vdash \lambda x : A. t : \prod_{x:A} B(x)}, \quad \frac{\Gamma \vdash f : \prod_{x:A} B(x) \quad \Gamma \vdash u : A}{\Gamma \vdash f(u) : B(u)},$$

with the displayed types required to be formed. The notation $\Gamma \vdash t : A$ says that the term t has type A under the assumptions in Γ .

Reflexivity supplies $\text{refl}_x : \text{Id}_A(x, x)$. The identity eliminator, called J , works as follows. Fix $x : A$ and a type family $Q(y, p)$ in the context $y : A, p : \text{Id}_A(x, y)$. From $d : Q(x, \text{refl}_x)$ it supplies

$$J(A, x, Q, d, y, p) : Q(y, p), \quad J(A, x, Q, d, x, \text{refl}_x) \equiv d.$$

Thus a construction on an identity proof is specified at reflexivity and transported along that proof. Empty-type elimination supplies a term of any formed type from a term of $\mathbf{0}$.

The typing rules for **I**, **R** and **E** require exactly the types of their arguments stated in the preceding section, and $x : O$. All earlier variables in the context remain available as parameters. Computation uses the six orbit equations, the J equation, and function evaluation

$$(\lambda x : A. t)(u) \equiv t[u/x].$$

An equation is available only in a fragment that permits its orbit operation.

We optionally add the *start discriminator*, with equations

$$D(0) \equiv \text{Id}_O(0, 0), \quad D(Sx) \equiv \mathbf{0}. \quad (9.7)$$

It gives a type-level distinction between the start and a successor. The superscripts $-$ and $+$ will mean that this discriminator is disabled or enabled, respectively.

For precision, raw conversion is the least equivalence relation containing the enabled computation equations and closed under every expression constructor and capture-avoiding substitution. It is defined on raw expressions: an intermediate expression in a conversion chain need not be well typed. A *typed* equality requires both endpoints to have the stated type; a type equality requires both endpoint types to be formed. A term may be converted from A to B only through such a type equality. We do not turn an identity proof into a raw conversion, and we add neither function extensionality nor an η equation.

The word “raw” deserves an example. As an expression, the function application $(\lambda x : \mathbf{0}. x)(0)$ rewrites to 0 . But its function expects an input from the empty type, while the supplied 0 has type O . The application cannot be typed. A soundness proof cannot therefore justify every intermediate raw rewrite merely by appealing to the typing of the original endpoints. Section 9.8 handles this issue in two steps: typed expressions satisfy a safety condition, and convertible safe endpoints reduce forward to a common expression without losing their interpreted value. That is why the later proof needs both safety and confluence.

Finally, a substitution from a context Γ to a context Δ assigns to each declared variable of Γ a term of its substituted type in Δ . Substitution acts on types, terms and typed equalities. Only the finitely many variables in the source context are required. These formation, introduction, elimination, conversion and substitution rules constitute the whole calculus used below.

Definition 9.1 (Admissibility). A rule is admissible in a fragment if, whenever its premises have derivations in that fragment, its conclusion has a derivation there too. For indexed recursion and dependent elimination, we require a term with both displayed computation equations as typed equalities, uniformly over every finite parameter context and every permitted type expression.

This definition makes a negative result strong enough to be useful. It rules out every construction from the available rules, including one we have not thought to try. A single instance with derivable premises and an underivable conclusion refutes admissibility.

9.3 What indexed recursion immediately gives

The additional position argument in R has a direct use. Choose target O , base 0 , and step $(x, y) \mapsto x$. Define

$$\text{pred}(x) = R(O, 0, \lambda z : O. \lambda y : O. z, x). \quad (9.8)$$

Then the computation rules give

$$\text{pred}(0) \equiv 0, \quad \text{pred}(Sx) \equiv x.$$

The second equation is a left-inverse law for successor.

Theorem 9.2 (Indexed recursion supplies injectivity). *In either indexed fragment, with or without D , there is a term of*

$$\prod_{x:O} \prod_{y:O} (\text{Id}_O(Sx, Sy) \rightarrow \text{Id}_O(x, y)).$$

Proof. The term in (9.8) is well typed because its step accepts two orbit elements and returns the first. Given $p : \text{Id}_O(a, b)$, identity elimination applies pred to that path: use the family $Q(z, q) = \text{Id}_O(\text{pred}(a), \text{pred}(z))$ and the base $\text{refl}_{\text{pred}(a)}$. This constructs a proof of $\text{Id}_O(\text{pred}(a), \text{pred}(b))$. Set $a = Sx$ and $b = Sy$. The two predecessor computations convert the resulting type to $\text{Id}_O(x, y)$. Abstracting over x, y, p gives the required term. No discriminator is used. $\square$

The start discriminator gives a different fact.

Theorem 9.3 (The discriminator separates the start). *In every fragment with D enabled, there is a term of*

$$\prod_{x:O} (\text{Id}_O(0, Sx) \rightarrow \mathbf{0}).$$

Proof. By (9.7), refl_0 is a term of $D(0)$. If $p : \text{Id}_O(0, Sx)$, apply identity elimination with family $Q(y, q) = D(y)$ and that base term. The result has type $D(Sx)$, which converts to $\mathbf{0}$. Abstract over x and p . The construction uses neither recursion nor induction. $\square$

The two arguments identify two different resources. The predecessor retrieves the input of a successor. The discriminator distinguishes the start from a successor. Neither argument silently assumes the other fact.

9.4 An iterator can merge two roots

Consider distinct elements $a, u, v_1, v_2, \dots$, with designated start a and step

$$S(a) = v_1 = S(u), \quad S(v_n) = v_{n+1} \quad (n \geq 1).$$

The subscripts here are the finite natural numbers of the ambient mathematics. They are being used to construct a model of the language; they do not add an induction rule to that language.

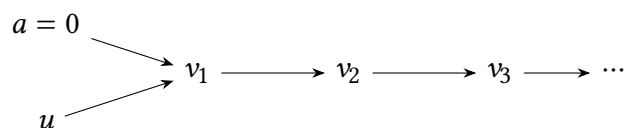


Figure 9.1. Two roots with one common tail. Iteration can give both roots the same starting value, so its equations do not detect the merger.

For any set B , any $b \in B$, and any function $g : B \rightarrow B$, define

$$I(a) = b, \quad I(u) = b, \quad I(y_n) = g^n(b). \quad (9.9)$$

Both roots satisfy $I(Sx) = g(I(x))$: each side is $g(b)$. At a tail element, the equation is $g^{n+1}(b) = g(g^n(b))$. Thus the iterator exists for every target and step, although successor is not injective.

The discriminator can be interpreted too. Give $D(a)$ and $D(u)$ the one-element set, and give each $D(v_n)$ the empty set. Then $D(a)$ interprets $\text{Id}(a, a)$, and every $D(Sx)$ is empty. The extra root u is allowed to share the start's discriminator value; the discriminator equations never require the converse implication $D(x) \Rightarrow x = 0$.

Theorem 9.4 (The iterator boundary). *In the fixed calculus of Section 9.2, the iterator fragment, with either choice of discriminator, has no closed term proving successor injectivity. Indexed recursion is not admissible in it.*

Proof. Interpret the orbit type by the merging structure. Interpret product types by sets of dependent functions, the empty type by the empty set, and an identity type by a singleton when its endpoints are equal and the empty set otherwise. The soundness proof in Section 9.8 verifies all the rules of the iterator fragment for this interpretation, in every finite parameter context and with either choice of D .

If an injectivity term existed, apply its interpreted function to a, u and the identity proof between $S(a)$ and $S(u)$. It would produce an element of the empty identity set between the distinct elements a and u . This is impossible.

For indexed recursion, use the closed premises $B = O$, $b = 0$ and $g(x, y) = x$, all expressible in the iterator fragment. An admissible recursor would give a function r with $r(Sx) = x$. In the model this would require both $r(v_1) = a$ and $r(v_1) = u$. Since $a \neq u$, no such function exists. Soundness therefore excludes a term with the required equations. $\square$

The first example answers a common temptation: the ability to repeat an operation does not, by itself, make the underlying step reversible. The position argument in indexed recursion is doing real work.

9.5 Recursion can leave a whole chain unreachable

Now let

$$O = \{0, 1\} \times \mathbb{N}_\delta, \quad 0_O = (0, 0), \quad S(i, n) = (i, Sn).$$

There are two disjoint successor chains. Only the first starts at the designated 0_O .

$$(0, 0) \longrightarrow (0, 1) \longrightarrow (0, 2) \longrightarrow \cdots$$

$$(1, 0) \longrightarrow (1, 1) \longrightarrow (1, 2) \longrightarrow \cdots$$

Figure 9.2. Indexed recursion can start with the same value on both chains. Its step can read the chain's index without making the second root reachable from the designated start.

For a fixed B, b, g , define the recursor on each chain by

$$R(i, 0) = b, \quad R(i, Sn) = g((i, n), R(i, n)). \quad (9.10)$$

The initial equation holds at $(0, 0)$, and the successor equation holds everywhere by construction. There is no conflict in also starting the second chain at b . The equations specify no separate base value there. Ordinary iteration is obtained by choosing a step that ignores (i, n) . Set $D(0, 0)$ to the singleton and $D(i, n)$ to the empty set at every other point. The discriminator equations hold too. Thus even the indexed fragment with the discriminator leaves room for the second chain.

To expose what fails, we need a family expressible in the language. The informal property “belongs to the first chain” would not suffice unless the language could express it. Instead define the term

$$c(x) = R(O, 0, \lambda z : O. \lambda y : O. Sy, x), \quad P(x) = \text{Id}_O(c(x), x). \quad (9.11)$$

The term c rebuilds the successor history from the designated start. It satisfies

$$c(0) \equiv 0, \quad c(Sx) \equiv S(c(x)).$$

In the two-chain model, induction on the finite index gives

$$c(i, n) = (0, n). \quad (9.12)$$

At index zero the equation is the base clause of the recursor. If it holds at n , the successor clause applies S to $(0, n)$ and gives $(0, Sn)$. This proves the formula at every finite index.

Lemma 9.5 (A family with a base and a step). *In either indexed fragment, $P(x)$ from (9.11) is a formed type family, with a closed term of $P(0)$ and a closed term of $\prod_{x:O} (P(x) \rightarrow P(Sx))$.*

Proof. The term $c(x)$ is an indexed recursor with a well-typed step, so its identity type with x is formed. At 0, its computation makes refl_0 a term of $P(0)$. For the step, take $p : \text{Id}_O(c(x), x)$. Apply S to this path using J , with motive $\text{Id}_O(S(c(x)), Sy)$ and reflexivity base. The result is a proof of $\text{Id}_O(S(c(x)), Sx)$. Convert the first endpoint using $c(Sx) \equiv S(c(x))$. This is a proof of $P(Sx)$, and abstraction over x, p gives the required step. $\square$

Theorem 9.6 (The indexed-recursion boundary). *In either indexed fragment of the fixed calculus, there is no closed term of $\prod_{x:O} P(x)$ for this expressed family. Dependent elimination is not admissible. Nor is the rule asserting pointwise uniqueness of two maps with the same start and successor equations.*

Proof. The two-chain interpretation satisfies every rule of the indexed fragment by Section 9.8. At the second root, formula (9.12) gives

$$P(1, 0) = \text{Id}_O((0, 0), (1, 0)).$$

Its interpretation is empty. A function inhabiting $\prod_{x:O} P(x)$ would have to return an element of this empty set at $(1, 0)$, so no such function, and hence no such term, exists.

Lemma 9.5 supplies the premises of dependent elimination inside the indexed fragment itself. If that rule were admissible, it would supply exactly the impossible term.

For uniqueness, compare c with $q(x) = x$. Both are terms $O \rightarrow O$; both satisfy $f(0) = 0$ and $f(Sx) = S(f(x))$, with identity proofs obtained from their computation equations. But at $(1, 0)$, c returns $(0, 0)$ and q returns $(1, 0)$. A term asserting their pointwise equality would again produce an element of an empty identity set there. All the proposed rule's premises have terms, while its conclusion does not. $\square$

The failure occurs at a specific, expressible family. It does not rely on allowing an arbitrary external predicate as an induction motive. The two maps also show why merely solving recurrence equations does not establish uniqueness on a carrier whose generated part may be smaller than the whole.

9.6 Induction does not invent a discriminator

The preceding models separate the two weaker fragments. A different argument shows that, in this language, the start discriminator is still an additional resource even after dependent elimination is available.

Assign a truth value $v(A)$ to every raw type expression:

$$\begin{aligned} v(O) &= \text{true}, & v(\mathbf{0}) &= \text{false}, \\ v(\text{Id}_A(t, u)) &= \text{true}, & v(D(t)) &= \text{true}, \\ v\left(\prod_{x:A} B\right) &= v(A) \Rightarrow v(B). \end{aligned}$$

Here $\Rightarrow$ is ordinary Boolean implication. The valuation deliberately ignores the terms inside a type. It is a tool for checking derivations, not a set interpretation in which identity means equality of denoted elements.

Lemma 9.7 (Substitution cannot change this truth value). *Renaming variables or substituting terms in A preserves $v(A)$. When the discriminator is disabled, raw type conversion also preserves it.*

Proof. For substitution and renaming, induct on the type expression. The orbit and empty types do not change. Identity and discriminator types always have value true, whatever their contained terms. At a product, the two induction hypotheses preserve the antecedent and consequent of the Boolean implication.

For conversion, induct on its construction. Reflexivity, symmetry and transitivity preserve equality of truth values. Product congruence uses its two induction hypotheses. Identity and discriminator congruence preserve the constant value true. The substitution case uses the first part. The two special discriminator equations are unavailable when D is disabled. These are all the type-conversion cases; term conversions inside an identity or a discriminator cannot affect its value. $\square$

Theorem 9.8 (Truth preservation without the discriminator). *In any of the three fragments with D disabled, if $\Gamma \vdash t : A$ and every type declared in Γ has value true, then $v(A) = \text{true}$. In particular every inhabited closed type has value true.*

Proof. Induct simultaneously on the finite derivations of terms, equalities and substitutions. A variable inherits the true value of its declared type; weakening does not change it by Lemma 9.7. Zero and successor have type O . Reflexivity has an identity type, whose value is true.

For a function, if the domain has value false, its product type has value true immediately. If the domain has value true, extend the true context by that domain and apply the induction hypothesis to the body. The codomain is true, so the product is true. For application, a true implication together with a true argument type gives a true codomain. Substitution for the argument preserves that value.

Identity elimination has a base term of $Q(x, \text{refl}_x)$. The induction hypothesis makes this type true. Since substitution does not change the value of Q , the target $Q(y, p)$ is true as well. Empty-type elimination cannot have its premise in a true context: that premise would make $v(\mathbf{0})$ true.

For iteration and indexed recursion, the base term already makes the fixed target B true. For dependent elimination, the base term makes $P(0)$ true; substitution invariance then makes $P(x)$ true at every argument. These observations handle the orbit rules even when the strongest one is enabled.

Type equality preserves the valuation by Lemma 9.7; term equality inherits truth from either typed endpoint. Type conversion therefore preserves the term assertion. A derivable substitution sends a true target context to a true source context, since each source variable has a term of its substituted type in the target. Apply the induction hypothesis to the original judgment in that source context, then use substitution invariance. This handles substituted terms and equalities. Context and type formation add no assertion that a type is inhabited. All derivation rules have now been covered. $\square$

Corollary 9.9 (A strict discriminator boundary). *Without D , even the dependent fragment has no closed term of*

$$\prod_{x:O} (\text{Id}_O(0, Sx) \rightarrow \mathbf{0}).$$

Proof. The inner implication has value $\text{true} \Rightarrow \text{false} = \text{false}$. The outer product has the same false value because $v(O)$ is true. Theorem 9.8 excludes an inhabitant. $\square$

The hypothesis disabling D is essential. When its equations are enabled, $D(S0) \equiv \mathbf{0}$ equates a type assigned true by this valuation with one assigned false. The valuation then ceases to preserve conversion, exactly where the positive construction in Theorem 9.3 becomes available.

9.7 The strict sequence of resources

Let $\mathcal{F}^-$, $\mathcal{R}^-$, and $\mathcal{E}^-$ denote the iterator, indexed and dependent fragments without D . Let $\mathcal{E}^+$ be the dependent fragment with D .

Theorem 9.10 (Three strict extensions). *Every derivation in one of the following fragments is also a derivation, of the same judgment, in the next:*

$$\mathcal{F}^- \subsetneq \mathcal{R}^- \subsetneq \mathcal{E}^- \subsetneq \mathcal{E}^+.$$

The three strictness witnesses are, respectively, successor injectivity, the section $\prod_{x:O} P(x)$ from (9.11), and start separation.

Proof. Each extension retains the earlier grammar and rules. Induct on a conversion derivation to retain its equations, and on a typing derivation to retain its judgment. At a rule that requires an enabled operation, the larger fragment still enables it. Every other rule and every context is unchanged. Thus these are inclusions on actual finite derivations, not only comparisons of their interpretations.

Theorem 9.2 supplies the first witness in $\mathcal{R}^-$, and Theorem 9.4 excludes it from $\mathcal{F}^-$. Lemma 9.5 supplies the inputs for a dependent eliminator; applying that eliminator and abstracting over x gives the second witness in $\mathcal{E}^-$. Theorem 9.6 excludes it from $\mathcal{R}^-$. Finally, Theorem 9.3 supplies start separation in $\mathcal{E}^+$, and Corollary 9.9 excludes it from $\mathcal{E}^-$. $\square$

The first two separations also hold with the discriminator enabled throughout: both countermodels interpret it. The third separation concerns adding the discriminator itself. How much arithmetic the top fragment $\mathcal{E}^+$ proves is a separate question. Chapter 16 answers it: $\mathcal{E}^+$ proves the negative translations of exactly the theorems of Peano arithmetic (Theorem 16.6).

There is no conflict with the arithmetic developed in Part I. There we constructed the finite tallies and used their structural elimination rule; generation from the start was built into the construction. Here we ask which conclusions follow from selected rules on a type O before its full structure has been established. The two countermodels identify precisely what is lost when the stronger rule is withheld. They are not alternative faithful representations of the already constructed tallies.

9.8 Why the countermodels cover every rule

The negative results used a soundness claim for the complete fixed calculus. This section proves that claim. A reader following the main structural argument can return to it after the chapter; its purpose is to close the gap between the two pictures and the statements about all derivations.

9.8.1 A total interpretation and its limits

Work in ordinary classical set theory for this external proof. Encode each orbit carrier injectively as a set, choosing the designated start different from the empty set. The two concrete encodings may be made from tagged ordered pairs of finite tally codes. Extend decoding to all sets by returning the designated start when a set is not an orbit code. Consequently S , I and R have total external interpretations, although their typing rules will permit only arguments in the appropriate domains. The unused indexed recursor in the merging model, and the unused dependent eliminator in both models, receive the default value $\emptyset$.

An environment ρ assigns a set to each variable. Define the interpretation of types recursively by

$$\begin{aligned} \llbracket O \rrbracket_\rho &= \text{the encoded orbit carrier,} \\ \llbracket \mathbf{0} \rrbracket_\rho &= \emptyset, \\ \left[\prod_{x:A} B \right]_\rho &= \prod_{a \in \llbracket A \rrbracket_\rho} \llbracket B \rrbracket_{\rho[x:=a]}, \\ \llbracket \text{Id}_A(t, u) \rrbracket_\rho &= \begin{cases} \{\emptyset\}, & \llbracket t \rrbracket_\rho = \llbracket u \rrbracket_\rho, \\ \emptyset, & \text{otherwise.} \end{cases} \end{aligned}$$

The product consists of all graphs of functions with the stated domain and values in the corresponding fibers. Interpret $D(t)$ by the discriminator set at $\llbracket t \rrbracket_\rho$.

A variable is interpreted by its environment entry. Zero and successor use the model's start and step. A function is interpreted by the graph

$$\llbracket \lambda x : A. t \rrbracket_\rho = \{(a, \llbracket t \rrbracket_{\rho[x:=a]}) : a \in \llbracket A \rrbracket_\rho\}.$$

For application, look for an output paired with the given input in the interpreted graph: choose one if it exists, and use $\emptyset$ otherwise. On a function graph at an input in its domain the output is unique, so this choice has no effect. Reflexivity, empty-type elimination and the unavailable dependent eliminator are interpreted as $\emptyset$. The interpretation of $J(A, x, Q, d, y, p)$ is that of its base d . Finally I and R use the concrete operations (9.9) and (9.10), with interpreted base and step.

All raw expressions now have values. This does *not* make every raw computation equation preserve values. For example,

$$(\lambda x : \mathbf{0}. x)(0) \equiv 0$$

is a raw function-evaluation equation. The function's graph is empty, so the left side has default value $\emptyset$, whereas the right side is the designated nonempty start code. This application has no valid typing derivation. The soundness claim must use its typing hypotheses.

Two elementary facts about the interpretation will be used repeatedly. If $a \in A$, applying the graph of f on A to a returns $f(a)$. If a graph with domain A belongs to a dependent function set with domain B , then $A = B$: the first coordinates of the pairs in either description are exactly its domain. These facts follow directly from the definition of a function graph.

9.8.2 Substitution and safe expressions

For a substitution σ , let $\rho\sigma$ assign to each variable the value of its substituting term in ρ . Structural recursion on expressions gives

$$\llbracket A[\sigma] \rrbracket_\rho = \llbracket A \rrbracket_{\rho\sigma}, \quad \llbracket t[\sigma] \rrbracket_\rho = \llbracket t \rrbracket_{\rho\sigma}. \quad (9.13)$$

For a variable the assertion is the definition of $\rho\sigma$. For a constant it is immediate. At a binder, extend both environments by the same new value and leave the bound variable fixed; the induction hypothesis then identifies the body values. At every remaining constructor, apply its defining operation to the equal constituent values. The J family has two binders and uses two such extensions. This proves both identities, together with their renaming versions.

We next define a property called *safety*, relative to a model and an environment. It records the domain conditions needed by evaluation, without claiming that the whole expression is typed. The definition follows the raw syntax.

For types, O and $\mathbf{0}$ are safe. A product is safe when its domain is safe and its codomain is safe at every value in that domain. An identity type is safe when its two endpoint terms are safe. A discriminator type is safe when its argument is safe.

For terms, variables, zero, reflexivity, empty-type elimination and the unavailable dependent eliminator are safe. A successor is safe when its argument is safe and has value in the orbit carrier. A function is safe when its domain is safe and its body is safe at every value in that domain. An application is safe when both constituents are safe and there is a dependent function set containing the value of its function, whose domain contains the value of its argument. A J term is safe exactly when its base term is safe; the interpretation reads only that base.

An iterator is safe when its base, step and orbit argument are safe, the argument lies in the orbit carrier, and there is a set B containing the base value on which the step value is a function $B \rightarrow B$. For an indexed recursor, replace that last condition by a step function from the orbit carrier to $B \rightarrow B$. The type annotations ignored by these interpretations impose no additional safety condition.

Lemma 9.11 (Safety under substitution). *Suppose e is safe in $\rho\sigma$, and every substituting term for a free variable of e is safe in ρ . Then $e[\sigma]$ is safe in ρ .*

Proof. Induct on e . A variable uses the corresponding substitution hypothesis. Constructors that are safe without conditions remain so. At a successor, application, iterator or recursor, the induction hypotheses give safety of the constituents, while (9.13) preserves all the required set memberships and function domains. At an identity or discriminator type, use the hypotheses for its term arguments. At a product or function, fix a value in the domain, extend the environment and lift the substitution beneath the binder. The new variable is safe by definition; all previous variables retain their hypotheses by renaming. Apply the body induction hypothesis in that extended environment. The J case uses its base hypothesis. This also shows why no condition is needed on substitution components outside the finite set of free variables. $\square$

9.8.3 Joining conversions without changing the language

A conversion can pass through untyped intermediate expressions. We therefore cannot justify it by asserting that every step in its original zigzag preserves values. Instead we join the two endpoints by forward reductions that preserve values whenever they start safely.

Define an external parallel reduction relation $e \Rightarrow e'$. It permits constituents to reduce in parallel under every constructor, including binders. It also permits a visible function application or enabled iterator/recursor equation to contract, with its constituents reduced in parallel. For example,

$$b \Rightarrow b', \quad t \Rightarrow t' \quad \Longrightarrow \quad (\lambda x : A. b)(t) \Rightarrow b'[t'/x].$$

It includes the two enabled discriminator contractions. Finally, it has the external erasures

$$J(A, x, Q, d, y, p) \Rightarrow d' \quad (d \Rightarrow d'),$$

and sends reflexivity, empty-type elimination and raw dependent elimination to refl_0 . These erasures reflect the chosen set interpretation. They are not new computation rules or new equalities in the object language.

Lemma 9.12 (Parallel substitution and joining). *Parallel reduction is preserved by substitution. More generally, if $e \Rightarrow e'$ and each component of σ reduces to the corresponding component of τ , then $e[\sigma] \Rightarrow e'[\tau]$. The parallel relation has the diamond property, and its finite-step closure is confluent. In either weaker fragment, raw-convertible expressions have a common reduct in this external relation.*

Proof. Prove the substitution assertion by induction on the parallel derivation. Constructor cases use their constituent induction hypotheses; beneath a binder, lift both substitutions and retain the bound variable. In the function-contraction case, the substitution composition identity identifies the two results of substituting first or contracting first. The iterator and recursor contractions retain their form under substitution, as do the discriminator contractions. A J erasure reduces to the substituted base, and the other erasures still reduce to refl_0 . This covers every parallel rule.

For each expression e , define its complete development e^* by developing the constituents and contracting the outer redex if its original outer form is a visible redex. Erase J to its developed base and the other erasable forms to refl_0 . An operation disabled by the fragment retains its constructor. A redex newly exposed only after developing a constituent need not be contracted at this stage.

We claim that $e \Rightarrow u$ implies $u \Rightarrow e^*$. Induct on the displayed parallel step. For a constructor whose outer form is not a redex, reduce its constituents by the induction hypotheses. If an application originally has a lambda at its head, a congruence step leaves a lambda there, so contract it using the developed body and argument. If the first step already contracted it, parallel substitution gives the same developed result. For an iterator or recursor at zero, both possibilities lead to the developed base. At a successor, both lead to the developed step applied to the corresponding recursive expression; the recursor also receives the developed predecessor. The discriminator cases lead to the same unit or empty type. A congruence step

on J can be followed by erasure; an erasure step can be followed by development of its base. For the remaining erasable forms both paths end at refl_0 . These exhaust the possible overlaps.

Thus any two parallel steps from e are joined at e^* . This is the diamond property. For finite reduction chains, use the diamond repeatedly to move a single step past a finite chain; induction on the other chain then fills a finite grid of diamonds and produces a common endpoint. This proves confluence of the finite-step closure.

Having a common reduct is consequently an equivalence relation: reflexivity and symmetry are immediate, and confluence joins the two reducts needed for transitivity. It is closed under constructors and substitution by the corresponding parallel rules. Each enabled computation equation has a common reduct: function, iterator, recursor and discriminator equations use their contractions, while the J equation uses erasure. The dependent computation equations are absent in the two weaker fragments. Induction on raw conversion now proves its inclusion in this common-reduct relation. $\square$

Lemma 9.13 (Safe parallel reduction preserves values). *In the merging model for the iterator fragment, and in the two-chain model for the indexed fragment, if e is safe and $e \Rightarrow e'$, then e' is safe and $\llbracket e \rrbracket = \llbracket e' \rrbracket$. Both choices of discriminator are allowed.*

Proof. The concrete iteration and recursion operations stay within their target set whenever their base and step do. For the merging model this follows by induction on the tail level in (9.9). For the two-chain model it follows by induction on the second coordinate in (9.10). The base and successor equations hold by those definitions. The discriminator equations hold by the definitions at the roots and tails.

Now induct on the parallel step, simultaneously for types and terms. At a product, the domain values agree by induction; at each point of that common domain, the fiber values agree too. Their sets of dependent functions are therefore equal. At a lambda, the same pointwise argument gives equality of graphs. Identity and discriminator types depend only on the relevant term values. Successor and application congruence preserve their domain witnesses and values using the constituent equalities. Iterator and recursor congruence similarly preserve their common target set and interpreted operations.

For a function contraction, safety of the application supplies a function-set domain containing the argument. The graph-domain fact identifies that domain with the lambda's declared domain. Graph application therefore equals evaluation of the body with that argument substituted. Use the induction hypotheses for the body and argument, then Lemma 9.11, to obtain both the value and safety of the contracted term. No off-domain default is used.

At an iterator or recursor zero contraction, use its base equation and the base induction hypothesis. At a successor contraction, safety supplies the predecessor in the orbit carrier. The target-closure fact proved above puts the recursive result in B . The step can therefore be applied, once for the iterator and twice for the indexed recursor; its concrete successor equation gives the claimed value. The induction hypotheses preserve all the needed memberships.

The discriminator contractions use its two equations. The interpretation and safety of J are precisely those of its base, so its congruence and erasure cases follow from the base induction hypothesis. Reflexivity, empty-type elimination and raw dependent elimination all have value $\emptyset$, as does refl_0 ; their erasures preserve safety and values directly. These are all the parallel

rules. □

Corollary 9.14 (Coherence at safe endpoints). *In either concrete interpretation, two raw-convertible expressions that are both safe in an environment have equal values there.*

Proof. By Lemma 9.12, the endpoints have a common external reduct. Apply Lemma 9.13 successively along each finite reduction chain. Each chain preserves safety and value. Both original values equal the value of the common reduct, hence equal one another. □

9.8.4 Soundness of the complete typing derivations

Theorem 9.15 (The two concrete models are sound). *Every derivation in the iterator fragment has a valid interpretation in the merging model. Every derivation in the indexed fragment has a valid interpretation in the two-chain model. Both statements hold with or without the discriminator and in every finite parameter context. In particular, a typed term denotes a member of its type, and typed equalities denote equal values.*

Proof. Prove a stronger assertion by simultaneous induction on the derivation. In an environment satisfying its context, every formed type is safe; every typed term is safe and belongs to its interpreted type; both endpoints of each typed equality are safe and have equal values. A derivable substitution takes a satisfying target environment to a satisfying source environment and has safe components at all declared source variables.

Context extension adds exactly the required membership. Variables use their context entries. Orbit and empty-type formation are safe directly. Product formation uses safety of the domain and the codomain in each extended environment; identity and discriminator formation use the safety of their term arguments. Zero belongs to the carrier, and the concrete successor preserves that carrier.

For function introduction, the body induction hypothesis gives a safe value in the correct fiber at every domain point. Its graph therefore belongs to the product type and is safe. For application, the two premises give a dependent function and an argument in its domain. Applying that graph yields a member of the required fiber, and (9.13) identifies the fiber with the substituted result type. The same premises give safety.

Reflexivity denotes $\emptyset$, which is an element of the singleton identity set at equal endpoints. For J , the identity premise says that the two endpoint values agree and the path value is $\emptyset$. Substituting these equal values into the motive makes the target fiber the same as the base fiber, which already contains the base value. Safety comes from that base. Empty-type elimination has no possible premise in a satisfying environment, since nothing belongs to $\emptyset$.

Iteration and indexed recursion use the proved closure of their concrete operations in the target set; their premises supply exactly the safety conditions stated above. Dependent elimination cannot be a rule of either fragment under consideration.

For type or term equality, the induction hypotheses supply safety of both endpoints. Corollary 9.14 then proves equality of their values, despite any untyped intermediates in the raw conversion. Converting a term's type preserves membership because the two interpreted types are equal.

For substitution, the premises type every component associated with a declared source variable. Their membership conclusions give a satisfying source environment, and their safety conclusions cover every variable that can occur in a source judgment. Formula (9.13) transports membership and equalities; the finite-variable form of Lemma 9.11 transports safety. This handles substitution of types, terms and both kinds of equality. It finishes the induction over all the original rules. $\square$

The external set theory and its choices belong to this nonderivability proof. No universe, external family, semantic soundness assumption or additional equation has been inserted into the object calculus. The negative conclusions therefore apply to the exact rules stated at the beginning of the chapter.

Exercises

1. Check the iterator equation separately at a , at u , and at v_n in the merging model. Explain why setting $D(u)$ to the singleton does not violate either discriminator equation.
2. Carry out the two function-evaluation steps in $\text{pred}(Sx)$ after the recursor equation has been applied. Identify the point at which the indexed step uses information unavailable to an ordinary iterator step.
3. Verify the base and successor equations for both c and $q(x) = x$ in the two-chain model. Locate the element at which their pointwise equality fails.
4. Why does dependent elimination preserve truth when the valuation ignores the motive's argument? Which discriminator equation invalidates this valuation?

Chapter 10

Finite construction and effective description

A finite tally can be written down. A rule for producing tallies can also be written down. These are different objects. The first records a result; the second may produce different results when given different inputs, and may fail to finish on some inputs.

There is a third distinction. Suppose that, for every requested accuracy, a description specifies a finite rational interval. Each interval is finite data. It does not follow that one finite rule produces all those intervals. The placement of the word “one” matters. This chapter makes that difference precise, while showing how finite arithmetic and finite programs fit into the same generated records.

10.1 An explicit budget for a tally

Write $L(x)$ for the number of steps in $x \in \mathbb{N}_\delta$, as in Chapter 1. The inverse reading takes an ordinary natural number k to the tally with k steps. The two readings are inverse by their construction. We use ordinary natural numbers to index budgets in the surrounding mathematics.

Definition 10.1 (A finite stage). For a budget n , let

$$F_n = \{x \in \mathbb{N}_\delta : L(x) \leq n\}.$$

If $n \leq m$, the inclusion $j_{n,m} : F_n \rightarrow F_m$ retains the same tally and regards it as satisfying the larger budget.

There are exactly $n + 1$ elements in F_n : the tallies with lengths $0, 1, \dots, n$. This is an explicit correspondence. Send a tally to its length and send an index back to the tally with that many steps. The two composites are identities. In particular, a stage is finite because its elements have been enumerated, not because a finiteness assumption has been attached to it.

The inclusions satisfy

$$j_{n,n} = \text{id}, \quad j_{m,k} \circ j_{n,m} = j_{n,k} \quad (n \leq m \leq k).$$

Both equations say that enlarging a budget does not change a record. Every $x \in \mathbb{N}_\delta$ lies in its own finite stage $F_{L(x)}$. No search for a suitable stage is needed: the record supplies its budget.

This gives more than an increasing list of finite sets. It tells us exactly when calculations or readings defined separately on those sets determine a single reading of the generated orbit.

Definition 10.2 (Agreement across budgets). Let A be a fixed target. A compatible family consists of maps $f_n : F_n \rightarrow A$ such that

$$f_m(j_{n,m}(x)) = f_n(x) \quad (n \leq m, x \in F_n). \quad (10.1)$$

Compatibility is a substantial condition. It says that the answer assigned to an already available record does not change merely because the reader is given a larger permitted budget.

Theorem 10.3 (Finite readings glue uniquely). *For every compatible family (f_n) , exactly one map $f : \mathbb{N}_\delta \rightarrow A$ restricts to f_n on every F_n . It is given explicitly by*

$$f(x) = f_{L(x)}(x).$$

Proof. The expression on the right is defined because $x \in F_{L(x)}$. If $x \in F_n$, then $L(x) \leq n$, and compatibility gives

$$f_n(x) = f_n(j_{L(x),n}(x)) = f_{L(x)}(x) = f(x).$$

Thus f has every required restriction. Conversely, any map g with those restrictions satisfies $g(x) = f_{L(x)}(x)$ by applying its restriction at the particular budget $L(x)$. Hence $g(x) = f(x)$ for every x . $\square$

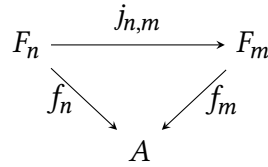


Figure 10.1. A larger budget retains the record. Agreement along the two routes is the hypothesis that makes the single global reading possible.

The proof uses one stage for each input. It does not require an apparatus that stores the whole orbit. Nor does it turn an arbitrary compatible family into an executable rule: the maps f_n were supplied as mathematical data. The theorem establishes precisely the unique assembly of those maps.

10.2 Arithmetic carries its own finite bounds

The length identities from Chapter 3 immediately give finite output budgets:

$$\begin{aligned} S &: F_n \longrightarrow F_{n+1}, \\ + &: F_n \times F_m \longrightarrow F_{n+m}, \\ \cdot &: F_n \times F_m \longrightarrow F_{nm}. \end{aligned} \tag{10.2}$$

For example, $L(x + y) = L(x) + L(y) \leq n + m$. Similarly, $L(xy) = L(x)L(y) \leq nm$. These are bounds for the size of the resulting tally. They do not count the intermediate work of a chosen calculating device.

If the input budgets are enlarged, the output tally is unchanged. For $n \leq n'$ and $m \leq m'$, adding after inclusion gives the same tally as adding first and then including into $F_{n'+m'}$. The multiplication statement is identical with the product budget. The proof is especially short: an inclusion retains its underlying tally, and the operations are the same operations on those tallies on both routes.

There is a useful discipline here. A claim of finite construction should identify what its budget measures. Equations (10.2) measure record length. An execution-time claim or a physical storage claim would require its own construction and bound.

10.3 A record with a decoder

We can make the representation entirely explicit. Prepare a finite bit record $U(x)$ consisting of $L(x)$ repetitions of the bit 0. Define a budgeted decoder D_n as follows: count the length of the supplied bit record r . If that length is at most n , return the tally with that many steps, regarded as an element of F_n . Otherwise return failure.

The decoder uses the record's length. It does not inspect whether its bits are all zero. Preparing only zeros makes one canonical record for each tally; the decoder is allowed to accept other records of the same length too.

Proposition 10.4 (Exact finite recovery). *For every $x \in F_n$,*

$$D_n(U(x)) = x, \quad \text{length}(U(x)) \leq n.$$

Every record longer than n is rejected.

Proof. The length of $U(x)$ is $L(x)$. The budget condition on x therefore selects the successful branch of D_n . Its answer is the tally reconstructed from $L(x)$, which is x by the length round trip. The same length identity proves the preparation bound. A longer input selects the failure branch by definition. $\square$

This construction gives both directions of a representation, together with an exact recovery law. It is stronger than merely counting possible records. It is still a mathematical preparation and reading procedure. Identifying the physical actions that prepare, retain and read it is a further task, addressed in the final part of the book.

10.4 A finite rule can describe an unbounded calculation

We now specify the programs used in this chapter. They are the usual finite expressions for partial recursive functions. A *partial* function may return a natural number on an input, or may remain undefined. Write $c(n) \downarrow v$ when program c returns v on input n . Returning a value is single-valued.

First fix an effective pairing of two natural numbers into one. One explicit choice is

$$\langle a, b \rangle = \begin{cases} b^2 + a, & a < b, \\ a^2 + a + b, & b \leq a. \end{cases} \quad (10.3)$$

It lists pairs by successive square boundaries. Given k , set $s = \lfloor \sqrt{k} \rfloor$ and $t = k - s^2$. If $t < s$, recover (t, s) ; otherwise recover $(s, t - s)$.

To check the round trip, note that $0 \leq t \leq 2s$. In the first case, $t < s$ selects the first branch of (10.3), giving $s^2 + t = k$. In the second, $0 \leq t - s \leq s$ selects the second branch, again giving $s^2 + t = k$. Conversely, if $a < b$, the paired number lies between b^2 and $(b + 1)^2$, and its remainder is $a < b$. If $b \leq a$, it lies between a^2 and $(a + 1)^2$, with remainder $a + b \geq a$. The decoder therefore returns the original pair in both cases.

Definition 10.5 (The program expressions). There are four basic programs: the constant zero function, successor, and the two projections that recover the first and second members of a paired input. From programs f, g form the following further programs.

1. Pairing runs both on the same input and pairs their answers.
2. Composition runs g and then runs f on its answer.
3. Primitive recursion, on input $\langle a, n \rangle$, starts with $f(a)$ and performs n steps. At step y , with previous answer v , it runs $g(\langle a, \langle y, v \rangle \rangle)$.
4. Search, on input $\langle a, m \rangle$, runs f successively on $\langle a, m \rangle, \langle a, m + 1 \rangle, \dots$, returning the first tested position whose answer is zero.

Each program expression is a finite tree built from these eight constructors.

The partial behavior is part of the definition. Composition cannot proceed until its inner program returns. A recursive calculation cannot proceed past an undefined base or step. Search continues past a position only after that call returns a nonzero answer. If it waits forever at an earlier position, it does not skip ahead to a later zero.

Primitive recursion uses a supplied finite step count. Search need not have such a bound. Both have finite descriptions. Consequently, a finite program description is neither a certificate that execution terminates nor a bound on how long execution takes.

Proposition 10.6 (Expressions capture partial recursion). *A partial function obtained from the four basic functions by pairing, composition, primitive recursion and unbounded search has a program expression. Conversely, every program expression defines such a partial recursive function.*

Proof. Induct on a construction of the partial recursive function. A basic function has the corresponding basic expression. At a pairing, composition or primitive-recursion step, apply that expression constructor to the programs given by the induction hypotheses. Their evaluation clauses give exactly the required function.

For a search beginning at zero, pair the input with zero and pass it to the search expression. This supplies its initial position. In the converse direction, induct on the finite program tree. The basic and three binary constructors are their corresponding closure rules. Search from a supplied position is ordinary search on the offset j , applying the test at $m + j$, then adding m to the successful offset. These are compositions and a search of partial recursive functions. Thus both directions include the stated search behavior. $\square$

Here a total function is called *computable* when its value on every input is given by such a partial recursive function. This is the conventional mathematical definition used for the remainder of the chapter. No premise about the capabilities of every physical process enters that definition.

10.5 Storing the program preserves its execution

The expression tree can itself be assigned a natural number. To remove any ambiguity, here is a complete coding. Give the four basic programs the codes 0, 1, 2, 3, in the order listed above. If the codes of f, g are u, v , set

$$\begin{aligned} e(\text{pair}(f, g)) &= 4\langle u, v \rangle + 4, \\ e(\text{comp}(f, g)) &= 4\langle u, v \rangle + 6, \\ e(\text{rec}(f, g)) &= 4\langle u, v \rangle + 5, \\ e(\text{search}(f)) &= 4u + 7. \end{aligned} \tag{10.4}$$

For a number at least four, subtract four and inspect its remainder modulo four. Remainders 0, 2, 1, 3 select pairing, composition, recursion and search, respectively. Divide by four to recover the paired subcodes, or the single subcode for search. Decode the subcodes by the same procedure. Every subcode is smaller than the number being decoded, so the procedure terminates.

Lemma 10.7 (The program coding is reversible). *Decoding $e(c)$ returns the identical finite expression c .*

Proof. Induct on the expression tree. The four basic cases follow from their distinct codes. Each composite code has the remainder selecting its own constructor. Integer division removes that remainder and the initial offset. For a binary constructor, the pairing round trip recovers the two subcodes; for search, it recovers the one subcode directly. Apply the induction hypotheses to those subcodes. The reconstructed constructor and children are exactly the original tree. $\square$

Store c as the tally $P(c)$ with $e(c)$ steps. Recovery first reads its length and then applies the program decoder. This connects the program coding to the orbit constructed in Part I.

Theorem 10.8 (Faithful program storage). *Recovery after storage returns c . In particular, storage is injective, and executing the recovered program has exactly the same partial function as executing c . Both returned answers and undefined behavior are preserved.*

Proof. The length of $P(c)$ is $e(c)$, so the recovery assertion is Lemma 10.7. If two stored tallies are equal, recovering both gives equality of their program expressions. Finally, execution after recovery is execution of the identical expression. Equality of these partial functions preserves undefined inputs as well as defined outputs; no termination assumption is needed. $\square$

For a bit representation, prepare $U(P(c))$. Its length is exactly $e(c)$. The decoder with budget $e(c)$ first recovers $P(c)$ and then recovers c , by Proposition 10.4 and Theorem 10.8. Longer records are refused at the budget check. This unary representation is deliberately simple. Its size bound concerns this specified representation, with no claim that it is the shortest encoding of the program.

10.6 One program for every requested precision

We next apply program storage to rational intervals. The full construction of the continuum from these intervals is the subject of the next chapter. For now we need only the finite data and their consistency conditions.

Definition 10.9 (An interval protocol). An interval protocol p assigns to each natural number n a closed rational interval

$$I_n(p) = [\ell_n(p), u_n(p)]$$

with $\ell_n(p) \leq u_n(p)$, such that

$$I_{n+1}(p) \subseteq I_n(p), \quad u_n(p) - \ell_n(p) \leq \frac{1}{n+1}. \quad (10.5)$$

The index is a precision request. The interval at that index gives the specified rational enclosure, and later requests refine it. The definition requires a family with these properties. It does not yet require a method for producing its endpoints.

Fix the usual effective encoding of rational pairs by natural numbers, with an inverse decoder on valid codes. Rational numbers are finite data: they have reduced signed numerators and positive denominators, as developed in Chapter 4. Write $E_p(n)$ for the code of $(\ell_n(p), u_n(p))$. We use the injectivity of this coding when comparing answers.

Definition 10.10 (A uniform program name). A program c names p if

$$\forall n \quad c(n) \downarrow E_p(n). \quad (10.6)$$

One finite expression must work at every precision. The nesting and width bounds remain properties of the specified protocol. A terminating program that prints arbitrary rational pairs does not acquire those properties merely by terminating.

Lemma 10.11 (A name specifies at most one protocol). *If a program names both p and q , their interval families are equal.*

Proof. For each n , single-valued execution gives $E_p(n) = E_q(n)$. Injectivity of the pair encoding gives equality of both rational endpoints. Thus the intervals agree at every index, which is equality of the protocol families. $\square$

Different programs may name the same family. The lemma asserts the other direction: a fixed program does not leave the answers undetermined.

The constant protocol $[q, q]$ has a concrete name for every rational q . If k is the code of (q, q) , start with the zero program and compose with successor k times. Induction on k shows that this finite expression returns k on every input. The interval conditions hold because its width is zero and every interval is the same.

Theorem 10.12 (The criterion for an executable name). *An interval protocol has a uniform program name if and only if its endpoint code function $n \mapsto E_p(n)$ is computable. The same criterion holds when the name must be stored as a distinction tally and recovered before execution.*

Proof. If c names p , its partial evaluation is the total function returning $E_p(n)$ at every n . Proposition 10.6 therefore gives computability. Conversely, that proposition supplies a finite expression for a computable endpoint function, and equality with that total function is precisely (10.6).

For the storage assertion, store a supplied name as $P(c)$ and use Theorem 10.8. In the reverse direction, recover the program from a purported stored name. Its execution is the required uniform name. Storage has changed the representation without changing the criterion. $\square$

10.7 A finite prefix is a different quantifier

Suppose we ask only for the first N endpoint pairs. These form a finite table

$$T_{p,N} = (E_p(0), E_p(1), \dots, E_p(N-1)).$$

Given this table, a program can return its n th entry when $n < N$, and return zero otherwise. Finite list lookup is primitive recursive: it follows the list until the requested position or its end, with a default answer at the end. Its fixed entries are finite constants. Thus the expression construction of Proposition 10.6 gives a program for the lookup.

Theorem 10.13 (Every supplied finite prefix has a program). *For every interval protocol p and every N , there is a program c such that*

$$\forall n < N \quad c(n) \downarrow E_p(n).$$

The program may depend on the entire supplied finite table.

Proof. Use the lookup program just constructed from $T_{p,N}$. Below N , its entry at n is $E_p(n)$ by the definition of the table. The default branch is irrelevant to the stated range of inputs. $\square$

The premise is access to the table as data. The proof supplies no method for extracting it from an unknown physical quantity. Nor does the theorem supply a procedure that takes N and generates the required table.

For a fixed p , compare the two assertions:

$$\begin{aligned} \forall N \exists c \forall n < N \quad c(n) \downarrow E_p(n), \\ \exists c \forall n \quad c(n) \downarrow E_p(n). \end{aligned}$$

The first permits a different finite table and program at each budget. The second requires one program for all budgets. Chapter 11 will prove that the first does not imply the second for general interval protocols in the classical real display. The proof will use a precise countability obstruction, not a failure to find a clever program.

This is the point at which finite construction needs uniformity. Every individual tally has a finite stage. Every finite program has a tally representation that preserves its execution.

A whole family of rational intervals has such a program name exactly when its endpoint function is computable. Those are three established statements, each with its own object and its own quantifiers.

These program expressions have a second use. The terms of the typed language of Chapter 9 can be compiled into them and stored as numbers by (10.4). Identity proofs compile to the program that returns 0, identity elimination to the program of its base term, elimination from the empty type to a default program fixed by its target type, and **E** in the same way as **R**. A function is then a program code, applied through one fixed universal program that searches for a finite record of the computation and checks each candidate with one fixed test program. Chapter 16 uses this to compare the strongest typed fragment, $\mathcal{E}^+$, with Peano arithmetic: it proves the negative translations of exactly the theorems of Peano arithmetic. One direction of that comparison translates Peano derivations into typed terms. The other runs the typed terms as compiled programs. It needs derivations, by the arithmetic rules, that the test accepts a record of every computation and accepts only genuine records; that chapter shows how the record and its test are built so that the rules can supply them.

Questions to work through

1. In the proof of Theorem 10.3, identify the particular inclusion used for an input $x \in F_n$. Where is compatibility needed, and where is the own-budget construction sufficient?
2. Explain why increasing both input budgets in an addition changes neither its underlying inputs nor its result. What quantity is bounded by $n + m$, and what quantity has not been bounded?
3. Follow storage, bit preparation, budgeted decoding and program recovery for an arbitrary program c . Cite the round-trip identity used at each step. Why does this argument need no assertion that c terminates?
4. A program prints correct endpoint pairs for the first N requests and then prints zeros. Which conclusion of this chapter can it satisfy? Which additional obligation is required for it to be a uniform name?

Chapter 11

The continuum through rational enclosures

A rational interval tells us both an answer and how much remains unresolved. If a length lies between two rational bounds, narrowing those bounds improves the description without pretending that a finite record contains every digit. The question is how a coherent sequence of such records relates to a real number, and what is lost when only that number is retained.

We use the interval protocols of Definition 10.9. Their intervals are nested and have width at most $1/(n+1)$ at request n . The arithmetic of the endpoints is rational arithmetic. A protocol is a mathematical family of these intervals; a uniform executable name is the additional structure studied in Chapter 10.

This chapter compares protocols with the ordinary classical real line. That comparison uses its completeness and Archimedean property, together with classical reasoning. These are resources of the surrounding mathematics in which the comparison is proved. They are not new rules silently inserted into the object languages of Part II.

11.1 Why every later interval constrains every earlier one

Fix a protocol p , and abbreviate its interval by $I_n = [\ell_n, u_n]$. Regard its rational endpoints as real numbers when making the comparison. Nesting says

$$\ell_n \leq \ell_{n+1}, \quad u_{n+1} \leq u_n.$$

Induction on the distance between indices therefore gives

$$a \leq b \implies \ell_a \leq \ell_b \quad \text{and} \quad u_b \leq u_a.$$

The indices in a lower and upper bound need not be the same. For any a, b , let $k = \max(a, b)$. Then

$$\ell_a \leq \ell_k \leq u_k \leq u_b. \tag{11.1}$$

This simple observation is the reason a single value can satisfy every interval. Every upper endpoint bounds all the lower endpoints, even those arising at much later requests.

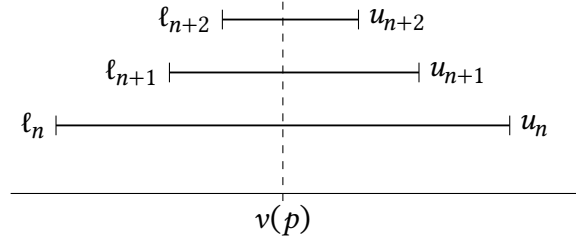


Figure 11.1. Nesting retains earlier constraints. The displayed value lies in every interval; the width condition makes that value unique. The drawing shows the containment pattern, not a prescribed sequence of widths.

11.2 A unique value lies in every enclosure

The set of lower endpoints is nonempty and bounded above by u_0 , using (11.1). Completeness of the classical real line therefore supplies its least upper bound. Define the *displayed value* by

$$v(p) = \sup\{\ell_n : n \in \mathbb{N}\}. \quad (11.2)$$

The word “displayed” identifies the comparison being made: one real number is retained from the more detailed interval family.

Proposition 11.1 (The value satisfies every finite enclosure). *For every n ,*

$$\ell_n \leq v(p) \leq u_n.$$

Proof. Every member of a set is at most its supremum, so $\ell_n \leq v(p)$. Equation (11.1) says that u_n is an upper bound for the entire set of lower endpoints. The least-upper-bound property then gives $v(p) \leq u_n$. $\square$

Existence used nesting and completeness. Uniqueness uses the quantitative width bound. We first isolate the small estimate that will be used twice.

Lemma 11.2 (No positive quantity is below every reciprocal bound). *If a real number a satisfies $0 \leq a \leq 1/(n+1)$ for every natural number n , then $a = 0$.*

Proof. If a were positive, the Archimedean property would give a natural number $n > 1/a$. Hence $n+1 > 1/a$, and taking reciprocals of these positive numbers would give $1/(n+1) < a$, contradicting the supplied bound. Nonnegativity then leaves $a = 0$. $\square$

Theorem 11.3 (Uniqueness of the displayed value). *If y belongs to every interval of p , then $y = v(p)$.*

Proof. Both y and $v(p)$ lie between ℓ_n and u_n . Therefore

$$0 \leq |y - v(p)| \leq u_n - \ell_n \leq \frac{1}{n+1}$$

for every n . Lemma 11.2 gives $|y - v(p)| = 0$, hence equality. $\square$

This result provides a useful way to prove an identity of displayed values. Instead of calculating a supremum directly, show that the proposed answer lies in every interval. The uniqueness theorem then identifies it with the displayed value.

11.3 When two protocols describe the same point

Two different interval families may retain different finite information while describing the same limiting point. Exact equality of the families would be too strong for this comparison.

Definition 11.4 (Observational equality). Protocols p and q are observationally equal, written $p \simeq q$, if their intervals overlap at every matching precision:

$$\forall n \quad \ell_n(p) \leq u_n(q) \quad \text{and} \quad \ell_n(q) \leq u_n(p). \quad (11.3)$$

Each pair of inequalities is a rational comparison. The definition requires all of them. It is a mathematical condition on two families, not a claim that a finite inspection decides the condition in every case.

Theorem 11.5 (Overlap at every precision is exactly equality of value). *For all interval protocols p, q ,*

$$p \simeq q \iff v(p) = v(q).$$

Proof. Suppose first that the intervals overlap at request n . Since each value lies in its own interval, one direction of the difference satisfies

$$\begin{aligned} v(p) - v(q) &\leq u_n(p) - \ell_n(q) \\ &\leq (u_n(p) - \ell_n(p)) + (u_n(q) - \ell_n(q)). \end{aligned}$$

The second inequality uses $\ell_n(p) \leq u_n(q)$. Interchanging p and q gives the same bound for the negative difference. Consequently,

$$|v(p) - v(q)| \leq \frac{2}{n+1}.$$

Apply this bound at request $2m+1$. Its right side is $1/(m+1)$, so Lemma 11.2 implies equality of the values.

Conversely, if the values are equal, their common value belongs to both intervals at every request. Thus $\ell_n(p) \leq v(p) = v(q) \leq u_n(q)$, and the reverse cross inequality follows in the same way. These are precisely the overlap conditions. $\square$

In particular, $\simeq$ is reflexive, symmetric and transitive: each property follows by translating to equality of values and back. Overlap of arbitrary single intervals is not what proves transitivity here. The theorem uses overlap at every precision, nesting and shrinking widths together.

There are therefore two distinct equalities in use. Equality of protocols retains every rational answer. Observational equality retains the displayed point. Chapter 10's uniqueness of a program name concerns the first equality, because the program returns the exact endpoint pair.

11.4 Rationals and arithmetic act on the intervals

A rational q determines the constant protocol $[q, q]$. It contains q at every request and has width zero, so its displayed value is q by Theorem 11.3. Two such constant protocols are observationally equal exactly when their rational values are equal. The rational arithmetic constructed earlier is thus preserved faithfully in this comparison.

For addition, the main issue is the precision budget. Adding intervals adds their widths. Reading both operands at request n would give only the bound $2/(n+1)$. Instead, read both at request $k = 2n+1$ and define

$$I_n(p+q) = [\ell_{2n+1}(p) + \ell_{2n+1}(q), u_{2n+1}(p) + u_{2n+1}(q)]. \quad (11.4)$$

Proposition 11.6 (Addition respects the protocol conditions). *The intervals in (11.4) form an interval protocol, and*

$$v(p+q) = v(p) + v(q).$$

Proof. The lower endpoint does not exceed the upper endpoint because that is true for both operands. As n increases, the index $2n+1$ increases; monotonicity of both lower endpoint sequences and antitonicity of both upper sequences give nesting of the sum intervals.

The width of the sum interval is the sum of the two operand widths. Hence it is at most

$$\frac{1}{2n+2} + \frac{1}{2n+2} = \frac{1}{n+1}.$$

Finally, add the two lower bounds and the two upper bounds for the operand values. They show that $v(p) + v(q)$ lies in every sum interval. Apply Theorem 11.3 to identify its value. $\square$

Negation reverses and negates the endpoints:

$$I_n(-p) = [-u_n(p), -\ell_n(p)]. \quad (11.5)$$

The new interval has the same width. The two nesting inequalities exchange roles under sign reversal, so it is again a protocol. Negating the value bounds shows that $-v(p)$ lies in every new interval, giving

$$v(-p) = -v(p).$$

Define $p - q = p + (-q)$. The two established value identities then give

$$v(p - q) = v(p) - v(q).$$

These constructions work on the rational enclosures themselves. Their agreement with real addition, negation and subtraction is proved afterward. The sum's reindexing is also visible: its finite answers need not be the same interval family as another construction with the same displayed sum. Observational equality is the comparison that retains the common value.

11.5 Every classical real has such a display

We have sent each protocol to a real number. The reverse coverage statement has an explicit interval formula. Fix a classical real r , and let

$$k_n = \lfloor r2^n \rfloor, \quad I_n^r = \left[\frac{k_n}{2^n}, \frac{k_n + 1}{2^n} \right]. \quad (11.6)$$

The endpoints are rational because k_n is an integer. The floor inequalities show that r lies in each interval.

Lemma 11.7 (The next dyadic floor stays in the previous interval). *For every n ,*

$$2k_n \leq k_{n+1} \leq 2k_n + 1.$$

Proof. The definition of floor gives $k_n \leq r2^n < k_n + 1$. Multiply by two:

$$2k_n \leq r2^{n+1} < 2k_n + 2.$$

Since both k_{n+1} and the bounds are integers, taking the floor gives the lower bound $2k_n$ and the upper bound $2k_n + 1$. $\square$

Theorem 11.8 (Coverage of the classical real line). *The family in (11.6) is an interval protocol with displayed value r . Consequently every classical real is the displayed value of an interval protocol.*

Proof. The lower dyadic bound in Lemma 11.7, divided by 2^{n+1} , says that the new lower endpoint is at least the old one. The upper bound, after adding one and dividing by 2^{n+1} , says that the new upper endpoint is at most the old one. This is nesting.

The interval width is 2^{-n} . We have $n + 1 \leq 2^n$ by induction: it is equality at zero, and from $n + 1 \leq 2^n$ and $1 \leq 2^n$ we obtain $n + 2 \leq 2^n + 2^n = 2^{n+1}$. Taking reciprocals proves $2^{-n} \leq 1/(n + 1)$, the required width bound.

The floor inequalities put r in every interval. Uniqueness of the value therefore gives $v(I^n) = r$. $\square$

The formula takes r as a supplied classical real. Evaluating an exact floor at every request is not, by itself, a procedure for extracting these integers from an unknown quantity. The theorem establishes coverage by general interval families. A finite program name still requires the computability condition of Theorem 10.12.

Remark 11.9 (What the floor digits cost). For $0 \leq r < 1$, the integers k_n of (11.6) record the binary digits of r . Here $k_0 = 0$, and by Lemma 11.7 each digit $k_{n+1} - 2k_n$ is 0 or 1: it says which half of the previous interval contains r . The decision between the halves is classical. On the line $\mathbb{R}_\delta$ of Chapter 12, which assumes no classical completeness, the statement that every point of $[0, 1]$ has a binary expansion is equivalent to LLPO given countable choice, and binary expansions for all points already yield LLPO with no choice at all (Section 12.9). The obstacle is a point that may lie on a dyadic boundary: choosing its half requires a decision that no finite stage supplies. If a digit may also be -1 , consecutive intervals overlap and the obstacle disappears. Every point of $[-1, 1]$ has a signed-digit expansion, each digit read off one stage of one sequence naming the point, with no omniscience principle and no choice principle.

Taken together, the results identify exactly what the display retains. It covers the classical real line; it identifies two protocols exactly when they overlap at every precision; and it preserves the rational embedding, addition, negation and subtraction constructed above. None of those assertions requires identifying all interval protocols with finite programs.

11.6 Why finite prefixes do not provide a uniform name

We can now prove the promised distinction between finite data at each precision and one program for every precision. The argument has two steps. There are only countably many program-named values. The classical real line has more than countably many points.

Lemma 11.10 (There are only countably many named values). *The set of displayed values of uniformly program-named interval protocols is countable.*

Proof. A named protocol retains a finite program and the protocol it names. For a fixed program, Lemma 10.11 permits at most one such protocol. Retaining the program is therefore an injection of named protocols into the finite program expressions. Those expressions inject into the natural numbers by Lemma 10.7. Named protocols are consequently countable. Their displayed values form an image of a countable set, so that image is countable too. $\square$

For completeness, the uncountability needed here can be seen directly from Cantor's argument. Binary sequences cannot be listed as $b_0, b_1, b_2, \dots$: the sequence whose n th bit is the opposite of $b_n(n)$ differs from every sequence at its corresponding position.

These sequences inject into the classical real line. For a binary sequence b , with its bits regarded as 0 or 1, use the convergent sum

$$C(b) = \sum_{n=0}^{\infty} b(n) 3^{-n}. \quad (11.7)$$

Convergence follows by comparison with the geometric series. If two sequences first differ at position k , their contributions before k agree. The contribution at k differs by 3^{-k} , while all later contributions together can change the difference in the opposite direction by at most

$$\sum_{n>k} 3^{-n} = \frac{1}{2} 3^{-k}.$$

The first difference therefore cannot be canceled. The map is injective, and a countable real line would make the binary sequences countable as well, contradicting the diagonal argument. This is a classical cardinality argument; the infinite sums and the deduction from distinct sequences to a first differing position are used in that setting.

Theorem 11.11 (The finite-prefix and uniform-name boundary). *In the classical comparison of this chapter, every interval protocol has a program for each supplied finite prefix, but some interval protocol has no single program naming all its intervals. The same failure holds if names are required to be stored in distinction tallies.*

Proof. The first assertion is Theorem 10.13. By Lemma 11.10 and the uncountability just proved, choose a real r outside the set of named values. By Theorem 11.8, it is the value of its dyadic protocol p . If p had a uniform name, its value r would belong to the excluded countable set, a contradiction.

If p had a stored program name, recovering it would give an ordinary uniform name by Theorem 10.8. The same contradiction therefore applies after storage. $\square$

The result rules out a specific quantifier reversal. It does not say that a particular unnamed protocol has been obtained by a finite measurement, or that a physical process has performed a noncomputable operation. It concerns the difference between all families allowed by a mathematical definition and the families produced by finite programs.

There is a further question even for executable families. Producing successively narrower intervals is one task; deciding exact equality or choosing the sign of an arbitrary limiting quantity is another. The next chapter asks what additional logical principles those decisions require. It also takes apart the completeness used in this chapter. On a line built from the ratios of Chapter 4, with no classical real line assumed, limits, least upper bounds and suprema each receive their own exact price.

Questions to work through

1. Prove the cross-endpoint inequality (11.1) when $a \leq b$, then when $b \leq a$. Explain how the maximum-index proof combines the two cases.
2. In Theorem 11.5, identify where overlap is used and where the width bound is used. Why is the request $2m + 1$ convenient?
3. Check both nesting inequalities for the negated protocol. Then explain why proving enclosure of $-v(p)$ avoids a direct calculation with suprema.
4. Trace the contradiction in Theorem 11.11. Which result supplies a protocol for the selected real, and which result shows that a name for this protocol is impossible?

Chapter 12

Exact comparison has a cost

Suppose successive measurements enclose a quantity ever more closely, yet zero remains inside every enclosure received so far. More precision can eventually reveal that the quantity is positive or negative. What would justify a procedure that always decides whether it is exactly zero?

The difficulty is the word “always.” A finite calculation can settle a strict inequality once the separation is larger than its error. An exact zero decision must also settle the case in which no separating witness ever appears. This chapter identifies the additional logical principle required by each of four precise forms of comparison: deciding exact zero, producing a witnessed sign, choosing one of two sides, and turning a proof that a quantity is not zero into a sign. Each result has two directions: the principle supplies the comparison, and the comparison supplies the principle. The chapter then asks the same question of completeness, the property of the line that the classical comparison of Chapter 11 took for granted.

We use a regular dyadic presentation to keep every finite comparison in integer arithmetic. This is a specified presentation with its own error condition. The classical completeness argument of Chapter 11 is not used in the proofs below. Nor do we silently transfer an operation between this presentation and the interval protocols of that chapter. Section 12.8 shows that nothing depends on the dyadic denominators: rounding matches these presentations, point for point, with regular sequences of the ratios of Chapter 4, and every price proved here holds there unchanged.

12.1 Finite stages and exact predicates

A dyadic rational is an integer divided by a power of two. At stage n , write the approximation as

$$x_n = \frac{q_n}{2^n}, \quad q_n \in \mathbb{Z}.$$

The numerator is finite data. A sequence of arbitrary numerators need not describe one coherent quantity, so the presentation includes a regularity condition.

Definition 12.1 (Regular dyadic presentation). A regular dyadic presentation consists of a sequence $(q_n)_{n \geq 0}$ of integers such that, for every m, n ,

$$|q_m 2^n - q_n 2^m|(m+1)(n+1) \leq 2^{m+n}(m+n+2). \quad (12.1)$$

In rational notation, the same condition reads

$$|x_m - x_n| \leq \frac{1}{m+1} + \frac{1}{n+1}. \quad (12.2)$$

The rational notation abbreviates the integer condition: multiply by the positive denominator $2^{m+n}(m+1)(n+1)$ to recover (12.1).

We will write x for the whole presentation. This notation does not discard the numerators or their regularity. In particular, each decision made at a specified stage uses the numerator at that stage.

Definition 12.2 (Five comparisons with zero). For a regular dyadic presentation x , define

$$\begin{aligned} Z(x) : & \quad \forall n, |q_n|(n+1) \leq 2 \cdot 2^n, \\ P(x) : & \quad \exists n, 2 \cdot 2^n < q_n(n+1), \\ N(x) : & \quad \exists n, q_n(n+1) < -2 \cdot 2^n, \\ U(x) : & \quad \forall n, -2 \cdot 2^n \leq q_n(n+1), \\ D(x) : & \quad \forall n, q_n(n+1) \leq 2 \cdot 2^n. \end{aligned}$$

These are, respectively, the zero predicate, positive and negative witness predicates, and nonnegative and nonpositive predicates for this presentation.

In rational notation, $Z(x)$ says that every stage lies within $2/(n+1)$ of zero. The predicate $P(x)$ supplies a stage above this tolerance; $N(x)$ supplies one below its negative. The predicates $U(x)$ and $D(x)$ say that all stages respect the corresponding one-sided bounds. Their letters will keep the full quantified statements visible while we compare their logical strength.

This is the link with the enclosure picture of the preceding chapter. Finite data are tested against a stated error allowance. A positive or negative witness clears that allowance at a particular stage; the zero predicate keeps every stage within a shrinking band around zero. The nonnegative and nonpositive predicates prohibit witnesses on the other side. The regularity condition keeps the stages consistent enough that opposite witnesses cannot both occur, as the next lemma proves. These names refer to the displayed predicates of this presentation; using them does not invoke a map from an interval protocol or a completed classical real.

The factor two is part of these definitions. The theorems concern this regularity condition and these thresholds. They do not claim that an arbitrary choice of tolerances would give the same statements without a separate argument.

Every inequality at a fixed index is decidable by finite integer arithmetic. The quantifiers are where the questions change. To establish $P(x)$ it suffices to produce one index. To establish $Z(x)$ requires a reason that works at every index. Failing to find a witness in the stages already examined supplies neither such a reason nor an exact zero decision.

Lemma 12.3 (Opposite witnesses are incompatible). *A regular dyadic presentation cannot satisfy both $P(x)$ and $N(x)$.*

Proof. Suppose m witnesses $P(x)$ and n witnesses $N(x)$. In rational notation,

$$x_m > \frac{2}{m+1}, \quad x_n < -\frac{2}{n+1}.$$

Subtracting gives

$$x_m - x_n > \frac{2}{m+1} + \frac{2}{n+1}.$$

The right side is strictly greater than $1/(m+1) + 1/(n+1)$, whereas regularity bounds $|x_m - x_n|$ by that smaller quantity. This is a contradiction. All denominators here are positive integers. Clearing them gives the same contradiction entirely in integer arithmetic. $\square$

This lemma will do essential work in the proof of Theorem 12.8. Without regularity, a list could alternate between large positive and negative values, and that argument would fail.

12.2 Questions about a binary sequence

Let $\alpha : \mathbb{N} \rightarrow \{0, 1\}$ be a binary sequence. The value at each specified index is decidable. The principles WLPO, LPO and LLPO of Definition 7.8 in Chapter 7, together with the binary form MP_{bin} of Markov's principle stated there, concern what can be concluded about all its values together.

WLPO decides whether the all-zero assertion holds. Its negative answer does not itself provide an index. LPO supplies an index in its second branch. LLPO has a different premise and conclusion: with at most one event, one parity class is entirely free of events. It need not tell us whether an event occurs in the other class. MP_{bin} decides nothing: it turns a proof that not every value is zero into an index whose value is one.

The binary form of LPO makes the witness distinction of Chapter 6 concrete. These are principles about arbitrary sequences in the stated domain; there is no unmentioned restriction to sequences generated by a selected class of programs.

12.3 Encoding an event as a small jump

The reverse implications need a construction that turns information about a binary sequence into information about a regular dyadic presentation. It must be possible to construct each finite stage without first deciding whether the sequence ever contains a 1.

At stage n , inspect only

$$\alpha(0), \alpha(1), \dots, \alpha(n).$$

If they are all zero, report that no event has yet been found. Otherwise, report the least index $N \leq n$ with $\alpha(N) = 1$. This is a bounded search. Once such a first index has been found, every later search reports the same index: all smaller entries have already been checked.

Choose a sign $\sigma(N) \in \{-1, 1\}$ for each possible first index. Define the jump numerators by

$$q_n = \begin{cases} 0, & \text{if no event has been found through stage } n, \\ \sigma(N)2^{n-N}, & \text{if the first event is at } N \leq n. \end{cases} \quad (12.3)$$

Thus the rational stages are zero before the first event and $\sigma(N)2^{-N}$ from then on. The exponent $n - N$ in the numerator is nonnegative whenever that branch is used. No unbounded search is required to produce q_n .

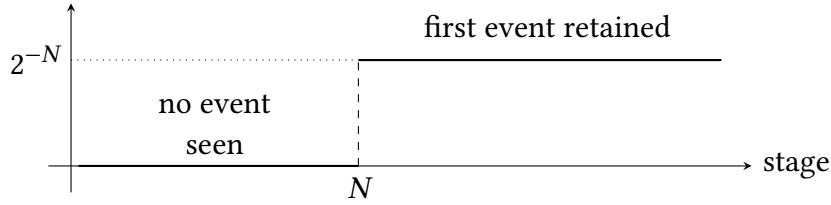


Figure 12.1. A positive jump presentation. The first event fixes the later rational stage. A later first event produces a smaller jump; the drawing shows the construction, with no claim that the vertical scale is uniform across different choices of N .

Lemma 12.4 (The jump presentation is regular). *For every binary sequence and every choice of signs of absolute value one, (12.3) defines a regular dyadic presentation.*

Proof. Compare stages m and n . If neither bounded search has found an event, both stages are zero. If both have found one, they report the same first index, so both rational stages equal $\sigma(N)2^{-N}$. In either case their difference is zero.

In the remaining case, interchange m, n if necessary so that the search at m has found no event and the search at n has found its first event at N . Then $m < N \leq n$, and the absolute difference between the rational stages is 2^{-N} .

For every positive integer N , $N < 2^N$. This follows by induction: $1 < 2$, and $N+1 \leq 2N < 2^{N+1}$ when $N \geq 1$ and $N < 2^N$. Since $m+1 \leq N$, it follows that

$$2^{-N} < \frac{1}{m+1} \leq \frac{1}{m+1} + \frac{1}{n+1}.$$

This is the required regularity bound. The case $N = 0$ cannot occur in this branch, since it requires $m < N$. $\square$

Let $j_{\sigma, \alpha}$ denote this presentation. Its decisive property is an exact relationship between its zero predicate and the original sequence.

Lemma 12.5 (A jump is zero exactly when no event occurs). *For every allowed sign sequence,*

$$Z(j_{\sigma,\alpha}) \iff \forall k, \alpha(k) = 0.$$

Proof. If every entry is zero, every jump numerator is zero, so every inequality in $Z(j_{\sigma,\alpha})$ holds.

Conversely, suppose $Z(j_{\sigma,\alpha})$. Fix an index k . Its binary value is decidable. If $\alpha(k) = 1$, bounded search through k supplies a first event $N \leq k$. Choose the later stage

$$n_0 = 2^{N+1}.$$

This stage is at least N , and its rational absolute value is 2^{-N} . The zero predicate at n_0 would require

$$\frac{1}{2^N} \leq \frac{2}{2^{N+1} + 1}.$$

Multiplying by the positive denominators gives $2^{N+1} + 1 \leq 2^{N+1}$, a contradiction. Therefore $\alpha(k) = 0$. Since k was arbitrary, every entry is zero. $\square$

The displayed stage is a sufficient witness, not a claim about an optimal running time. Its purpose is to show exactly how a nonzero jump eventually exceeds the shrinking zero tolerance.

12.4 Exact zero decisions are equivalent to WLPO

Theorem 12.6 (The logical price of an exact zero decision). *The following statements are equivalent:*

1. *Every regular dyadic presentation x satisfies $Z(x) \vee \neg Z(x)$.*
2. *WLPO holds for every binary sequence.*

Proof. Suppose first that the exact zero decision is available. Given α , construct its positive jump presentation, using $\sigma(N) = 1$ for every N . Apply the decision to it. By Lemma 12.5, the zero branch gives $\forall n, \alpha(n) = 0$. In the other branch, that all-zero assertion would imply the zero predicate, contradicting the decision. Thus it gives $\neg \forall n, \alpha(n) = 0$. This is WLPO.

Conversely, assume WLPO and let x be any regular dyadic presentation. Form a binary sequence of tolerance violations:

$$b(n) = \begin{cases} 1, & 2 \cdot 2^n < |q_n|(n+1), \\ 0, & |q_n|(n+1) \leq 2 \cdot 2^n. \end{cases}$$

Each value is obtained by a decidable integer comparison. The assertion that every $b(n)$ is zero is exactly $Z(x)$. WLPO applied to b therefore gives $Z(x) \vee \neg Z(x)$. $\square$

The second branch is still only a negative assertion. Asking instead for an exhibited positive or negative stage is a stronger request. The next theorem prices the full three-way answer; Theorem 12.11 prices the passage from the negative assertion to a sign on its own.

12.5 Sign trichotomy is equivalent to LPO

Theorem 12.7 (The logical price of a witnessed sign). *The following statements are equivalent:*

1. *Every regular dyadic presentation x satisfies $P(x) \vee Z(x) \vee N(x)$.*
2. *LPO holds for every binary sequence.*

Proof. Assume the three-way comparison. Given α , construct its positive jump presentation. A zero answer gives that every entry of α is zero, by Lemma 12.5. A negative answer is impossible: every numerator of this presentation is nonnegative.

A positive answer supplies a stage n whose numerator exceeds the positive tolerance. The bounded search through n must therefore have found an event; otherwise the numerator would be zero. Its reported index N satisfies $\alpha(N) = 1$. These branches give LPO.

Conversely, assume LPO and use the violation sequence b from the preceding proof. If all entries vanish, $Z(x)$ holds. Otherwise LPO supplies an index n such that

$$2 \cdot 2^n < |q_n|(n + 1).$$

The sign of the integer q_n is decidable. If $q_n \geq 0$, the inequality is a positive witness. If $q_n \leq 0$, it becomes $q_n(n + 1) < -2 \cdot 2^n$, a negative witness. Thus the required three-way comparison follows. $\square$

Both reverse constructions use the same event encoding. What changes is the information returned by the comparison: a negative zero assertion in one case, an actual separating stage in the other.

12.6 One-sided order is equivalent to LLPO

The third comparison asks only for $U(x) \vee D(x)$: every presentation is nonnegative or non-positive in the precise sense of Definition 12.2. It does not require a strict sign witness or a zero decision.

Theorem 12.8 (The logical price of order dichotomy). *The following statements are equivalent:*

1. *Every regular dyadic presentation x satisfies $U(x) \vee D(x)$.*
2. *LLPO holds for every binary sequence with at most one event.*

Proof. Assume the order dichotomy. Let β have at most one entry equal to 1. Construct its jump presentation using the parity sign

$$\sigma(N) = \begin{cases} 1, & N \text{ even,} \\ -1, & N \text{ odd.} \end{cases}$$

If $U(j_{\sigma,\beta})$ holds, every odd entry of β must vanish. Indeed, an event at an odd index N would be its unique event, hence its first. At stage $n_0 = 2^{N+1}$ the rational value would be -2^{-N} . The nonnegative predicate would require

$$-\frac{2}{n_0 + 1} \leq -\frac{1}{2^N},$$

contradicting the same denominator calculation as in Lemma 12.5. The decidable binary value at each odd index is consequently zero. Similarly, $D(j_{\sigma,\beta})$ forces every even entry to vanish. These alternatives give LLPO.

For the converse, assume LLPO and fix a regular presentation x . First make a sequence of signed events by interleaving its finite comparisons:

$$\begin{aligned} e(2k) = 1 &\iff 2 \cdot 2^k < q_k(k+1), \\ e(2k+1) = 1 &\iff q_k(k+1) < -2 \cdot 2^k. \end{aligned}$$

Put zero in each remaining position. There may be many positive events, or many negative events. LLPO requires at most one event, so applying it directly to e would be invalid.

Instead retain only the first event. Define $f(j) = 1$ precisely when the bounded search through $e(0), \dots, e(j)$ reports its first event at j . Put $f(j) = 0$ otherwise. Uniqueness of a first index proves that f has at most one event. LLPO can now be applied to f .

Suppose its even positions are all zero. We show $D(x)$. Fix k and decide the integer inequality defining $D(x)$ at k . If it holds, there is nothing more to prove. If it fails, k is a positive witness, so $e(2k) = 1$. Bounded search through $2k$ supplies a first event at an index j , with $f(j) = 1$. If j is even, this contradicts the LLPO branch. If j is odd, it supplies a negative witness for x , contradicting the positive witness by Lemma 12.3. Hence the inequality must hold. Since k was arbitrary, $D(x)$ follows.

If instead all odd positions of f are zero, the symmetric argument gives $U(x)$. A hypothetical negative witness would force a first event that is either forbidden by that branch or is an incompatible positive witness. Thus $U(x) \vee D(x)$ follows in both cases. $\square$

Two details make this proof work. Taking only the first event supplies the premise required by LLPO. Regularity then excludes a first event of the opposite sign. Neither step can be replaced by an appeal to the visual appearance of a converging sequence.

12.7 The difference between denying and finding

The form of Markov's principle used here is the binary form MP_{bin} of Definition 7.8: for every binary sequence, $\neg(\forall n, \alpha(n) = 0)$ implies $\exists n, \alpha(n) = 1$. We keep to this form rather than silently changing between formulations of Markov's principle.

Proposition 12.9 (LPO separates into a decision and a witness principle). *For the binary principles of Definition 7.8,*

$$\text{LPO} \iff \text{WLPO} \wedge \text{MP}_{\text{bin}}.$$

Moreover, LPO implies LLPO.

Proof. LPO gives either an all-zero assertion or an event. An event contradicts the all-zero assertion, so these alternatives give WLPO. If the all-zero assertion is already denied, LPO's all-zero branch is impossible, leaving an event. This gives MP_{bin} .

Conversely, WLPO decides the all-zero assertion. Keep its positive branch; apply MP_{bin} in its negative branch to obtain an event. The resulting alternatives are exactly LPO.

Finally, suppose a sequence has at most one event. Apply LPO to its subsequence of even entries. If every even entry vanishes, one LLPO branch is already established. Otherwise there is an even event. An odd event would then be a different second event, contrary to the premise. Since each odd entry is binary, every odd entry is zero. This gives the other LLPO branch. $\square$

WLPO's negative branch, applied to a presentation, gives only $\neg Z(x)$. Here is what that leaves undone. Let α be a binary sequence that is known not to be all zero, and let j be its positive jump presentation. Lemma 12.5 gives $\neg Z(j)$. Yet every stage of j before the first event is 0, so a stage above the positive tolerance can be exhibited only once an event of α has been found. For this presentation, turning $\neg Z(j)$ into $P(j)$ is the same task as turning $\neg(\forall n, \alpha(n) = 0)$ into an event. We now show that the passage from a proof that a presentation is not zero to an actual sign costs exactly MP_{bin} in general. First we identify exactly what such a proof says.

Lemma 12.10 (Zero is the absence of both witnesses). *For every regular dyadic presentation x ,*

$$Z(x) \iff \neg P(x) \wedge \neg N(x).$$

Consequently $\neg Z(x)$ holds exactly when $\neg\neg(P(x) \vee N(x))$ holds.

Proof. Suppose $Z(x)$. A positive witness at stage n would give

$$2 \cdot 2^n < q_n(n+1) \leq |q_n|(n+1),$$

contrary to the zero bound $|q_n|(n+1) \leq 2 \cdot 2^n$ at the same stage. A negative witness fails in the same way, with $-q_n$ in place of q_n .

Conversely, suppose that neither witness exists, and fix n . The zero bound at n is a decidable integer comparison. If it failed, the sign of the integer q_n would turn the failure into a positive witness, when $q_n \geq 0$, or a negative witness, when $q_n \leq 0$, exactly as in the proof of Theorem 12.7. Both are excluded, so the bound holds at n . Since n was arbitrary, $Z(x)$ follows.

The equivalence says that $Z(x)$ is $\neg(P(x) \vee N(x))$. Negating both sides gives the second statement. $\square$

A proof that x is not zero is therefore exactly a proof that a sign witness cannot be ruled out. Removing that double negation is the content of the next theorem.

Theorem 12.11 (The logical price of a sign for a nonzero quantity). *The following statements are equivalent:*

1. Every regular dyadic presentation x satisfies $\neg Z(x) \rightarrow P(x) \vee N(x)$.
2. MP_{bin} holds for every binary sequence.

Proof. Assume MP_{bin} , and let x satisfy $\neg Z(x)$. Form the violation sequence b from the proof of Theorem 12.6. The assertion that every $b(n)$ is zero is exactly $Z(x)$, so $\neg Z(x)$ says that b is not all zero. MP_{bin} supplies an index n with $b(n) = 1$. The sign of q_n then gives a positive or a negative witness at n , as in the converse half of Theorem 12.7.

Conversely, assume the first statement, and let α be a binary sequence that is not all zero. Construct its positive jump presentation j , with $\sigma(N) = 1$ for every N . By Lemma 12.5, $Z(j)$ would make every entry of α zero, so $\neg Z(j)$. The first statement gives $P(j) \vee N(j)$. The negative branch is impossible because every numerator of j is nonnegative. A positive witness at stage n has a nonzero numerator, so the bounded search through n has found a first event $N \leq n$, and $\alpha(N) = 1$. This is the conclusion of MP_{bin} . $\square$

The same price appears in a statement that mentions only one sign.

Corollary 12.12 (Stability of positivity). *Every regular dyadic presentation satisfies $\neg\neg P(x) \rightarrow P(x)$ if and only if MP_{bin} holds. The same is true of $\neg\neg N(x) \rightarrow N(x)$.*

Proof. The predicate $P(x)$ says that the binary sequence of positive tests, $p(n) = 1$ exactly when $2 \cdot 2^n < q_n(n+1)$, has an event. If $\neg\neg P(x)$, then p is not all zero, and MP_{bin} supplies the event.

Conversely, let α be a binary sequence that is not all zero, and let j be its positive jump presentation. As in the preceding proof, $\neg Z(j)$, so Lemma 12.10 gives $\neg\neg(P(j) \vee N(j))$. Since $N(j)$ is impossible, this is $\neg\neg P(j)$. Stability gives $P(j)$, and the event of α is read off as before.

For N , replace every numerator q_n by $-q_n$. The regularity condition is unchanged, and the new presentation has a positive witness exactly where the old one had a negative witness. $\square$

The four comparisons now fit together. For a single presentation, a witnessed sign is precisely an exact zero decision together with a sign for the nonzero case.

Proposition 12.13 (Trichotomy is a decision plus a sign). *For every regular dyadic presentation x ,*

$$P(x) \vee Z(x) \vee N(x) \iff (Z(x) \vee \neg Z(x)) \wedge (\neg Z(x) \rightarrow P(x) \vee N(x)).$$

Proof. Suppose the three-way alternative. In the middle case $Z(x)$ holds, and the implication is satisfied because its premise is refuted. In either outer case, Lemma 12.10 gives $\neg Z(x)$, and the required sign is already present. Conversely, decide $Z(x)$. The positive answer is the middle alternative; the negative answer yields a sign through the implication. $\square$

Quantify over all presentations and price the three parts with Theorems 12.7, 12.6 and 12.11. The result is again $\text{LPO} \iff \text{WLPO} \wedge \text{MP}_{\text{bin}}$, this time obtained through the comparisons rather than from the binary sequences directly. Proposition 12.9 is the direct proof. Table 12.1 collects the four comparisons.

Comparison, for every regular dyadic presentation	Exact price	Proved in
$Z(x) \vee \neg Z(x)$	WLPO	Theorem 12.6
$P(x) \vee Z(x) \vee N(x)$	LPO	Theorem 12.7
$U(x) \vee D(x)$	LLPO	Theorem 12.8
$\neg Z(x) \rightarrow P(x) \vee N(x)$	MP_{bin}	Theorem 12.11

Table 12.1. Four comparisons with zero and their exact prices. Each row is an equivalence between the displayed statement, asserted for every presentation, and the named principle for every binary sequence in its domain.

The four equivalences therefore distinguish exact comparison tasks without assuming that they are interchangeable. They also show why approximation alone does not discharge the comparison obligation: the jump construction turns an arbitrary sequence question into that obligation.

An equivalence identifies the logical resource a task supplies and uses. Our claim is the explicit two-way reduction for the stated carrier, thresholds and binary principles. No classical completeness, choice principle, unbounded witness selection or physical measurement postulate is hidden in those reductions. That the four principles differ from one another is a separate, independence question, and Section 7.10 settles it: for each pair, a model of intuitionistic second-order arithmetic satisfies one of the two and refutes the other. In particular Markov's principle is comparable with neither WLPO nor LLPO.

12.8 One line built from ratios

The dyadic denominators kept every stage comparison in integer arithmetic. They are a convenience, not the source of the prices. To see this, build the continuum directly from the ratio records of Chapter 4.

Definition 12.14 (The line built from ratios). A *regular sequence* is a sequence $s = (s_n)_{n \geq 0}$ of ratio records such that

$$|s_m - s_n| \leq \frac{1}{m+1} + \frac{1}{n+1} \quad \text{for all } m, n,$$

each inequality read in integer arithmetic after its positive denominators are cleared. Regular sequences s and t *name the same point*, written $s \approx t$, if for every k there is an N such that $|s_n - t_n| \leq 1/(k+1)$ for every $n \geq N$. The *line* $\mathbb{R}_\delta$ is the set of regular sequences taken up to $\approx$. A ratio record q determines the point $\eta(q)$ of its constant sequence.

The relation $\approx$ is an equivalence relation; for transitivity, use the tolerance $1/(2k+2)$ for both given pairs and add the two bounds. Every regular dyadic presentation, read stage by stage as

the ratio records $(q_n, 2^n)$, is a regular sequence: condition (12.2) is exactly the condition of the definition.

Naming the same point asks for eventual agreement within every tolerance, a condition with three quantifiers. For the point zero it collapses to a single bound at each stage: the zero predicate of this chapter.

Lemma 12.15 (Bishop's zero condition). *A regular sequence s satisfies $s \approx \eta(0)$ if and only if $|s_n| \leq 2/(n+1)$ for every n .*

Proof. If every stage satisfies the bound, take $N = 2k + 1$ for the tolerance $1/(k+1)$: every $n \geq N$ has $|s_n| \leq 2/(n+1) \leq 1/(k+1)$.

Conversely, suppose $s \approx \eta(0)$, and fix m . The bound at m is a decidable comparison, so it suffices to refute its failure. If $|s_m| > 2/(m+1)$, regularity gives, for every n ,

$$|s_n| \geq |s_m| - \frac{1}{m+1} - \frac{1}{n+1} > \frac{1}{m+1} - \frac{1}{n+1}.$$

For $n \geq 2m + 2$ the right side is at least

$$\frac{1}{m+1} - \frac{1}{2m+3} = \frac{m+2}{(m+1)(2m+3)} \geq \frac{1}{2m+3}.$$

So every stage from $2m + 2$ on lies outside the tolerance $1/(2m+3)$, contradicting $s \approx \eta(0)$ at that tolerance. $\square$

A second estimate shows that a clearly positive stage is never lost.

Lemma 12.16 (A positive stage persists). *If a regular sequence satisfies $s_N > 2/(N+1)$, then $s_m > 1/(2N+2)$ for every $m \geq 2N+1$.*

Proof. Since $m+1 \geq 2N+2$, regularity gives

$$s_m \geq s_N - \frac{1}{N+1} - \frac{1}{m+1} > \frac{1}{N+1} - \frac{1}{m+1} \geq \frac{1}{N+1} - \frac{1}{2N+2} = \frac{1}{2N+2}. \quad \square$$

Call a point x *positive*, written $x > 0$, when a regular sequence naming it has a stage with $s_N > 2/(N+1)$. This belongs to the point, not to the sequence. To see it, let $t \approx s$, and take M such that $|s_m - t_m| \leq 1/(4N+4)$ for every $m \geq M$. For every $m \geq \max(M, 8N+7)$, Lemma 12.16 gives

$$t_m > \frac{1}{2N+2} - \frac{1}{4N+4} = \frac{1}{4N+4} \geq \frac{2}{m+1},$$

so t has a positive stage too. Define $x < 0$ in the same way, with a stage below $-2/(N+1)$. Write $x \geq 0$ for $\neg(x < 0)$ and $x \leq 0$ for $\neg(x > 0)$.

For a regular dyadic presentation, read as a regular sequence, these are the predicates of Definition 12.2. Its stages $q_n/2^n$ give $x > 0$ exactly when $P(x)$ and $x < 0$ exactly when $N(x)$: the inequalities are the same, with the denominator 2^n cleared. Each stage comparison is decidable, so $U(x)$, which says that no stage is a negative witness, is $\neg N(x)$, and $D(x)$ is $\neg P(x)$. In rational notation $Z(x)$ says that every stage lies within $2/(n+1)$ of zero, so by Lemma 12.15 it says that the point is $\eta(0)$.

The line contains nothing beyond the dyadic presentations.

Theorem 12.17 (One continuum, two presentations). *For a regular sequence s , let r_n be the integer nearest to $2^n s_{2n+1}$, rounding halves upward. Then (r_n) is a regular dyadic presentation naming the same point as s . Two regular dyadic presentations name the same point exactly when $|q_n - q'_n|(n+1) \leq 2 \cdot 2^n$ for every n . Under this correspondence Z, P, N, U and D become $x = \eta(0)$, $x > 0$, $x < 0$, $x \geq 0$ and $x \leq 0$. No choice principle is used.*

Proof. The rounding error is at most one half, so $|r_n/2^n - s_{2n+1}| \leq 2^{-(n+1)}$, and $2n+2 \leq 2^{n+1}$ makes this at most $1/(2n+2)$. Using regularity of s at the stages $2m+1$ and $2n+1$ for the middle term,

$$\left| \frac{r_m}{2^m} - \frac{r_n}{2^n} \right| \leq \frac{1}{2m+2} + \left(\frac{1}{2m+2} + \frac{1}{2n+2} \right) + \frac{1}{2n+2} = \frac{1}{m+1} + \frac{1}{n+1}.$$

This is exactly (12.2), with nothing to spare. In the same way,

$$\left| \frac{r_n}{2^n} - s_n \right| \leq \frac{1}{2n+2} + \left(\frac{1}{2n+2} + \frac{1}{n+1} \right) = \frac{2}{n+1}.$$

A bound of $2/(n+1)$ at every stage gives agreement within $1/(k+1)$ from stage $2k+1$ on, as in the first half of Lemma 12.15. So the rounded presentation names the point of s .

The same remark proves one direction of the second assertion, whose bound reads $|x_n - x'_n| \leq 2/(n+1)$ in rational notation. Conversely, suppose that x and x' name the same point, and fix n . For any k , choose a stage $p \geq k$ beyond the one from which the two presentations agree within $1/(k+1)$. Regularity of both presentations gives

$$|x_n - x'_n| \leq |x_n - x_p| + |x_p - x'_p| + |x'_p - x'_n| \leq \frac{2}{n+1} + \frac{2}{p+1} + \frac{1}{k+1} \leq \frac{2}{n+1} + \frac{3}{k+1}.$$

If $|x_n - x'_n|$ exceeded $2/(n+1)$, it would exceed it by a positive ratio (u, d) with $u \geq 1$; choosing $k+1 > 3d$ makes $3/(k+1)$ smaller than that excess, a contradiction. The bound at n is a decidable comparison, so it holds. We call this argument the *Archimedean step*: a ratio that is at most $c + C/(k+1)$ for every k is at most c .

The predicates correspond as explained before the theorem. Every step is an explicit integer computation, a bounded estimate or the Archimedean step; no choice principle enters. $\square$

Corollary 12.18 (The four prices on the line). *For the points of $\mathbb{R}_\delta$, each of the following assertions, made for every point x , is equivalent to the principle named: $x = \eta(0) \vee x \neq \eta(0)$, to WLPO; $x > 0 \vee x = \eta(0) \vee x < 0$, to LPO; $x \geq 0 \vee x \leq 0$, to LLPO; $x \neq \eta(0) \rightarrow x > 0 \vee x < 0$, to MP_{bin}; and $\neg\neg(x > 0) \rightarrow x > 0$, to MP_{bin}.*

Proof. Through Theorem 12.17, each assertion for all points is equivalent to the corresponding assertion for all regular dyadic presentations. In one direction, every presentation names a point and the predicates correspond. In the other, an assertion about one given point is a proposition, so we may use one regular sequence naming that point and its rounded presentation; no selection among infinitely many points is needed. Theorems 12.6, 12.7, 12.8 and 12.11 and Corollary 12.12 then give the prices. $\square$

The line also carries the arithmetic one expects, at no logical price. For regular sequences s and t , put

$$(s+t)_n = s_{2n+1} + t_{2n+1}, \quad (-s)_n = -s_n, \quad (st)_n = s_j t_j \quad \text{with } j = 2K(n+1) - 1,$$

where K is the larger of $\lfloor |s_0| \rfloor + 3$ and $\lfloor |t_0| \rfloor + 3$. Regularity keeps every stage within 2 of the first, so $|s_n| < K$ and $|t_n| < K$ for every n . Write $x < y$ for $y - x > 0$ and $x \leq y$ for $\neg(y < x)$.

Proposition 12.19 (The ordered field structure is free). *These operations give regular sequences and respect $\approx$. They make $\mathbb{R}_\delta$ a commutative ring into which η embeds the ratios, preserving and reflecting their order; for records with positive denominators, $(u, d) < (v, e)$ means $ue < vd$. Moreover:*

1. $<$ is irreflexive and transitive, and it is cotransitive: if $x < y$, then $x < z$ or $z < y$ for every z . Adding a point preserves $<$, and multiplying by a positive point preserves $<$;
2. $\leq$ is reflexive and transitive, and $x \leq y \leq x$ implies $x = y$;
3. x has a multiplicative inverse exactly when $x < 0$ or $x > 0$;
4. every point lies below $\eta(n)$ for some natural number n ;
5. every two points have a least upper bound $\max(x, y)$ and a greatest lower bound $\min(x, y)$, and every $x \geq 0$ has exactly one square root $y \geq 0$.

None of these statements uses an omniscience principle or a choice principle.

The proofs are stage estimates of the kind already used; here are the ones that carry the weight. The sum is regular because each of its stages adds two stages read at twice the precision. The product is regular because $|s_i t_i - s_j t_j| \leq K(|s_i - s_j| + |t_i - t_j|)$, and at the stages $i + 1 = 2K(m + 1)$ and $j + 1 = 2K(n + 1)$ the right side is at most $1/(m + 1) + 1/(n + 1)$. Lemma 12.16 makes positivity closed under addition and multiplication. It also gives an explicit inverse for a point apart from zero. From a stage N with $|s_N| > 2/(N + 1)$, every stage $m \geq 2N + 1$ has $|s_m| > 1/(2N + 2)$; between such stages $|1/a - 1/b| \leq 4(N + 1)^2 |a - b|$, so the reciprocals of the stages $4(N + 1)^2(n + 1) - 1$ form a regular sequence. Cotransitivity rests on one decidable comparison. If a sum $a + b$ has the positive stage N , then $a_{2N+1} + b_{2N+1} > 2/(N + 1)$, so one of the two stages exceeds $1/(N + 1) = 2/((2N + 1) + 1)$, which is a positive stage of a or of b . Applied to $(y - z) + (z - x)$, which names the point $y - x$, this gives $z < y$ or $x < z$. The maximum is computed stagewise. These are the constructions of Bishop's analysis [6], carried out here on ratio records. The square root is found by trisection; each decision is computed from one sequence naming the point, so neither a choice principle nor an omniscience principle enters.

With two points the prices of Corollary 12.18 reappear, together with the field and lattice statements that name a side. Table 12.2 lists them.

Two rows deserve a word. Deciding $x < y$ is deciding $x \leq y$ together with the stability $\neg(y \leq x) \rightarrow x < y$, the same split as $\text{LPO} = \text{WLPO} + \text{MP}_{\text{bin}}$. And the classical field axiom, that a nonzero element is invertible, holds exactly under MP_{bin} . Without it the line is a field in the sense of item 3 of Proposition 12.19: what has an inverse is what is apart from zero.

Remark 12.20 (Reasoning by cases costs nothing for stable conclusions). Every priced row has a conclusion that names a side. A conclusion that names no side can be reached by cases for free. Any two points satisfy $\neg\neg(x \leq y \vee y \leq x)$: if neither held, we would have $\neg\neg(y < x)$ and $\neg\neg(x < y)$, hence $\neg\neg(x < x)$, contrary to irreflexivity. An inequality $x \leq y$ is itself a

Assertion, for all points x, y of $\mathbb{R}_\delta$	Exact price
$x < y \vee \neg(x < y)$	LPO
$x < y \vee x = y \vee y < x$	LPO
$x \leq y \vee \neg(x \leq y)$; also $x = y \vee x \neq y$	WLPO
$x \leq y \vee y \leq x$	LLPO
$\max(x, y) = x \vee \max(x, y) = y$	LLPO
$xy = \eta(0) \rightarrow x = \eta(0) \vee y = \eta(0)$	LLPO
$\neg(y \leq x) \rightarrow x < y$; also $x \neq y \rightarrow x < y \vee y < x$	MP_{bin}
$x \neq \eta(0) \rightarrow x$ has a multiplicative inverse	MP_{bin}
$\neg\neg(x \leq y) \rightarrow x \leq y$	none

Table 12.2. Order and algebra on the line. Each priced row is equivalent to the named principle for binary sequences, in both directions, with no choice principle; the last row is proved outright.

negation, so it is stable under double negation, and an equation is stable because it follows from two such inequalities. Hence every equation or $\leq$ -inequality proved by cases on the order of finitely many points holds outright. Identities such as $\min(x, y) \max(x, y) = xy$ and $|x||x| = xx$ are proved in this way. Allowing the same argument for an arbitrary conclusion costs exactly LLPO.

12.9 What completeness costs

Chapter 11 used completeness of the classical line as one resource: a bounded set of lower endpoints had a supremum, and the displayed value was defined from it. On $\mathbb{R}_\delta$ that resource comes apart. Some forms of completeness cost nothing. One, that every located cut is the cut of a point, costs only countable choice, applied once to select an approximating pair of ratios for each precision. The other forms that need nothing beyond countable choice follow from that one with no further choice, and the limit of a Cauchy sequence needs only a choice of representatives. Each of the others costs exactly one principle: LPO, WLPO or LLPO, or, for arbitrary sets of points, weak excluded middle or excluded middle.

A principle of a different kind enters here.

Definition 12.21 (Countable choice). *Countable choice*, AC_ω , states: if for every natural number n there is some x with $R(n, x)$, then there is a sequence f with $R(n, f(n))$ for every n .

Countable choice selects witnesses; it decides nothing. The next theorem makes this exact.

Theorem 12.22 (Countable choice decides nothing). *Consider intuitionistic arithmetic with variables for sequences, induction for every formula, and countable choice both for numbers and for sequences. In that theory none of LPO, WLPO, LLPO and MP_{bin} is provable, and none is refutable. More generally, if it proves $\forall x(A(x) \vee \neg A(x))$ for a property A built from equations by conjunction, implication and universal quantification over numbers, then A is decided by a computer program. With MP_{bin} added, none of WLPO, LLPO and LPO becomes provable.*

Outline. Read every formula as a task for programs, in the way of realizability [4]: a program realizes $\forall x \exists y \varphi$ when on each x it returns a y together with a realizer of $\varphi(x, y)$, and a

disjunction is realized by a program that names a side. For a property built from equations by conjunction, implication and universal quantification over numbers, being realized is the same as being true, so a realizer of $\forall x(A(x) \vee \neg A(x))$ is a program that names, for each x , the side that holds. Every rule of the theory preserves realizability, and the choice schemas are realized by programs that only rearrange what their premise supplies: the premise already returns a witness for each n , and the chosen sequence is that program. A realizer of LPO or WLPO would decide whether a program halts, which no program does [8]. A realizer of LLPO would separate the programs that return 0 on their own code from those that return 1, and no program does that either: a program returning 1 on the first kind and 0 on the second, run on its own code, would contradict itself. So none of the three is provable. Markov's principle is realized in this reading, by a search that runs until it meets a 1, so the same argument proves the last sentence of the theorem. To show that MP_{bin} itself is not provable, one uses the variant called modified realizability: realizers are typed programs, and a realizer of an implication must return an output on every input of its type, not only on realizers of the premise. The choice schemas are realized there by the same rearranging programs. Apply a realizer of MP_{bin} to the sequence recording whether a program has halted by each step, and to any input for the premise that not every value is 0. It returns a number m . When the program halts, that premise is true, every input realizes it, and so the program has halted by step m . Running the program for m steps would then decide halting, so MP_{bin} is not realized. None of the four is refutable, because each holds in the classical model, which also satisfies countable choice. $\square$

Below countable choice appears only as a stated hypothesis, and each result says where it is used.

Distances are order statements. Say that x and y are *within* $1/(k+1)$ when $x \leq y + \eta(1/(k+1))$ and $y \leq x + \eta(1/(k+1))$. For sequences s, t naming the points, this is the stagewise bound $|s_n - t_n| \leq 1/(k+1) + 2/(n+1)$ for every n . A sequence of points (x_n) *converges* to L if for every k there is an N such that x_m is within $1/(k+1)$ of L for all $m \geq N$. It is *Cauchy* if for every k there is an N such that x_m and x_n are within $1/(k+1)$ for all $m, n \geq N$. Points within every $1/(k+1)$ of each other are equal: the Archimedean step leaves the bound $2/(n+1)$ at every stage, and that bound gives $\approx$ as in the first half of Lemma 12.15.

Where countable choice enters

A sequence of points is a sequence of classes. To calculate with it we want stages, so we want one regular sequence from each class. Producing infinitely many such representatives at once is an application of AC_ω . Once they are supplied, the limit is explicit.

Theorem 12.23 (The diagonal limit). *Let $s^{(0)}, s^{(1)}, \dots$ be regular sequences whose points are within $1/(k+1)$ of one another whenever both indices are at least k . The diagonal $d_n = s_{2n+1}^{(2n+1)}$ is a regular sequence, and the point of $s^{(m)}$ is within $1/(k+1)$ of the point of d whenever $k \leq m$. Assuming AC_ω , every Cauchy sequence of points of $\mathbb{R}_\delta$ converges, and its limit is unique.*

Proof. Write $a = 2m + 1$ and $b = 2n + 1$. Compare $d_m = s_a^{(a)}$ with $d_n = s_b^{(b)}$ through an arbitrary stage p :

$$|d_m - d_n| \leq |s_a^{(a)} - s_p^{(a)}| + |s_p^{(a)} - s_p^{(b)}| + |s_p^{(b)} - s_b^{(b)}|.$$

Regularity bounds the outer terms by $1/(a+1) + 1/(p+1)$ and $1/(b+1) + 1/(p+1)$. The hypothesis at $c = \min(a, b)$ bounds the middle term by $1/(c+1) + 2/(p+1)$, and $1/(c+1) \leq 1/(a+1) + 1/(b+1)$. The total is

$$\frac{2}{a+1} + \frac{2}{b+1} + \frac{4}{p+1} = \frac{1}{m+1} + \frac{1}{n+1} + \frac{4}{p+1}.$$

Since p is arbitrary, the Archimedean step removes the last term, and this is the regularity of d .

For the second assertion, let $k \leq m$, put $a = 2n + 1$, and compare $s_n^{(m)}$ with $d_n = s_a^{(a)}$ through a stage p . Regularity of $s^{(m)}$ and of $s^{(a)}$ contributes $1/(n+1) + 1/(p+1)$ and $1/(p+1) + 1/(a+1)$. Neither m nor a is below $c = \min(k, a)$, so the hypothesis at c bounds $|s_p^{(m)} - s_p^{(a)}|$ by $1/(c+1) + 2/(p+1)$, and $1/(c+1) \leq 1/(k+1) + 1/(a+1)$. Since $2/(a+1) = 1/(n+1)$, the total is $1/(k+1) + 2/(n+1) + 4/(p+1)$, and the Archimedean step leaves the stagewise form of “within $1/(k+1)$ ”.

Now let (x_n) be a Cauchy sequence of points. For each k some index N satisfies the Cauchy condition at every $j \leq k$: take the largest of the finitely many indices that the condition supplies for $j = 0, \dots, k$. The point x_N is named by some regular sequence. Countable choice, applied once, supplies for each k a regular sequence $s^{(k)}$ naming x_N for some such N ; the index itself is never selected. For $m, m' \geq k$ the indices behind $s^{(m)}$ and $s^{(m')}$ both satisfy the condition at k , and the condition at the smaller of them covers both, so the points of $s^{(m)}$ and $s^{(m')}$ are within $1/(k+1)$. These sequences therefore satisfy the hypothesis of the first part, so their diagonal names a point L , and the point of $s^{(k)}$ is within $1/(k+1)$ of L for every k . Given k , let N be an index behind $s^{(2k+1)}$. For $m \geq N$, the point x_m is within $1/(2k+2)$ of x_N , which is within $1/(2k+2)$ of L ; adding the two order statements, x_m is within $1/(k+1)$ of L . Two limits are within every $1/(k+1)$ of each other, hence equal. $\square$

The proof uses countable choice once, to pick representatives; the limit itself is computed from the data. A second description of the continuum avoids even that step. A *located cut* is a pair $(\mathcal{L}, \mathcal{U})$ of sets of ratios, both inhabited, with $\mathcal{L}$ closed downward and without a greatest element, $\mathcal{U}$ closed upward and without a least element, the two disjoint, and *located*: whenever $q < r$, either $q \in \mathcal{L}$ or $r \in \mathcal{U}$. Every point x has the located cut $\{q : \eta(q) < x\}$, $\{r : x < \eta(r)\}$, and different points have different cuts. Call a sequence of located cuts *Cauchy* when for every k there is an N such that, whenever $m, n \geq N$, every member of the lower set of the m th cut lies less than $1/(k+1)$ above every member of the upper set of the n th. A *regular* sequence of cuts, in which every member of the lower set of the m th cut lies less than $1/(m+1) + 1/(n+1)$ above every member of the upper set of the n th, is one example. Every Cauchy sequence of located cuts has a unique limit cut, one that the cuts approach within every $1/(k+1)$ from some index on, in the sense of the Cauchy condition with the limit cut in place of either of the two cuts, and it is read off from the sets alone, with no choice principle at all: a ratio q lies in its lower set when, for some k , $q + 1/(k+1)$ lies in the lower set of every cut from some index on, and r lies in its upper set when, for some k , $r - 1/(k+1)$ lies in the upper set of every cut from some index on. Showing that this limit is located uses one index from the Cauchy condition, inside the proof of a disjunction, so that index is never selected. In particular the cuts of the points of any Cauchy sequence of points have a unique limit cut, with no choice; countable choice is used only to make that cut a point, and Theorem 12.27

shows that excluded middle cannot replace it. A cut is the cut of a point exactly when it has an approximation sequence: a pair $q_n \in \mathcal{L}$, $r_n \in \mathcal{U}$ with $r_n - q_n < 1/(n+1)$ for every n . Each single pair can be found without choice; assembling all of them into one sequence is precisely an application of AC_ω , at pairs of ratios. The same rung has an order form. Call a set of points *order-located* if, whenever $q < r$, either $\eta(r)$ is an upper bound of the set or some member exceeds $\eta(q)$. Every inhabited, order-located set that is bounded above has a least upper bound exactly when every located cut is a point. With countable choice the two descriptions of the continuum coincide, and the price of identifying them is that single selection.

The same selection settles a question asked of every stage of the number tower in Chapter 4: is the line the unique solution of its task? Its task is to receive exactly one map, preserving the arithmetic and the order, from every Archimedean ordered field: the structure that the next theorem defines by laws that Proposition 12.19 proves for the line.

Theorem 12.24 (The line is determined by its task). *Call a commutative ring with a strict order $<$ an Archimedean ordered field when $0 < 1$, the order is irreflexive, transitive and cotransitive, two elements neither of which lies below the other are equal, adding an element preserves $<$, a product of positive elements is positive, every positive element has an inverse, and every element lies below some natural number. Then $\mathbb{R}_\delta$ is an Archimedean ordered field, and:*

1. *from every Archimedean ordered field there is at most one map into $\mathbb{R}_\delta$ that preserves 0, 1, addition, multiplication and $<$, and the only such map from $\mathbb{R}_\delta$ to itself is the identity;*
2. *every Archimedean ordered field has such a map into $\mathbb{R}_\delta$ exactly when every located cut is the cut of a point.*

Neither assertion uses a choice principle. With countable choice, therefore, every Archimedean ordered field maps into $\mathbb{R}_\delta$ in exactly one such way.

Outline. Proposition 12.19 shows that $\mathbb{R}_\delta$ is an Archimedean ordered field. Let F be one, and let f be such a map from F . The ratios have images in F , with their order, because the positive natural numbers are positive in F and therefore invertible; f sends the image of q to $\eta(q)$, since it preserves sums, products and hence inverses. If $q < x$ in F , then $\eta(q) < f(x)$. Conversely, if $\eta(q) < f(x)$, take a ratio q' with $\eta(q) < \eta(q') < f(x)$; cotransitivity in F gives $q < x$ or $x < q'$, and the second would give $f(x) < \eta(q')$. So the lower set of the cut of $f(x)$ is $\{q : q < x\}$, and in the same way its upper set is $\{r : x < r\}$. A point is determined by its cut, so x determines $f(x)$. This is the first assertion; the identity is one such map from $\mathbb{R}_\delta$ to itself.

For the second, suppose every located cut is the cut of a point. In F the sets $\{q : q < x\}$ and $\{r : x < r\}$ form a located cut: transitivity and irreflexivity make them closed in the right directions and disjoint, cotransitivity locates them, and the Archimedean condition, with the inverses of positive elements, makes both sets inhabited and leaves the lower set without a greatest element and the upper set without a least one. Send x to the point with that cut; sums, products and the order carry over, because each can be read from the cuts. Conversely, suppose every Archimedean ordered field has such a map. The located cuts, with the arithmetic and order read from their sets, form an Archimedean ordered field, and the argument of the first paragraph, with cuts in place of points, shows that the only such map from the cuts to themselves is the identity. The map from the cuts into $\mathbb{R}_\delta$, followed by the

passage from a point to its cut, is such a map, so every cut is the cut of its image. No step selects anything. With countable choice every located cut is the cut of a point, as shown above, and the last sentence of the theorem follows. $\square$

With countable choice the line is therefore determined by this task up to exactly one isomorphism. If an Archimedean ordered field T also receives exactly one such map from every Archimedean ordered field, the maps $\mathbb{R}_\delta \rightarrow T$ and $T \rightarrow \mathbb{R}_\delta$ compose to such maps of $\mathbb{R}_\delta$ and of T into themselves, and the first assertion, with the same uniqueness for T , makes both composites identities. No other isomorphism exists, since T receives only one such map from $\mathbb{R}_\delta$.

Table 12.3 collects the prices of the familiar forms of completeness. Its terms have their usual meanings, read in the order of $\mathbb{R}_\delta$. A *least upper bound* of a set of points is an upper bound lying below every upper bound. In the table, f is a function on the points of $\mathbb{R}_\delta$, *uniformly continuous on* $[0, 1]$ when for every k there is a j such that points of $[0, 1]$ within $1/(j+1)$ have values within $1/(k+1)$; a *supremum* of f is an upper bound of its values on $[0, 1]$ such that, for every k , some value lies within $1/(k+1)$ of it. Write $c_n = \sum_{i < n} e_i 2^{-(i+1)}$ for the n th partial sum of a digit sequence (e_i) . A *binary expansion* of x has digits in $\{0, 1\}$ and $c_n \leq x \leq c_n + 2^{-n}$ for every n ; a *signed-digit expansion* has digits in $\{-1, 0, 1\}$ and $c_n - 2^{-n} \leq x \leq c_n + 2^{-n}$ for every n .

Two reversals show how the prices arise. Both use only the order properties of Proposition 12.19.

Theorem 12.25 (Monotone convergence gives LPO). *If every bounded nondecreasing sequence of points of $\mathbb{R}_\delta$ converges, then LPO holds. The proof uses no choice principle.*

Proof. Let α be a binary sequence. Put $t_n = \eta(1)$ if $\alpha(k) = 1$ for some $k \leq n$, and $t_n = \eta(0)$ otherwise. Each term is fixed by a bounded search, so the sequence is given outright. It is nondecreasing and bounded above by $\eta(1)$. By hypothesis it has a limit L ; take N such that t_m is within $1/4$ of L for every $m \geq N$.

The points $\eta(0)$ and $\eta(1)$ cannot both be within $1/4$ of L . That would give

$$\eta(1) \leq L + \eta(1/4) \leq \eta(1/4) + \eta(1/4) = \eta(1/2),$$

contrary to $\eta(1/2) < \eta(1)$.

Search $\alpha(0), \dots, \alpha(N)$. If an event appears, the second alternative of LPO holds. Otherwise $t_N = \eta(0)$. Fix n . If $\alpha(n) = 1$, then $t_{N+n} = \eta(1)$, and t_N and t_{N+n} would both be within $1/4$ of L , which was just excluded. Since $\alpha(n)$ is binary, $\alpha(n) = 0$. This is the first alternative of LPO. $\square$

Assertion on $\mathbb{R}_\delta$	Price	Countable choice
The diagonal of a Cauchy family of regular sequences is its limit (Theorem 12.23)	none	not used
Every Cauchy sequence of points converges	none	used, to select representatives
Every Cauchy sequence of located cuts has a limit cut	none	not used
Every located cut is the cut of a point	none	used once, at pairs of ratios
Every bounded nondecreasing sequence converges	LPO	in the converse
Every bounded nondecreasing sequence, given by regular sequences, is Cauchy	LPO	not used
Every bounded sequence has a convergent subsequence; every bounded sequence has a cluster point; every sequence has a monotone subsequence	LPO	in the converse
Every sequence bounded above has a least upper bound	WLPO	in the converse
... one that every smaller ratio is exceeded by some term	LPO	in the converse
Every finite inhabited set has a least upper bound	none	not used
Every inhabited set of points bounded above has a least upper bound	WEM	in the converse
... one that every smaller ratio is exceeded by some member	EM	in the converse
A uniformly continuous f on $[0, 1]$ with $f(0) \leq 0 \leq f(1)$ has a root	LLPO	in the converse
such an f has, for each k , a point with $ f(x) \leq 1/(k+1)$	none	not used
such an f , strictly increasing, has a root	none	used
A uniformly continuous f on $[0, 1]$ attains a maximum	LLPO	in the converse
such an f has a supremum	none	used
Every point of $[0, 1]$ has a binary expansion	LLPO	in the converse
Every point of $[-1, 1]$ has a signed-digit expansion	none	not used

Table 12.3. Completeness on the line built from ratios. “Price” is the principle equivalent to the assertion; “none” means that no omniscience principle is needed, so the assertion is proved outright, from countable choice where the last column says it is used. WEM is the weak excluded middle, $\neg P \vee \neg\neg P$ for every proposition P , and EM is excluded middle, $P \vee \neg P$ for every proposition P . In every priced row the passage from the assertion to the principle uses no choice principle; “in the converse” means that countable choice is used in the passage from the principle back to the assertion.

Theorem 12.26 (A least upper bound gives WLPO). *If every bounded sequence of points of $\mathbb{R}_\delta$ has a least upper bound, then WLPO holds. The proof uses no choice principle.*

Proof. Let α be a binary sequence, and let $b_n = \eta(\alpha(n))$, reading each bit as the ratio 0 or 1. The sequence is bounded, below by $\eta(0)$ and above by $\eta(1)$; let L be a least upper bound. Since $\eta(0) < \eta(1)$, cotransitivity gives $\eta(0) < L$ or $L < \eta(1)$.

If $L < \eta(1)$, no entry of α is 1: an entry $\alpha(n) = 1$ would give $\eta(1) = b_n \leq L < \eta(1)$, and $u \leq v < w$ implies $u < w$ by cotransitivity. Every entry is therefore 0.

If $\eta(0) < L$, then α is not all zero. Otherwise $\eta(0)$ would be an upper bound, and leastness would give $L \leq \eta(0)$, which denies $\eta(0) < L$.

These are the two alternatives of WLPO. □

The proof never decides whether L is $\eta(0)$ or $\eta(1)$; cotransitivity only places L away from one of them. If the least upper bound is also approached by the terms, the case $\eta(0) < L$ yields a term above $\eta(0)$, that is, an event, and the same argument gives LPO.

The passages back that produce a supremum share one pattern. The principle decides, for each ratio q , a question about the terms: whether some term exceeds $\eta(q)$, for LPO, or whether every term stays at most $\eta(q)$, for WLPO. Those decisions make the candidate supremum a located cut, and countable choice turns the cut into a point; where the terms are given only as points, it also supplies their representatives. In the rows priced “none” that use a cut, the cut is located with no decision at all. The root of a strictly increasing f , for example, is located by the signs of f at ratios, so only the passage from cut to point remains. The LLPO rows go back differently: they bisect, and countable choice collects the decision made at each dyadic step.

Theorem 12.27 (Excluded middle does not replace countable choice). *Consider arithmetic with variables for sequences, induction for every formula, excluded middle for every formula, and countable choice for the relations whose witnesses a search can find: a decidable condition, or a decidable condition under one existential quantifier. Write each set of ratios, sequence of points and function on the points as a formula. Then none of the following is provable in that theory: every Cauchy sequence of points converges; every located cut is the cut of a point; every uniformly continuous, strictly increasing f on $[0, 1]$ with $f(0) \leq 0 \leq f(1)$ has a root; every uniformly continuous f on $[0, 1]$ has a supremum. Each of the four follows from countable choice at pairs of ratios.*

Outline. Interpret the theory with numbers ranging over the natural numbers and sequences over the computable sequences. Excluded middle holds because the interpretation is classical, and the restricted choice holds because a search finds a witness for each n and the sequence of witnesses is computable, so every theorem is true in this model. Let $c = \sum_{e \in K} 4^{-(e+1)}$, where K is the set of programs that halt on input 0. No computable regular sequence names c : its base-4 digits would decide halting. The rational points c_s , where c_s adds the terms of c for the programs that halt within s steps, form a Cauchy sequence of points with no limit in the model. The cut of c is located there and is the cut of no point. Functions that are affine with rational coefficients on pieces read off the digits of c are uniformly continuous and take computable points to computable points; one of them is strictly increasing with c as its only root, and another has c as its supremum on $[0, 1]$. In the same model each assertion holds at

computable data, for example at the cut of $\frac{1}{2}$, so the failures come from c . For the last sentence, countable choice at pairs of ratios makes every located cut the cut of a point, as shown above, and the other three assertions follow from that one with no further choice, because the limit, the root and the supremum are each located as a cut first. $\square$

In a classical theory, countable choice for numbers is the same as the principle that a relation with exactly one witness for each n defines a sequence, so the theorem shows that one of the two must be added. Over an intuitionistic base that already contains the second principle, whether these four assertions need countable choice is not settled here.

The same prices appear away from the line. Weak König's lemma for decidable binary trees is equivalent to LLPO, and the infinite pigeonhole principle, for any finite number of colours, and Ramsey's theorem for pairs with two colours are each equivalent to LPO; in each case the passage to the principle uses no choice, and countable choice is used only on the way back. Equivalences of this kind are familiar from constructive analysis [6, 4]. What is particular here is the carrier: every row is proved on the line built from the ratios of Chapter 4, and every passage from a theorem to its principle is made without choice.

Finally, the table repeats the split of LPO. A nondecreasing sequence that converges converges to its least upper bound, with no principle. That every least upper bound of a sequence is approached by its terms is equivalent to MP_{bin} ; the passage to MP_{bin} uses no choice, and the passage back uses countable choice. So, with countable choice, monotone convergence is the least-upper-bound principle together with MP_{bin} , just as LPO is WLPO together with MP_{bin} .

12.10 Exercises

1. Starting from (12.2), clear the positive denominators and recover (12.1). Identify where positivity is needed to preserve the direction of the inequality.
2. In the proof of Lemma 12.5, explain why the later stage 2^{N+1} is available once a finite search has found the first event N . Why is deciding whether any event exists unnecessary for constructing each jump stage?
3. Reconstruct both directions of Theorem 12.6, then identify the exact step that requires LPO instead of WLPO in Theorem 12.7. Use Proposition 12.13 and Theorem 12.11 to show that this step, taken alone, costs exactly MP_{bin} .
4. Explain why the interleaved sequence e in Theorem 12.8 need not satisfy LLPO's premise. Prove that retaining only its first event supplies that premise. Trace the two uses of parity in the converse proof.
5. Prove the converse half of Lemma 12.10 in integer arithmetic. Where is decidability of a single stage used, and why does the proof not need to decide $P(x)$?
6. Derive the statement of Corollary 12.12 for N from the statement for P . Check that replacing each q_n by $-q_n$ preserves (12.1).
7. In Theorem 12.25, show that the test sequence is nondecreasing and bounded. The proof uses one limit point and one index N . Explain why using them requires no choice principle.

8. The proof of Theorem 12.26 never decides whether L equals $\eta(0)$ or $\eta(1)$. Explain why cotransitivity is enough. Then assume in addition that every ratio below L is exceeded by some term, and show that the same test sequence yields LPO.

Part IV

Mathematics and finite physical records

Chapter 13

What a finite record can establish

An instrument returns one of two marks. The experimenter wants to know which of three preparations was supplied. If that mark is the whole available record, a reader cannot always give the right answer: two preparations must leave the same mark, and the reader then receives the same information in two cases requiring different answers.

The conclusion changes if the question changes. Perhaps the experimenter only needs to separate one preparation from the other two. Or perhaps the apparatus can perform another test. A limit on recognition must therefore name both the question and the record. Counting possible preparations alone does not tell us which of their differences an observer must recover.

This chapter gives that distinction a precise form. We first prove the collision argument for an arbitrary record. We then construct the record of an adaptive testing procedure and count its possible outcomes. Finally, we replace exact answers by answers with a stated error. Each argument concerns the information supplied to the reader; physical preparation, retention and readout enter as separate requirements on the apparatus.

13.1 The target, the record and the reader

Let W be the allowed preparations or worlds. A member $w \in W$ contains whatever distinctions the experiment can depend on. We need not reconstruct all of w . Instead, fix a target map

$$\tau : W \longrightarrow Y.$$

The value $\tau(w)$ is the answer the task requires. Two worlds may differ while having the same target value.

Let R be the possible retained records and let $r : W \rightarrow R$ give the record actually obtained in each world for the procedure under consideration. A reader d takes a record and either returns an element of Y or reports failure. We write its answer as $d(r)$ when it succeeds.

Definition 13.1 (Exact readout). The procedure gives an exact readout of τ if the reader succeeds on every actual record and

$$d(r(w)) = \tau(w) \quad \text{for every } w \in W. \quad (13.1)$$

Records that never occur may be rejected.

The order of these maps matters. The apparatus acts on the world to produce a record; the reader acts on the record to return the target. An arbitrary assignment of a different code to every answer does not establish that the apparatus can obtain the assigned code from an unknown preparation. Equation (13.1) is the required connection.

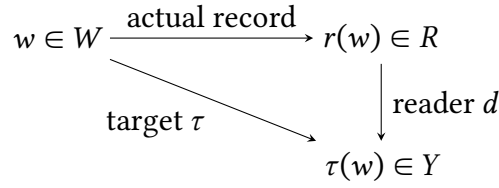


Figure 13.1. Exact readout requires the two routes to agree for every allowed world. A mathematical code supplies the upper route only when the apparatus actually produces it.

Lemma 13.2 (A record cannot separate its own collisions). *If the readout is exact and $r(w) = r(v)$, then $\tau(w) = \tau(v)$.*

Proof. Apply the same reader to the equal records. Exactness identifies its answer on one side with $\tau(w)$ and on the other with $\tau(v)$. Thus

$$\tau(w) = d(r(w)) = d(r(v)) = \tau(v).$$

□

Only distinctions relevant to the target must survive. If the target itself separates all worlds, the lemma says that the record map must be injective. Otherwise, several worlds can share a record without causing an error, provided their required answers agree.

13.2 Several worlds can remain compatible with one record

The map r describes one determined record per world. Some experiments have several possible outcomes, and some records leave several preparations possible. We can state the essential condition without choosing a probability distribution or a deterministic outcome map.

Write $C(s, w)$ when record s is compatible with world w under the stated experimental model. The condition includes the experiment that was performed and every relevant control or calibration assumption. A reader is sound on this relation if, whenever $C(s, w)$ holds, it succeeds and returns $\tau(w)$.

Proposition 13.3 (A sound answer is constant on compatible worlds). *Suppose the reader is sound. If $C(s, w)$ and $C(s, v)$, then $\tau(w) = \tau(v)$. Moreover, if an actual record map satisfies $C(r(w), w)$ for every w , soundness gives exact readout for that map.*

Proof. Soundness says that the single answer $d(s)$ equals both target values, which proves the first assertion. For the second, apply soundness to $s = r(w)$ and the stated actual-world compatibility condition. □

This formulation separates two possible failures. A proposed record may be compatible with worlds requiring different answers. Or the compatibility model may omit the world that actually produced the record. The first failure concerns discrimination; the second concerns

the model's relation to the experiment. A proof of the first condition does not repair the second.

The proposition is a necessary condition on a sound reader. It does not construct an effective reader from an arbitrary compatibility relation. Finding the compatible worlds, deciding their common target and making the record readable can each require further work.

13.3 A finite capacity bound

Choose a finite family of allowed worlds w_i , indexed by a finite set I , whose target values are pairwise different. The family need not include every allowed world. Suppose the whole retained record has only M possible values.

Theorem 13.4 (Capacity required by exact recognition). *If the readout is exact on the allowed worlds, then*

$$|I| \leq M.$$

Proof. Associate i with its actual record $r(w_i)$. If two indices have the same record, Lemma 13.2 gives the same target value. The choice of the family then gives the same index. We have constructed an injection from I into the set of possible records, so the finite cardinal inequality follows. $\square$

The bound concerns distinguishable records, including everything the reader may use. If timing, position, the choice of output channel or a separate control register supplies additional information, it belongs in R . Counting only the printed mark would then count an incomplete record.

No optimality claim is built into the theorem. An injection into a large record space is necessary for exact recovery; it does not show how a particular apparatus reaches the needed records. The construction of a working procedure is a separate positive task.

13.4 Adaptive tests still produce a bounded transcript

An experimenter may choose the next test after seeing the previous result. To count this procedure correctly, we must preserve that dependence. Fix a set A of available tests and an alphabet of $B \geq 1$ possible outcomes. For this section, a test a has a determined outcome

$$o(a, w) \in \{0, \dots, B - 1\}$$

in each world w . Fix the procedure in advance. It may adapt to the outcomes, but it receives no information about w other than through those outcomes.

Definition 13.5 (A bounded adaptive test tree). A tree with budget m has one of two forms:

1. Stop and return a specified answer, using none of the remaining tests.
2. If $m > 0$, choose a test $a \in A$ and, for each of its B outcomes, supply a continuation tree with budget $m - 1$.

Execution follows the continuation selected by the observed outcome. Write $\text{run}(T, w)$ for the answer eventually returned by tree T .

The budget decreases at every test, so every execution stops. Different branches can choose different tests and stop at different depths. This is a finite inductive description of the whole policy, rather than a list of independent tests to which adaptation is added afterward.

Choose one padding symbol p from the outcome alphabet. We now construct a transcript $c_T(w)$ with exactly m slots. At a stop node, fill all remaining slots with p . At a test node, write the observed outcome in the first slot and follow it with the continuation's transcript.

The corresponding reader already knows T . At a stop node it returns that node's answer and ignores the remaining slots. At a test node it reads the first symbol, chooses the corresponding continuation, and reads the tail there. Denote this reader by D_T .

Lemma 13.6 (The transcript retains the answer of the procedure). *For every bounded tree and every allowed world,*

$$D_T(c_T(w)) = \text{run}(T, w).$$

Proof. Induct on the tree. At a stop node, both sides are its specified answer. At a test node, the first transcript symbol is $o(a, w)$. The reader therefore chooses exactly the continuation chosen in execution. The remaining transcript is the transcript of that continuation. Apply the induction hypothesis to its smaller tree. $\square$

Padding loses no information needed for this answer. The fixed tree tells the reader when a stop node has been reached. Even if a padding symbol is also a genuine test outcome, the reader knows which slots it is supposed to read from the branch it has followed.

Theorem 13.7 (Capacity of a bounded adaptive procedure). *Suppose a tree performs at most m tests with B possible outcomes per test and returns the correct target for every world in a finite family of N distinct target values. Then*

$$N \leq B^m. \tag{13.2}$$

Proof. There are B^m strings of length m over the outcome alphabet: each of the m slots has B choices. Lemma 13.6 and correctness of the tree give exact readout from its constructed transcript. Distinct target values must therefore give distinct transcripts. Apply Theorem 13.4. $\square$

In particular, no such procedure with one binary test can always identify one of three distinct targets: the conclusion would require $3 \leq 2$. Adaptation helps choose useful questions, but it cannot make a transcript carry more distinct answers than it has distinguishable values.

This tree model assumes determined outcomes $o(a, w)$ for the supplied world. When a test changes the system, its changing state and the available controls must be represented before applying this construction. The earlier collision and record-capacity theorems apply to any supplied actual record map with exact recovery; the tree theorem adds the particular structure that justifies the bound B^m .

13.5 An error tolerance gives a separation bound

Many tasks ask for an approximation. Let target values lie in a space with a distance dist satisfying symmetry and the triangle inequality. Let the reader return a reconstructed value $d(s)$, and fix an error allowance $\varepsilon \geq 0$. Suppose

$$\text{dist}(\tau(w), d(r(w))) \leq \varepsilon \quad \text{for every allowed } w. \quad (13.3)$$

Lemma 13.8 (A shared record bounds target separation). *If $r(w) = r(v)$, then*

$$\text{dist}(\tau(w), \tau(v)) \leq 2\varepsilon.$$

Proof. Both worlds are reconstructed as the same value $y = d(r(w)) = d(r(v))$. The triangle inequality and the two error guarantees give

$$\text{dist}(\tau(w), \tau(v)) \leq \text{dist}(\tau(w), y) + \text{dist}(y, \tau(v)) \leq 2\varepsilon.$$

□

Theorem 13.9 (Capacity for separated approximate targets). *Suppose a finite family has N target values, every distinct pair separated by more than 2ε . A readout satisfying (13.3) on this family requires at least N distinguishable records. For a bounded adaptive tree as above, it requires $N \leq B^m$.*

Proof. If two members had the same record, Lemma 13.8 would contradict their strict separation. The record map is therefore injective on the selected family, giving the first bound. For the tree, use its padded transcript and reader. Lemma 13.6 transfers the procedure's error guarantee to this reader, and there are B^m possible transcripts. □

The strict inequality is the relevant margin. An approximation is permitted to merge sufficiently nearby targets. The theorem counts a family whose members are far enough apart that the specified error cannot hide their differences. It supplies a necessary capacity, without assuming a geometry or an optimal code beyond the stated distance and separation.

13.6 Finite length without a common budget

A different question arises when every individual record is finite but there is no single length bound for all records. Such a record family need not be finite. Its binary strings can still be encoded by natural numbers. Define

$$c(\emptyset) = 0, \quad c(0s) = 2c(s) + 1, \quad c(1s) = 2c(s) + 2. \quad (13.4)$$

Here 0s and 1s prepend a symbol to the finite string s .

Lemma 13.10 (Finite binary records have distinct natural-number codes). *If $c(s) = c(t)$, then $s = t$.*

Proof. Induct on s . The empty string has code zero; every nonempty string has positive code, so equality with the empty string forces the other string to be empty. For two nonempty strings, a leading zero gives an odd code and a leading one gives an even code. Equal codes therefore have the same leading symbol. Subtract its offset and cancel the factor two to obtain equal tail codes. The induction hypothesis gives equal tails, hence equal strings. $\square$

Theorem 13.11 (Exact finite-record identification is countable). *Suppose every world yields a finite binary record, a reader exactly recovers its target, and different worlds have different target values. Then the world family injects into the natural numbers and is countable.*

Proof. Lemma 13.2 makes the record map injective under the target-distinctness assumption. Compose that map with the injective code of Lemma 13.10. The result is an injection $w \mapsto c(r(w))$ into the natural numbers. $\square$

No uniform length bound was used. Conversely, countability alone does not give a uniform record length, a time bound or an apparatus that can acquire the record. This is why the fixed-budget theorem and the countability theorem answer different questions.

The hypothesis is exact identification of the stated target family. A finite record can support a universal mathematical statement when it contains a proof whose rules justify that statement. The record-capacity argument does not count the objects mentioned in a proposition as though the reader had to identify each of them separately. It concerns a task whose different required answers must be distinguished by actual records.

13.7 The physical task left by the mathematics

The record results expose a concrete obligation for a physical construction. Specify the allowed preparations and the target. Show how the procedure produces the record, how it is retained, and how its reader returns the required answer within the stated error. Account for all information available to the reader and for the resources the procedure uses. The next two chapters take up that positive construction.

A counting theorem can reject a proposed capacity before an apparatus is built. A large enough capacity does not complete the apparatus. Likewise, an exact logical principle from Chapter 12 concerns which conclusions follow from an infinite presentation; a finite physical record requires an account of how the relevant information reaches the reader. Both questions have exact mathematical forms. Keeping their premises visible is what makes the connection useful.

Questions to work through

1. In Lemma 13.2, identify the assumption that fails if the reader also receives an unrecorded control setting. How should the record space be repaired?
2. Follow the transcript reader through a tree that stops before using its full budget. Explain why the padding symbol need not be distinguishable from every genuine test outcome.
3. Prove Theorem 13.4 for a selected family that is smaller than the whole world space. Which map must be injective?
4. Identify the uses of symmetry and the triangle inequality in Lemma 13.8. Explain why the separated-family theorem uses a strict margin.
5. Compare the hypotheses of Theorems 13.7 and 13.11. Explain why finite length for each record does not supply the same conclusion as a fixed common budget.

Chapter 14

The construction of finite operations

A description of an apparatus must say more than which mark it prints. The apparatus can change its input, return a quantum system as well as a mark, and act on something correlated with another system. A construction that reproduces the marks while destroying those correlations may perform a different operation.

We will follow a finite instrument through four descriptions. First come the matrices that specify its action. Those matrices give an explicit reversible operation on a larger space. Finite sequences of elementary operations approximate that reversible operation. Finally, positive energy operators enact the required sequence, with its duration, error and scalar energy contribution retained.

There is a second question along the way: can a finite procedure find the approximating sequence from a description of the instrument? We prove an equivalence between two effective descriptions. One supplies rational approximations to the instrument's matrices; the other supplies finite operation sequences with a prescribed accuracy. The proof uses a search with rational certificates, so it never asks an exact equality question about an arbitrary real number.

The first calculation needs only multiplication of two-by-two matrices. Use it to keep the purpose of the larger construction in view: we must retain both the mark and the returned system. The later proofs use the finite spectral theorem to factor positive matrices and take unitary logarithms. The final control argument uses matrix exponentials and their derivatives. The matrix sections of Appendix C explain how these tools enter, including the difference between an isometry into a larger space and a unitary on that whole space.

14.1 An operation with a retained outcome

Fix positive integers a, b, s . The input has a basis states, the returned system has b basis states, and the classical mark has s possible values. The matrices in this chapter have complex entries. Write $K^\dagger$ for the conjugate transpose of K , and I for an identity matrix on the space indicated by the equation.

A state is a positive semidefinite matrix ρ with trace one. Positivity means $v^\dagger \rho v \geq 0$ for every vector v . This includes the state $vv^\dagger$ of a unit vector and also statistical mixtures of such states. The trace condition fixes the total probability.

The same marks can conceal a different operation

Consider a system with two basis states and two possible marks, 0 and 1. The matrices

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad K_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad (14.1)$$

select the first and second basis states. Write an input state as $\rho = \begin{pmatrix} p & z \\ \bar{z} & q \end{pmatrix}$, so $p, q \geq 0$ and $p + q = 1$. Multiplication gives the two output blocks

$$K_0 \rho K_0^\dagger = \begin{pmatrix} p & 0 \\ 0 & 0 \end{pmatrix}, \quad K_1 \rho K_1^\dagger = \begin{pmatrix} 0 & 0 \\ 0 & q \end{pmatrix}.$$

Their traces are p and q : the probabilities of the two marks. Their sum has trace one because $K_0^\dagger K_0 + K_1^\dagger K_1 = I$.

Now keep K_0 but replace the second matrix by

$$L_1 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

It still satisfies $K_0^\dagger K_0 + L_1^\dagger L_1 = I$, and

$$L_1 \rho L_1^\dagger = \begin{pmatrix} q & 0 \\ 0 & 0 \end{pmatrix}.$$

The probability of mark 1 is still q . For every input, both devices therefore print the same marks with the same probabilities. But after mark 1, the first device leaves the second basis state and the second device returns the first. On input $\rho = K_1$, mark 1 is certain and the returned states are different.

A description that keeps only the two probabilities cannot distinguish these devices. An instrument keeps the output blocks as well. The general definition below adds several possible branches for each mark; the example has one branch per mark.

Definition 14.1 (A finite instrument). Choose a finite branch set Z and matrices $K_{yz} : \mathbb{C}^a \rightarrow \mathbb{C}^b$, where y is the retained mark. Require

$$\sum_{y,z} K_{yz}^\dagger K_{yz} = I_a. \quad (14.2)$$

The output is the block diagonal matrix whose y th block is

$$\mathcal{J}_y(\rho) = \sum_z K_{yz} \rho K_{yz}^\dagger.$$

The trace of this block is the probability of mark y . The mark is retained even when the branch label z is not read.

Every block is positive, since $u^\dagger K \rho K^\dagger u = (K^\dagger u)^\dagger \rho (K^\dagger u) \geq 0$. Taking the trace and using (14.2) shows that the traces of the output blocks sum to one. Thus the formula describes both the returned system and the probabilities of all marks.

Different lists of branch matrices can describe the same instrument. To avoid treating a choice of branch decomposition as extra information, form one matrix for each outcome:

$$C_y[(u, i), (v, j)] = \sum_z K_{yz}[u, i] \overline{K_{yz}[v, j]}. \quad (14.3)$$

Here i, j are input indices and u, v are output indices. These are the *Choi matrices* [10], with output indices written first. They determine the instrument because

$$\mathcal{J}_y(\rho)[u, v] = \sum_{i,j} C_y[(u, i), (v, j)] \rho[i, j].$$

Proposition 14.2 (Valid matrix descriptions). *A family C_y describes a finite instrument precisely when*

$$C_y \geq 0 \quad \text{for every } y, \quad \sum_{y,u} C_y[(u,i), (u,j)] = \delta_{ij}. \quad (14.4)$$

For such a family, one can take $Z = \{1, \dots, b\} \times \{1, \dots, a\}$.

Proof. Given branch matrices, arrange their entries into columns indexed by z . Equation (14.3) is the product of this column matrix with its adjoint, so it is positive. Summing its output diagonal gives the transpose of $\sum_{y,z} K_{yz}^\dagger K_{yz}$, which is I_a .

Conversely, a positive finite matrix has a positive square root. This can be constructed by taking the nonnegative square roots of its eigenvalues in an orthonormal eigenbasis. Let $S_y = C_y^{1/2}$, and reshape its z th column into a b by a matrix:

$$K_{yz}[u, i] = S_y[(u, i), z].$$

The product $S_y S_y^\dagger = C_y$ gives (14.3). The normalization in (14.4), with i, j interchanged, gives (14.2). These are all the required conditions. $\square$

This is a finite-dimensional existence construction. An exact choice of eigenvectors is not yet a procedure for obtaining an operation from successively better rational data. We will obtain that procedure by a different route.

14.2 An explicit reversible extension

The next construction is a finite-dimensional dilation: it realizes an operation as part of a reversible one on a larger space. Dilation theory originates with Stinespring [9]; here the block formula makes the action on the entire enlarged space explicit.

Imagine keeping every branch label for the moment. Define the rectangular matrix V by stacking the branch matrices:

$$V[(u, y, z), i] = K_{yz}[u, i].$$

Equation (14.2) says $V^\dagger V = I_a$. Thus V preserves lengths and inner products. It embeds the input into the larger space D with basis labels (u, y, z) .

We need a reversible operation on a whole space, including states outside this embedded input. Rather than leave its action there unspecified, write it down.

Theorem 14.3 (A reversible extension with its unused space retained). *If $V^\dagger V = I_a$, put $P = VV^\dagger$. On $\mathbb{C}^a \oplus D$ define*

$$U_V = \begin{pmatrix} 0 & V^\dagger \\ V & I_D - P \end{pmatrix}. \quad (14.5)$$

Then $U_V^\dagger = U_V$, $U_V^2 = I$, and

$$U_V(v, 0) = (0, Vv).$$

Reading the output and mark coordinates, while retaining the complete branch sum, gives exactly the original instrument.

Proof. The matrix P is self-adjoint and $P^2 = V(V^\dagger V)V^\dagger = P$. Also $(I_D - P)V = 0$ and $V^\dagger(I_D - P) = 0$. Therefore block multiplication gives

$$U_V^2 = \begin{pmatrix} V^\dagger V & V^\dagger(I_D - P) \\ (I_D - P)V & VV^\dagger + (I_D - P)^2 \end{pmatrix} = \begin{pmatrix} I_a & 0 \\ 0 & I_D \end{pmatrix}.$$

Self-adjointness follows by transposing the blocks and conjugating them. Multiplication by $(v, 0)$ gives $(0, Vv)$. Its (u, y, z) component is $K_{yz}v$ at output index u , so summing the unread branches produces the instrument's y th block. The same calculation applies to mixed input states by matrix multiplication and linearity. $\square$

The extension is a concrete answer to the question, “What happens on the rest of the space?” Its complement is part of the operation. The input preparation must place the state in the designated input summand; that requirement is not supplied by the block identity itself.

14.3 A fixed register and a complete reader

Use a register of n eight-state cells. Its basis is the set of functions from $\{1, \dots, n\}$ to $\{0, \dots, 7\}$, so its dimension is 8^n . The choice of register can depend on a, b, s while remaining fixed for every valid instrument of those dimensions.

There is a convenient way to place a unitary in a determinant-one matrix. If U acts on a space E , use

$$\tilde{U} = U \oplus U^\dagger \oplus I_R. \quad (14.6)$$

This is unitary, has determinant $\det(U) \det(U^\dagger) = 1$, and agrees with U on the first summand. Choose $n \geq 2$ with $8^n \geq 2 \dim E$, take $\dim R = 8^n - 2 \dim E$, and fix a bijection of basis labels. Such an n exists; the deliberately generous choice $n = 2 \dim E + 2$ suffices. Applied to (14.5), this supplies one layout for all instruments with the chosen dimensions.

An approximation to $\tilde{U}$ need not keep the input entirely within the intended output rows. The reader must still account for the whole output. Split the rows into normal rows (u, y, z) and complementary rows ℓ . Choose a default output u_0 and mark y_0 . Each normal row retains its own output and mark. Each complementary row has a separate unread branch and is assigned to (u_0, y_0) .

Proposition 14.4 (Complete readout preserves normalization). *Let W be the columns of a unitary selected by the fixed input embedding. The branch matrices obtained by the above reader satisfy*

$$\sum_{y,z'} K_{yz'}^\dagger K_{yz'} = W^\dagger W = I_a,$$

where z' ranges over both normal and complementary branches. For the ideal placed unitary, the complementary branches vanish and its Choi matrices are the specified C_y .

Proof. Each row of W occurs exactly once in the sum of branch Gram matrices. Normal rows occur with their own output labels. A complementary row occurs at the default output and mark, with its own branch label. Expanding a matrix entry of the sum therefore gives exactly the same row sum as $W^\dagger W$. Distinct columns of a unitary are orthonormal. For the ideal operation, Theorem 14.3 puts the selected columns in the normal rows with entries $K_{yz}[u, i]$ and zero elsewhere. $\square$

No outcome has been discarded or conditionally renormalized. Leakage into an unintended row is an error in the instrument, and the comparison below counts it.

14.4 An error bound that keeps a reference system

Suppose the input is correlated with a retained reference Q of any finite dimension. Its joint state τ is a positive trace-one matrix on $Q \otimes \mathbb{C}^a$. Define the joint output by retaining the reference, the returned system and the mark. Its y th diagonal block has entries

$$\mathcal{J}_C^Q(\tau)[(y, r, u), (y, r', v)] = \sum_{i,j} C_y[(u, i), (v, j)] \tau[(r, i), (r', j)]. \quad (14.7)$$

Blocks with different marks are zero. The branch factorization proves positivity, since this is a sum of $(I_Q \otimes K_{yz}) \tau (I_Q \otimes K_{yz})^\dagger$.

For a Hermitian matrix X , write $\|X\|_1$ for the sum of the absolute values of its eigenvalues. We say that two instruments differ by at most ε on retained references if

$$\|\mathcal{J}_C^Q(\tau) - \mathcal{J}_D^Q(\tau)\|_1 \leq \varepsilon \quad (14.8)$$

for every finite Q and every joint state τ . This convention has no factor of one half. It requires no independence between input and reference.

Lemma 14.5 (Entry errors control the joint output). *Suppose C, D are valid instruments and, for every entry and mark, $|C_y[p, q] - D_y[p, q]| \leq \eta$, with $\eta \geq 0$. Then their error on retained references is at most*

$$sab^2\eta. \quad (14.9)$$

The bound does not grow with the reference dimension.

Proof. For a matrix A of size ab with entries bounded in absolute value by η , the triangle inequality and Cauchy–Schwarz give

$$|v^\dagger A v| \leq \eta \left(\sum_p |v_p| \right)^2 \leq ab\eta \sum_p |v_p|^2.$$

Take $A = C_y - D_y$, which is Hermitian. With $r = ab\eta$, this proves $rI \pm (C_y - D_y) \geq 0$ for every y .

Apply the positive output construction (14.7) to these two families. If X is the output difference and F is the output of the family with every block equal to rI , then $F + X \geq 0$ and $F - X \geq 0$. Thus, for every unit vector v , $|v^\dagger X v| \leq v^\dagger F v$. Choose an orthonormal eigenbasis of X and sum this inequality. It gives $\|X\|_1 \leq \text{tr } F$.

The diagonal sum in (14.7), applied to the family rI , is $\text{tr } F = sbr \text{tr } \tau = sbr$. Substituting $r = ab\eta$ proves (14.9). The only reference contribution was its joint state inside $\text{tr } \tau = 1$. $\square$

Let m be the number of normal plus complementary unread branch labels in the fixed complete reader. If two unitaries are within η in operator norm, every corresponding entry differs by at most η . Their entries also have absolute value at most one. The elementary identity

$$x\bar{y} - u\bar{v} = (x - u)\bar{y} + u\overline{(y - v)}$$

therefore bounds each product difference by 2η . A Choi entry sums over m branches. Applying the lemma gives the useful implication

$$\|U - W\| \leq \eta \implies \text{joint output error} \leq L\eta, \quad L = 2msab^2. \quad (14.10)$$

All dimensions and the branch count are fixed before the target instrument and requested tolerance.

14.5 Two descriptions of finite operations

We now distinguish a finite alphabet used to describe operations from the continuous controls used to enact them. A cell label has three binary coordinates. On these coordinates use the Hadamard matrix

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

the phase matrix $\text{diag}(1, i)$, and the Toffoli permutation that flips one coordinate when two selected control coordinates are both one. The target differs from each control, but the controls may coincide; that case is controlled-NOT. A word is a finite list of such gates with their coordinate labels. Matrix multiplication assigns a unitary to each word.

Two properties of this alphabet do different jobs. First, a word's matrix and complete Choi readout admit effective rational approximations. Every entry is obtained by finitely many sums and products in $\mathbb{Q}(\sqrt{2}, i)$; certified rational enclosures for $\sqrt{2}$, propagated through those finite operations, give any requested entry accuracy. Second, on a fixed register with at least two cells, these words are dense in its special unitary group. The finite gate and matrix-generation

arguments are developed in Appendix A. We need the existence of an approximating word here, without a runtime estimate for finding it.

Together with the explicit dilation and fixed placement, density gives a word whose complete Choi readout is within any positive entry tolerance of any valid target C . Indeed, approximate the placed unitary in operator norm and use the product estimate preceding (14.10).

Definition 14.6 (An effective instrument description). An ordinary name of C is one finite procedure which, on input k , returns the full array of rational complex entries $q_y^{(k)}[p, q]$ and guarantees

$$|q_y^{(k)}[p, q] - C_y[p, q]| \leq 2^{-k}.$$

A word name is one finite procedure which, on input k , returns a word whose complete readout differs from C by at most 2^{-k} on every finite retained reference. Its register and layout remain fixed as k varies.

The procedures are required to finish at every requested precision for the instrument they name. A mere assertion that a suitable word exists for each k does not yet give a word name.

14.6 Searching with a margin

To find a word, enumerate all finite words and test a rational certificate. For complex rationals z, w , use

$$d_1(z, w) = |\operatorname{Re}(z - w)| + |\operatorname{Im}(z - w)|.$$

This is decidable by rational arithmetic, and

$$|z - w| \leq d_1(z, w) \leq 2|z - w|. \quad (14.11)$$

The left inequality is the triangle inequality for real and imaginary parts; each part has absolute value at most $|z - w|$, proving the right. Test every Choi entry, not just the outcomes most likely on one input.

For requested output error 2^{-k} put

$$M = sab^2, \quad h = k + M + 3, \quad r = 2^{-h}, \quad e = 2^{-(h+3)} = r/8. \quad (14.12)$$

Obtain target data q at entry accuracy e . For each candidate word, compute its complete readout data q_W at the same accuracy. Accept the first word for which $d_1(q_W[y, p, q], q[y, p, q]) \leq r$ at every entry.

Lemma 14.7 (Soundness and termination of the search). *For a valid named target, the search terminates. Every accepted word has joint output error at most 2^{-k} .*

Proof. If a word is accepted, its exact Choi entries differ from those of the target by at most

$$e + r + e = \frac{5}{4}r \leq 2r.$$

The three terms are word-evaluation error, accepted rational discrepancy and target-naming error. Lemma 14.5 bounds the joint output error by $M(2r)$. Since $M \leq 2^M$,

$$M(2r) \leq 2^M 2^{-(k+M+3)} = 2^{-k}/4 \leq 2^{-k}.$$

For termination, density supplies a word whose exact Choi entries are within e of the target. Its rational data and the target data are then within $3e = 3r/8 < r/2$ in absolute complex value. By (14.11), their rational discrepancy is less than r , so this word passes. Every individual test finishes, and enumeration eventually reaches a passing word. $\square$

The slack has a purpose. Testing exact equality of two limits would not give a terminating decision. Here the acceptance test is finite, and density supplies a word strictly inside its tolerance. A nearby rejected candidate simply causes the search to continue.

14.7 Recovering the matrix description

The reverse direction also needs the retained reference. Suppose an operation is known to approximate an instrument on every joint input. Use a reference of dimension a and the normalized state

$$\tau_a = \frac{1}{a} \Omega \Omega^\dagger, \quad \Omega = \sum_{i=1}^a e_i \otimes e_i. \quad (14.13)$$

Its trace is one, and it is positive. Substitution in (14.7) gives

$$\mathcal{J}_C^Q(\tau_a)[(y, i, u), (y, j, v)] = \frac{1}{a} C_y[(u, i), (v, j)].$$

Thus this one joint input exposes every Choi entry, with the known factor $1/a$.

Lemma 14.8 (Joint output control recovers every entry). *If C, D have retained-reference error at most ε , then $|C_y[p, q] - D_y[p, q]| \leq a\varepsilon$ at every entry.*

Proof. Apply the error bound to (14.13). Every entry of a Hermitian matrix has absolute value at most its trace norm: expand it in an orthonormal eigenbasis, use the triangle inequality, and bound each product of eigenvector coordinates by one. The displayed probe identity therefore gives $|C_y[p, q] - D_y[p, q]|/a \leq \varepsilon$. $\square$

Theorem 14.9 (Equivalence of effective finite descriptions). *For fixed positive a, b, s and a valid finite instrument C , an ordinary name exists if and only if a word name exists. The forward translation can be carried out by one procedure fixed for those dimensions and their register layout, taking the ordinary name as an input.*

Proof. Given an ordinary name, run the rational search at each requested precision. Lemma 14.7 proves both termination and the required joint output accuracy. The instructions for evaluation, enumeration and testing are fixed; they call the supplied name at the precision in (14.12). This is the stated uniform forward translation.

Conversely, given a word name and requested entry accuracy 2^{-k} , call the word name at precision $j = k + (a+1) + 3$. Evaluate that word's Choi entries to accuracy 2^{-j} . The probe lemma

bounds the difference between its exact entries and the target entries by $a2^{-j}$. The triangle inequality gives total entry error at most $(a+1)2^{-j} \leq 2^{-k}$. The last inequality follows from $a+1 \leq 2^{a+1}$ and the extra factor 2^{-3} . This constructs a single ordinary naming procedure. $\square$

The equivalence concerns descriptions of the whole instrument. It neither decides whether an arbitrary proposed description is valid nor gives a general efficient search bound. Validity and the supplied name are the premises under which termination has been proved.

14.8 Enacting a sequence with positive energy

The word description is now connected to a physical evolution. We state the controls as well as the energy they require. On the native register, the continuous elementary operations are cyclic three-state rotations within a cell and the two orderings of the adjacent-cell coupling

$$E = \frac{I + iX}{\sqrt{2}}, \quad X = S \otimes F.$$

Here S is diagonal, with

$$S = \text{diag}(1, -1, -1, 1, -1, 1, 1, -1).$$

The operator F swaps each even label with the following odd label. Both S and F are Hermitian involutions. Each elementary operation is e^G for an explicitly specified skew-Hermitian generator G . The rotations have continuously selectable real angles. These controls generate the full traceless skew-Hermitian matrix algebra on registers of at least two cells; the bracket argument and passage to finite words appear in Appendix A. Consequently their finite words approximate every special unitary on that register.

Fix a positive background energy operator $H \geq 0$, independently of the target operation. Let $\hbar > 0$ and choose a pulse duration $t > 0$. For a skew-Hermitian generator G , use the full energy operator

$$H_{G,t} = H + \frac{\hbar}{t}(iG + \|G\|I). \quad (14.14)$$

Since iG is Hermitian and its eigenvalues are at least $-\|G\|$, the parenthesized operator is positive. Hence $H_{G,t} \geq 0$. The scalar term is part of the energy operator even when a particular readout is insensitive to its phase.

The evolution for duration t is $e^{-itH_{G,t}/\hbar}$. The scalar part commutes with everything, so this equals

$$e^{-i\|G\|} \exp(G - itH/\hbar).$$

For skew-Hermitian matrices A, B , differentiating $e^{sA}e^{(1-s)B}$ and integrating from zero to one gives $\|e^A - e^B\| \leq \|A - B\|$: both exponential factors in the integrand are unitary and have norm one. Therefore

$$\|e^{-itH_{G,t}/\hbar} - e^{-i\|G\|}e^G\| \leq t\|H\|/\hbar. \quad (14.15)$$

For a word w of ℓ pulses, subtract the two products one factor at a time. Every other factor has norm one. The errors add, giving

$$\|U_{w,t} - e^{-ic(w)}W_w\| \leq \ell t \|H\|/\hbar, \quad c(w) = \sum_{G \text{ in } w} \|G\|. \quad (14.16)$$

Here W_w is the ideal word matrix and $U_{w,t}$ is its actual evolution under the full pulse energies.

14.8.1 Retaining the phase

One can also match a target's full scalar phase. This matters when claiming an operator approximation, even though a common phase cancels in the quadratic instrument readout. A pair of inverse cyclic rotations leaves the ideal word unchanged and contributes 2θ to $c(w)$ when $\theta \geq 0$. For a desired phase e^{-ir} , choose

$$z = \left\lceil \frac{c(w) - r}{2\pi} \right\rceil, \quad \theta = \frac{r + 2\pi z - c(w)}{2}. \quad (14.17)$$

The defining inequalities for the ceiling give $0 \leq \theta < \pi$. Appending the inverse pair makes the total scalar action $r + 2\pi z$, so the phase is exactly e^{-ir} . Both added pulses retain H and their positive scalar baseline.

Every finite unitary has a Hermitian logarithm: diagonalize the unitary, choose a real argument for each unit-modulus eigenvalue, and conjugate the diagonal arguments back. Subtracting the mean trace separates this logarithm into a scalar part and a traceless part. Approximate the latter by the native controls, and apply (14.17) to the former. This supplies an ideal word with the requested phase to any positive operator tolerance.

Theorem 14.10 (Finite positive-pulse realization). *Fix a native register with at least two cells, $H \geq 0$, and $\hbar > 0$. For any unitary target U , error tolerance $\delta > 0$ and total duration bound $T > 0$, there is a finite native word and a positive common pulse duration t such that its full energy operators are positive, $\ell t < T$, and its actual evolution differs from U by less than δ in operator norm. The construction allows the amplitudes in (14.14) and externally selected switching.*

Proof. Choose the ideal word and phase correction with error less than $\delta/2$, and let ℓ be its full length, including the two phase pulses. Set

$$t = \min \left\{ \frac{\delta \hbar}{2(\ell \|H\| + 1)}, \frac{T}{\ell + 1} \right\}. \quad (14.18)$$

Both quantities are positive. The first gives $\ell t \|H\|/\hbar \leq \delta/2$, while the second gives $\ell t \leq \ell T/(\ell + 1) < T$. Equation (14.16) and the ideal approximation error then give a total error less than δ . Positivity was proved directly from (14.14). $\square$

Shorter duration does not make the controls free. Their strength includes the factor $\hbar/t$. The integrated scalar baseline is exactly

$$\sum_{G \text{ in } w} t \frac{\hbar}{t} \|G\| = \hbar c(w). \quad (14.19)$$

The duration cancels; the action remains. This equality records that baseline contribution, rather than a complete model of actuator work, switching equipment or total experimental cost. A bound on available drive strength would add a constraint that Theorem 14.10 does not impose.

14.9 The finite bridge and its physical interfaces

Combine the constructions in their actual order. Positivity and normalization give branch matrices. Stacking gives an isometry. The block matrix gives a reversible extension. Adjoint padding puts it on a fixed register, complete readout counts every row, and the pulse construction approximates the placed unitary.

Theorem 14.11 (Realization of a complete finite instrument). *Fix positive dimensions a, b, s and default output and mark labels. There is a finite native register and a fixed input and complete-output layout, chosen before the target instrument, with the following property. For any valid C , any fixed positive background energy H on that register, $\hbar > 0$, tolerance $\varepsilon > 0$ and duration bound $T > 0$, the positive-pulse construction produces a complete instrument whose retained-reference error is at most ε , with total pulse duration less than T . It retains the scalar baseline (14.19). It uses the selectable controls, input preparation and complete readout described above.*

Proof. Choose the register and layout using the fixed branch size ab and the dilation space; neither depends on the entries of C . Construct its ideal placed unitary U_C as above. Let $L = 2msab^2$ be the constant of the complete reader in (14.10). Apply Theorem 14.10 with operator tolerance $\eta = \varepsilon/(L + 1)$ and duration bound T . Its output is unitary, so the complete reader remains normalized. Its error on every retained reference is at most $L\eta \leq \varepsilon$. The ideal reader gives C exactly, and every pulse retains its positive energy and baseline contribution. $\square$

Section A.8 gives a concrete record-and-field assembly to which the pulse theorem applies. It retains an old source on $14 + 2N$ cells, adds three complete record–reference pairs and two field banks, and specifies the added holding and edge energies. Its full register has $20 + 2N + 4a + 4b$ eight-state cells. Positivity follows on the whole register, including coordinates unused by the field code.

The old source’s positive interior Hamiltonian is an explicit input to that assembly. The appendix does not derive the material source fixture or a microscopic actuator for it. Thus the application to the native two-flux source retains this supplied Hamiltonian, its fixed external fluxes and the same preparation, readout and control interfaces. The on-page construction establishes the record-and-field extension and the full-register transport needed here. It does not replace an unprovided source construction by a claim of universal physical access.

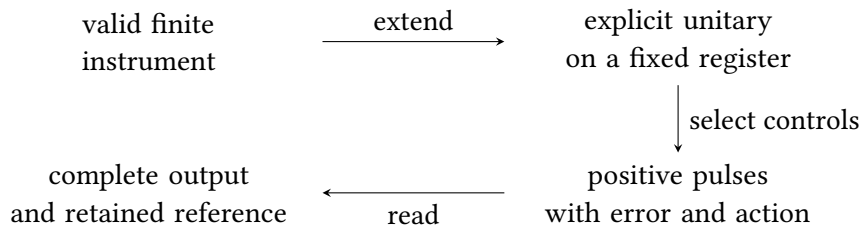


Figure 14.1. The finite bridge retains the whole operation. Input preparation, selectable controls and the complete reader are physical interfaces required by the construction. Error is compared on the returned system, mark and every finite retained reference.

The conclusion has two complementary strengths. The naming equivalence shows that effective matrix descriptions and effective finite operation descriptions determine the same

valid finite instruments. The pulse construction supplies explicit positive energy operators and carries their error through the complete reader. Establishing an autonomous apparatus with bounded controls, or establishing that every possible physical carrier belongs to this finite class, requires additional constructions. Those questions concern the remaining interfaces of this particular result.

Exercises

1. In the proof of Theorem 14.3, identify the exact use of $V^\dagger V = I$. Explain the role of $I_D - VV^\dagger$.
2. Expand one entry of the Gram sum for the complete reader. Explain why two complementary rows must not be combined into a single coherent branch before taking their Gram matrices.
3. Reproduce the three error terms in an accepted rational certificate and the three terms used to prove that some word is accepted. Why are these different arguments?
4. Trace the input-dimension factor a in Lemma 14.8. Identify the normalization that produces it.
5. Verify the two duration inequalities in (14.18) and the baseline identity (14.19). State which resource grows when the duration shrinks.

Chapter 15

Conservation and the cost of keeping a record

A recorder can have enough possible marks and still be unable to produce the right one. Imagine asking whether the first of two cells is occupied and the second is empty. The apparatus must couple to those cells, leave an outcome in a readable pointer, and obey its conservation laws throughout the interaction. These are requirements on one operation. They cannot be satisfied by choosing the record and the dynamics independently.

We will prove a quantitative limit for this question. Suppose the source is two native eight-state cells and the recorder supplies r more such cells, together with any finite neutral memory. Suppose the interaction exactly conserves the specified total clock energy and total occupation, and the final yes/no pointer is a sharp event in the neutral memory. If its yes probability differs from the desired one by at most e for every source state, then

$$e \geq \frac{1}{2(r^2 + 3)}. \quad (15.1)$$

The recorder's cells may be correlated with one another and with its neutral memory. The initial source and the complete recorder must be separate. We will state these conditions precisely before using them.

The proof has three physical steps. Conservation restricts the event that can be read at the end of the interaction. Accuracy bounds the noise of that event. A particular source state then makes the restriction incompatible with arbitrarily small noise unless the recorder supplies sufficient charge spread. The constants in (15.1) come from a four-entry vector and a bound on a sum of r projections.

The conservation argument belongs to the line of work begun by Wigner and Araki and Yanase [13, 14], with quantitative measurement-error bounds developed by Ozawa [15]. The native charges and explicit witness below determine the particular bound here.

15.1 The event, the interaction and the pointer

Work first with an arbitrary finite-dimensional source space $\mathcal{H}_S$ and recorder space $\mathcal{H}_R$. The inner product is conjugate-linear in its first argument. A *sharp event* is an orthogonal projection: an operator P satisfying $P^* = P = P^2$. For a unit source vector x , the desired yes probability is $\langle x, Px \rangle$.

Prepare the recorder in a unit vector $p \in \mathcal{H}_R$. Inserting that preparation gives the isometry

$$J : \mathcal{H}_S \longrightarrow \mathcal{H}_S \otimes \mathcal{H}_R, \quad Jx = x \otimes p.$$

The interaction is a unitary U on the full tensor product. A sharp pointer B is read after the interaction. Its actual yes probability is

$$\langle UJx, BUJx \rangle.$$

Pull this final event back to the initial time:

$$Z = U^*BU, \quad M = J^*ZJ. \quad (15.2)$$

The operator Z is a projection. The compressed operator M is an effect, meaning $0 \leq M \leq I$, but it need not be a projection. Its expectation is exactly the probability the apparatus produces:

$$\langle x, Mx \rangle = \langle UJx, BUJx \rangle.$$

Compression describes the given preparation and interaction. It does not supply a new event that the apparatus is free to choose.

Definition 15.1 (Uniform event accuracy). The recorder has error at most e for the event P if

$$|\langle x, (M - P)x \rangle| \leq e \quad \text{for every unit } x \in \mathcal{H}_S. \quad (15.3)$$

The expression inside the absolute value is real. The preparation p , interaction U and pointer B are fixed across these source inputs.

Uniformity matters. An apparatus designed for one known state need not answer the same question on an unknown state. Our lower bound concerns a single recorder that meets (15.3) for all inputs.

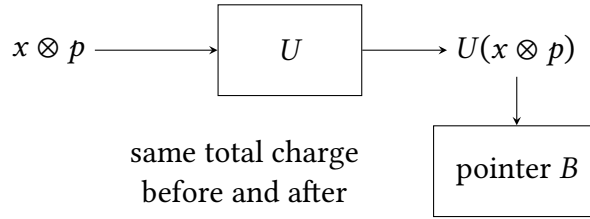


Figure 15.1. The pointer is read after the conserving interaction. Pulling it back gives the event $Z = U^*BU$ on the initial source and recorder.

15.2 Accuracy controls the noise of a sharp event

The discrepancy between the actual event and the ideal one can be measured on the full state, before any compression. Put $Q = P \otimes I$, so that $QJ = JP$, and define the noise operator $E = Z - Q$. On an input Jx , its squared norm is

$$\|EJx\|^2 = \langle x, (P + M - MP - PM)x \rangle. \quad (15.4)$$

Indeed, expand $(Z - Q)^2 = Z + Q - ZQ - QZ$, then use $Z^2 = Z$, $Q^2 = Q$, $J^*J = I$ and $QJ = JP$.

The following identity is what gives the useful constant. Let $D = M - P$. Expansion using $P^2 = P$ yields

$$P + M - MP - PM = (I - P)D(I - P) - PDP. \quad (15.5)$$

Thus the two contributions to the noise involve orthogonal parts of the same input.

Lemma 15.2 (Probability error bounds projection noise). *If (15.3) holds, then*

$$\|(Z - Q)Jx\|^2 \leq e\|x\|^2 \quad \text{for every } x \in \mathcal{H}_S.$$

Proof. Rescaling a nonzero vector in (15.3) gives $|\langle y, Dy \rangle| \leq e\|y\|^2$ for every y ; the zero vector satisfies the same inequality directly. Apply this bound to $(I - P)x$ and Px in (15.5). Then

$$\begin{aligned} \|(Z - Q)Jx\|^2 &\leq e\|(I - P)x\|^2 + e\|Px\|^2 \\ &= e\|x\|^2. \end{aligned}$$

The last equality is the orthogonal decomposition given by P . □

The orthogonal decomposition is essential to this estimate. Bounding the two terms separately by $e\|x\|^2$ would lose a factor of two.

15.3 What exact conservation does to the pulled-back event

Let A and R be Hermitian charge operators on the source and recorder. Their total charge is

$$C = A \otimes I + I \otimes R. \tag{15.6}$$

Suppose the interaction conserves it exactly, $UC = CU$, and the final pointer commutes with it, $BC = CB$. The pulled-back event also commutes:

$$ZC = U^* BUC = U^* BCU = U^* CBU = CZ.$$

A pointer that is neutral for this charge supplies the second commutation condition. It is an actual requirement on the readout.

For Hermitian operators E, C , any vector ξ and any real number c ,

$$\langle \xi, [E, C]\xi \rangle = \langle E\xi, (C - cI)\xi \rangle - \langle (C - cI)\xi, E\xi \rangle.$$

Here $[E, C] = EC - CE$. The scalar c cancels. The triangle and Cauchy–Schwarz inequalities give

$$|\langle \xi, [E, C]\xi \rangle|^2 \leq 4\|E\xi\|^2\|(C - cI)\xi\|^2. \tag{15.7}$$

Take $E = Z - Q$. Since $[Z, C] = 0$, we have $[E, C] = -[Q, C]$. Combining (15.7) with Lemma 15.2, for a unit source vector x , gives

$$|\langle Jx, [Q, C]Jx \rangle|^2 \leq 4e\|(C - cI)Jx\|^2. \tag{15.8}$$

This is the required measurement constraint. A nonzero commutator on the left forces either noise or charge spread on the right.

15.4 The native source and its two charges

We now specify the cells. One cell has basis $e_0, \dots, e_7$. Use the parity operator S and adjacent-pair flip F from Chapter 14. They obey

$$S^2 = F^2 = I, \quad SF = -FS.$$

Consequently iSF is a Hermitian involution, and

$$N = \frac{I - iSF}{2} \tag{15.9}$$

is the occupation projection. The four-step cyclic shift Q_4 sends e_j to $e_{j+4 \bmod 8}$. It is another Hermitian involution. Write

$$P_c = \frac{I - Q_4}{2}. \tag{15.10}$$

It projects onto the odd Fourier modes. We will use its sum across cells as a charge count.

For completeness, let W be the normalized Fourier matrix

$$W_{jk} = \frac{\omega^{jk}}{\sqrt{8}}, \quad \omega = e^{-2\pi i/8},$$

and define the dimensionless clock $K = W \text{diag}(0, 1, \dots, 7) W^*$. The native clock energy is $H_0 = \varepsilon_0 K$, with $\varepsilon_0 > 0$. The value of this positive scale will cancel from the argument. The other conserved native quantity is N .

For any one-cell operator T , let

$$\mathcal{L}(T) = \sum_{j=1}^{r+2} T^{(j)} \otimes I_{\mathcal{H}_0}$$

act on the two source cells, r recorder cells and neutral memory $\mathcal{H}_0$. Superscript (j) means that T acts in slot j and identities act in the other slots.

Lemma 15.3 (The two native charges conserve the count). *If U commutes with $\mathcal{L}(H_0)$ and $\mathcal{L}(N)$, then it commutes with $\mathcal{L}(P_c)$.*

Proof. Since $\varepsilon_0 \neq 0$, energy conservation implies commutation with $\mathcal{L}(K)$. Appendix B gives an explicit finite expression for P_c using I, K, N , complex linear combinations and commutators. It derives that expression from the entries of the specified eight-state matrices.

Operators on different slots commute. Hence $\mathcal{L}$ is linear and

$$\mathcal{L}([T_1, T_2]) = [\mathcal{L}(T_1), \mathcal{L}(T_2)].$$

It sends I to $(r+2)I$, which commutes with U . Commutation with U is preserved by sums, scalar multiples and commutators. Applying these facts in order to the finite expression proves the assertion. $\square$

The count is therefore derived from the two specified conservation laws. It has not been added as a third conservation assumption. The calculation in the appendix is needed for that distinction.

On the two source cells, set

$$A = P_c \otimes I + I \otimes P_c, \quad P = N \otimes (I - N). \quad (15.11)$$

The event P asks precisely whether the first cell is occupied and the second is empty. The count A and the event P do not commute. We next exhibit a source vector that quantifies their incompatibility.

15.5 A four-entry witness inside the actual source

The necessary calculation fits in a two-dimensional subspace of each cell. Define the eight-by-two matrix

$$V = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ -i & i \\ 0 & 0 \\ 0 & 0 \\ -1 & -1 \\ -i & i \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad C_2 = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad N_2 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}. \quad (15.12)$$

Its columns have norm one and are orthogonal. Substitution in (15.9) and (15.10) gives

$$V^*V = I, \quad P_c V = V C_2, \quad N V = V N_2.$$

Thus $V \otimes V$ is an isometric embedding of a four-dimensional space into the actual two-cell source, and it intertwines the source operators with

$$A_2 = C_2 \otimes I + I \otimes C_2, \quad P_2 = N_2 \otimes (I - N_2).$$

These are restrictions of the stated native matrices, with the same inner products and expectations.

Use the coordinate order 00, 01, 10, 11, and let

$$v_a = (i, 0, a, i)^T \quad (a \in \mathbb{R}).$$

Direct multiplication gives

$$\|v_a\|^2 = a^2 + 2, \quad (15.13)$$

$$(A_2 - I)v_a = (a/2, i, i, a/2)^T, \quad (15.14)$$

$$\langle v_a, (A_2 - I)v_a \rangle = 0, \quad (15.15)$$

$$\|(A_2 - I)v_a\|^2 = a^2/2 + 2, \quad (15.16)$$

$$\langle v_a, [P_2, A_2]v_a \rangle = 2ia. \quad (15.17)$$

For example, $P_2 v_a = (0, 0, a, 0)^T$. The two terms in the last commutator expectation are $a^2 + ia$ and $a^2 - ia$, explaining their difference. In (15.15), the four contributions are $-ia/2, 0, ia, -ia/2$ and cancel.

Choose $a = \sqrt{2}$ and return to the source:

$$\psi = \frac{1}{2}(V \otimes V)v_{\sqrt{2}}.$$

Equations (15.13)–(15.17) then give

$$\begin{aligned} \|\psi\| &= 1, & \langle \psi, (A - I)\psi \rangle &= 0, \\ \|(A - I)\psi\|^2 &= \frac{3}{4}, & |\langle \psi, [P, A]\psi \rangle|^2 &= \frac{1}{2}. \end{aligned} \tag{15.18}$$

This one vector will suffice. We need no assumption that every source state has a large commutator expectation.

15.6 How much charge spread can the recorder supply?

Let the recorder consist of r eight-state cells and a finite neutral memory. Its space and count are

$$\mathcal{H}_R = (\mathbb{C}^8)^{\otimes r} \otimes \mathcal{H}_0, \quad R = \sum_{j=1}^r P_c^{(j)} \otimes I_{\mathcal{H}_0}.$$

No product decomposition of $p \in \mathcal{H}_R$ is assumed. Its cells and memory may share arbitrary correlations.

Lemma 15.4 (A projection-sum budget). *For any r orthogonal projections $R_1, \dots, R_r$ on the same space and any unit vector p ,*

$$\left\| \left(\sum_{j=1}^r R_j - \frac{r}{2} I \right) p \right\|^2 \leq \frac{r^2}{4}.$$

The projections need not commute with one another.

Proof. For each projection, $(R_j - I/2)^2 = I/4$. Hence $\|(R_j - I/2)p\| = \|p\|/2 = 1/2$. The triangle inequality gives

$$\left\| \sum_{j=1}^r (R_j - I/2)p \right\| \leq \sum_{j=1}^r \|(R_j - I/2)p\| = r/2.$$

Squaring proves the claim. The empty sum also satisfies it. $\square$

Applied to the recorder count, this proves

$$\|(R - rI/2)p\|^2 \leq r^2/4. \tag{15.19}$$

The estimate already allows coherent and correlated recorder states. It has the quadratic scale required by that freedom. An independent-cell variance calculation would be a different premise and cannot be inserted here.

The complete initial state is $\psi \otimes p$. For $C = A \otimes I + I \otimes R$, we have

$$(C - (1 + b)I)(\psi \otimes p) = (A - I)\psi \otimes p + \psi \otimes (R - bI)p.$$

The two terms are orthogonal because $\langle \psi, (A - I)\psi \rangle = 0$. The source is a separate tensor factor from the whole recorder, even when the recorder is internally correlated. Using (15.18),

$$\|(C - (1 + b)I)(\psi \otimes p)\|^2 = \frac{3}{4} + \|(R - bI)p\|^2. \quad (15.20)$$

Also, $[P \otimes I, C] = [P, A] \otimes I$. Thus the left side of (15.8), on this input, is exactly $1/2$. We obtain the state-dependent inequality

$$\frac{1}{2} \leq e(3 + 4\|(R - bI)p\|^2) \quad \text{for every real } b. \quad (15.21)$$

15.7 The two-charge recorder theorem

We can now collect the physical requirements in one statement. A neutral memory carries the identity action of both native charges: it supplies storage and a pointer, but contributes no charge term.

Theorem 15.5 (Finite native recorder bound). *Let the source be two native cells, with the event $P = N \otimes (I - N)$. Let the recorder consist of r native cells and an arbitrary finite neutral memory $\mathcal{H}_0$. Assume:*

1. *The initial state is $x \otimes p$, where p is one fixed unit recorder vector and x ranges over every unit source vector.*
2. *The interaction U is unitary on the whole source–recorder space and exactly conserves both $\mathcal{L}(H_0)$ and $\mathcal{L}(N)$.*
3. *The final pointer is $B = I_{\text{source}} \otimes I_{\text{recorder cells}} \otimes D$, where $D = D^* = D^2$ acts on the neutral memory.*
4. *For every unit source vector x ,*

$$|\langle U(x \otimes p), BU(x \otimes p) \rangle - \langle x, Px \rangle| \leq e.$$

Then

$$e \geq \frac{1}{2(r^2 + 3)}.$$

Proof. Lemma 15.3 shows that U commutes with the total count C . The pointer B commutes with C because they act in different tensor factors. Thus $Z = U^*BU$ is a projection commuting with C . Its compression $M = J^*ZJ$ satisfies the uniform event bound.

Apply (15.21) with $b = r/2$, then Lemma 15.4. Accuracy implies $e \geq 0$, so the resulting inequalities have the indicated direction:

$$\frac{1}{2} \leq e(3 + 4\|(R - rI/2)p\|^2) \leq e(3 + r^2).$$

Divide by the positive number $r^2 + 3$. □

For every finite r , the lower bound is positive. An exact recorder of this event cannot satisfy all four conditions. A larger neutral memory alone does not remove the bound: its dimension never enters the charge budget. Increasing the number of charged cells relaxes the obstruction, but this inequality alone does not construct a recorder attaining it.

15.8 What the resource is, and what the result establishes

The resource appearing in the proof is the recorder's spread in the conserved count. Equation (15.21) retains the actual prepared state; (15.1) replaces that spread by a bound depending only on the number of native cells. The reference cells can exchange this count with the source while the total remains fixed. The neutral pointer reports an outcome without supplying that exchange itself.

The argument belongs to the conservation-law constraints on measurement associated with Wigner, Araki and Yanase. Its specific content here is the native realization: two stated charges imply the count conservation, an explicit embedded source vector fixes the constants, and the projection budget allows arbitrary internal recorder correlations. The statement is about uniform probability accuracy for one specified event. It is not a claim that every task has the same error bound or that the displayed constant is attainable by a conserving apparatus.

There is no conflict with the controlled constructions of Chapter 14. That chapter permits selectable pulses and records their positive energy operators. The theorem here requires an interaction conserving two exact total charges and a neutral sharp pointer. The control supply belongs in the complete physical account when those conditions are imposed. An operation that can be described and approximated with supplied controls need not be available to a closed recorder with a fixed finite charge resource.

The result sharpens the question raised by finite readout. We must ask both which distinctions a record can carry and which interactions can produce it. A physical construction answers the second question only after its preparation, conserved quantities, reference supply and readout have been specified together.

Exercises

1. Expand $(Z - Q)^2$ and derive (15.4). Identify every use of idempotence.
2. Verify (15.5). Explain why the two squared norms in Lemma 15.2 add to $\|x\|^2$.
3. Multiply the matrices in (15.12) to check $V^*V = I$, $P_c V = V C_2$ and $NV = V N_2$. Then check the four-entry witness.
4. Derive (15.20). Locate the exact step that uses the initial separation between source and recorder.
5. Trace where each of the four assumptions of Theorem 15.5 enters its proof. Which part of the proof permits correlations inside the recorder?

Chapter 16

What has been forced

Return to the first tally. One arrival leaves a mark; another arrival adds another. We wanted to know what survives when the marks change their shape, their material or their names. We now have several answers, because the question contains several requirements.

If a record preserves every finite step, its arithmetic structure is fixed up to the unique map that preserves the start and the step. If a reader identifies some records, the identifications must be studied as part of that reader. If a procedure answers a question at every requested precision, one rule must cover all those requests. And if an apparatus produces the record, its interaction must actually carry the required information into the available output.

These conclusions were reached by different arguments. Keeping the arguments connected, with their conditions intact, tells us what is unavoidable and what a further construction must provide.

16.1 The arithmetic structure is unique

The starting construction has two clauses: a start, and one further step. Every record is finite and has been obtained by those clauses. Structural induction follows that construction. It supplies the first proofs, including the facts that a step is never the start and that equal stepped expressions have equal predecessors. In the arithmetic language of Part II, each of these two facts, and induction, supplies statements that the other rules cannot (Section 5.7).

Now give any system a designated start and a step operation. There is exactly one map from the finite distinction orbit into that system that preserves those two operations. Theorem 2.3 constructs it: send the start to the designated start, then follow each step. Uniqueness comes from the same finite construction.

That map need not preserve every distinction. A clock that returns to a previous state is one possible target. Faithfulness and generation are the conditions that turn the unique map into an isomorphism. The target step must be injective, its start must not be a step, and every target element must be generated from the start. Under those conditions, no two finite records merge and no target element is omitted. This is Theorem 2.7.

Addition and multiplication are then fixed by their recursive clauses. Their algebraic laws follow by induction; they were not separately chosen to make the tally resemble familiar arithmetic. Conversely, a commutative semiring structure on the orbit in which adding its unit is the step must be the constructed one, and the orbit maps uniquely into every commutative semiring (Section 3.4). The reading into ordinary natural numbers preserves those operations and reflects equality. Signed differences and rational ratios extend the construction through explicit equivalence relations. Their operations are legitimate because changing a representative leaves the result in the same class. Each extension is also determined by its task (Section 4.9). The signed records complete the orbit's addition; the constructed

integers map uniquely into every commutative ring, and the constructed rationals into every field of characteristic zero; and each carries exactly one order compatible with its arithmetic. The line built from ratios in Chapter 12 is determined by its task in the same way, once countable choice is granted: every Archimedean ordered field maps into it in exactly one way that preserves the operations and the order (Theorem 12.24).

The necessity here has a precise form. A step system is faithful and generated exactly when its canonical reading is a bijection (Proposition 2.8), so any faithful generated tally has the same structure. Any operation satisfying the stated recursive clauses agrees with the constructed one; so does any associative operation for which adding 0 changes nothing and adding 1 takes one step (Theorem 3.10). The argument uses finite expressions, equality and induction in the mathematics with which the construction is studied. That induction ranges over all properties. Restricted to the formulas of an arithmetic language, it admits systems with elements that no tally reaches. Naming those resources makes the uniqueness claim exact.

16.2 Identifying records and extending logic are visible choices

A reader may retain less than the full tally. Chapter 7 describes identifications compatible with addition. Once a decidable congruence comes with an exhibited collision between distinct records, its form is forced: an initial segment followed by a repeating period. The classification proves that every collision has this form and that the least period controls all later identifications.

Changing the information supplied with the congruence, or the answer demanded of it, changes the price, and each price is exact. With a decision procedure and an exhibited collision the classification costs nothing. Replacing the collision by the assertion that the congruence is not equality costs Markov's principle. Keeping only the decision procedure, the parity of the index costs LLPO, the question whether the congruence is equality costs WLPO, and the full alternative costs LPO. With no decision procedure the alternative costs excluded middle. Theorems 7.6, 7.7, 7.9 and 7.10 prove these rows, and Theorem 7.11 gives the same prices for readers. In each case the reverse construction matters: the catalogue itself recovers the additional principle. These are equivalences, not merely observations that one familiar proof happens to use a stronger rule. The six prices are also different from one another. For each pair of principles, a model of intuitionistic second-order arithmetic satisfies one and refutes the other, and in each model every version of the classification has the truth value of its principle (Section 7.10). That system does not prove Markov's principle, so the classification from negative nontriviality genuinely needs it. Among the index-period congruences, exactly those of index zero extend to the signed records, each in one way, and the prices do not change there, provided the parity question is asked of the period: every index is zero, so its parity costs nothing, while the parity of the period costs exactly LLPO (Section 7.8).

The proof ledger of Chapter 6 makes a related distinction at the level of derivations. It records which of the specified additional rules occur, and combination retains every resource used by a premise. Soundness states which principles suffice to interpret the result, and interpreting any derivation needs at most the limited principle of omniscience in the metatheory, applied to the predicates that formulas define; for the derivations with an EM or LPO entry, taken

together, that is also the least assumption that serves. The rules themselves have exact prices. With induction on every formula, every instance of the restricted MP rule has a posit-free derivation, because a quantifier-free formula is eventually constant in its searched variable, and the LPO rule, which applies to every formula, derives every instance of excluded middle. The three names then mark only two strengths. Without induction, or with induction on quantifier-free formulas only, the LPO and MP rules are independent and EM is equivalent to the two together. The induction entry stays part of the accounting because these prices depend on it. The collapse concerns these arithmetic rules, not the principles about arbitrary decidable predicates priced in Chapters 7 and 12.

Chapter 9 goes further by separating rules of repetition inside a specified language. Repeating a fixed update, performing an indexed recursion and eliminating into an index-dependent family are different resources there. The countermodels satisfy the weaker rules while failing the proposed stronger judgment. The additional start discriminator has its own separating argument. These concrete models establish the strict boundaries in Theorem 9.10; the boundaries are attached to that calculus and its expressible judgments. The strongest fragment, $\mathcal{E}^+$, is the one compared with Peano arithmetic in Section 16.3.

Changing notation does not remove a logical cost. Where a translation is claimed to preserve a result, the translation must carry the relevant operations, assignments and quantifiers. Chapter 8 does this for numbers and lists of one mark. Every arithmetic formula has the same truth under related assignments. The proof transports witnesses in one direction and back in the other; the two round trips are what make the quantified statement possible.

With only one round trip, less transfers, and Theorem 8.8 with Proposition 8.9 says exactly what. Inside any structure obeying the six arithmetic rules the numerals form a copy of the numbers: quantifier-free statements transfer in both directions, purely existential statements go up and purely universal statements come down. A structure with two looping points shows that no other quantifier prefix transfers: a prefix containing a universal quantifier does not go up, and one containing an existential quantifier does not come down. The statement concerns all the formulas with a given prefix; a single universal sentence such as $\forall x(x + 0 = x)$ goes up because it is one of the six rules. The derivations can also be compared with an independent proof system. With excluded middle allowed and LPO and MP excluded, the rules of Chapter 5 prove exactly the theorems of Peano arithmetic, from every finite context (Theorem 8.11). With no added principle they prove exactly the theorems of Heyting arithmetic, and the posit-free derivations are the translations of its derivations, up to the way the injectivity of the successor is written (Corollary 8.12).

16.3 Peano arithmetic and a richer language

The first question a mathematician asks of these rules has a complete answer: with excluded middle they are Peano arithmetic, the system most mathematicians mean by arithmetic. A second question follows. Could a richer language, one that names functions and functions of functions, prove facts about numbers that those rules cannot? For sentences built from $=$, $\perp$, $\rightarrow$ and $\forall$, it cannot. Every such sentence that the richer language proves has a derivation by the rules of Chapter 5 with no added principle, and the strongest typed fragment of Chapter 9 proves the translations of exactly the theorems of Peano arithmetic. The proof runs the functions as programs. What lets the arithmetic rules follow those runs is a fixed test program whose correctness the rules themselves derive.

The classical derivations are Peano arithmetic

Chapter 8 presents first-order *Peano arithmetic*, abbreviated PA, as a classical calculus of its own over the formulas of Definition 5.1: six axioms for the start, the step, addition and multiplication, the rules of natural deduction with equality, induction for every formula, and excluded middle. It keeps no ledger. Write $\Gamma \vdash_{\text{PA}} \varphi$ when it derives φ from Γ . By Theorem 8.11, for every finite context Γ and every formula φ , $\Gamma \vdash_{\text{PA}} \varphi$ holds exactly when the rules of Chapters 5 and 6 derive φ from Γ by a derivation whose ledger records neither LPO nor MP.

The theorem places a familiar system inside the accounting of Chapter 6. Peano arithmetic is what the rules prove when the ledger may record EM and quantified induction but neither of the other two principles. The theorem compares two proof systems; it does not prove the consistency of either. The rest of this section uses it to compare a richer language with both.

A language of functions

The calculus of Chapter 9 can name more than numbers. Use its types built from O by forming function types $A \rightarrow B$, and its terms built from variables, 0 , successor, λ -abstraction, application, the iterator I and the indexed recursor R . Over these terms we use a logic of finite types. Its formulas are truth, falsehood, equations $t =_A u$ at every such type A , conjunction, implication, and universal quantification over every such type. Its nineteen rules are: use of a hypothesis and of a list of hypotheses; truth; the rule from falsehood; introduction and elimination for implication and the universal quantifier, and introduction and the two eliminations for conjunction; reflexivity; computation by the equations of the calculus; replacement of equals in a formula; the two successor laws $\neg(S(x) = 0)$ and $S(x) = S(y) \rightarrow x = y$; function extensionality, which infers $f =_{A \rightarrow B} g$ from $\forall x: A (f(x) =_B g(x))$; induction on O for every formula, with parameters; and substitution for variables. Addition and multiplication are terms of this language, defined by iteration:

$$x + y = I(O, x, \lambda z. S(z), y), \quad x \cdot y = I(O, 0, \lambda z. z + x, y).$$

Every formula of the arithmetic language that uses only $=$, $\perp$, $\wedge$, $\rightarrow$ and $\forall$ is therefore also a formula of the finite-type logic. Systems of this kind are known as extensional arithmetic in all finite types [5].

Running the functions

How can rules that speak only about numbers check a proof that quantifies over functions? By running the functions. Each term of the finite-type language is compiled into a program expression of Definition 10.5. A function is represented by the code of its program, which is a number, and it is applied through one fixed *universal program* $\mathbf{U}$. On an input $\langle e, n \rangle$, the program $\mathbf{U}$ searches $c = 0, 1, 2, \dots$ for a number that a fixed *record test* accepts as a *computation record* of program e on input n , and returns the output written in that record. A computation record is a finite table of evaluation facts, each justified through one of the evaluation clauses of the program expressions by facts that stand later in the table, together with the values that each justification uses. The record test is itself a program; on every input it returns 0, to accept, or 1, to reject.

Formulas of the finite-type logic are read as arithmetic formulas about these programs. At type O , equality is equality of numbers. At a function type, two codes are equal when, on equal inputs, both programs return outputs and those outputs are equal at the output type. This relation is called *hereditary extensional equality*. A universal quantifier over a type ranges over the numbers that are equal to themselves in this sense. An equation $t =_A u$ becomes the statement that the programs compiled from t and u return values and that those values are hereditarily equal at A . Truth, falsehood, conjunction and implication keep their meaning.

Every compiled application runs through $\mathbf{U}$, and so does every step of a compiled iteration. The compiled proofs therefore need one property of $\mathbf{U}$ as a theorem of the arithmetic rules themselves. It follows from two facts about the record test:

Acceptance. For every computation of program e on input n , the test accepts a record of that computation.

Record soundness. Whatever the test accepts is a genuine computation record. In particular, if it accepts a record for program e on input n , then e on input n returns the output written in that record.

Programs are numbers through the coding (10.4), and inputs, outputs and records are numbers as well; each fact is a sentence of the arithmetic language about these numbers. Both facts are derived inside the rules of Chapter 5 with no added principle (Theorem 16.1). The derivations use induction on quantified formulas, which the ledger records in its induction entry, but none of EM, LPO or MP. They are possible because of how the record and its test are built.

A record that carries its justifications

Most evaluation clauses mention values that the evaluation fact itself does not show. A composition of f after g returns w on m because g returns some value a on m and f returns w on a . A test that had to find a would search inside the check of a single fact, and a derivation about the test would have to follow that inner search as well. The record writes such values down instead.

A record of program e on input n therefore has two tables. Each row of the first states an evaluation fact: with allowance k , the program with code p returns w on input m . The allowance is a budget for the evaluation: it bounds the input, and each further step of a

recursion or a search uses up one unit of it. The second table gives, for each row, the few values on which that row's step depends: the predecessor s of its allowance, the codes of the two parts of its program, the two components of a paired value, an intermediate output, and the positions of the rows it relies on. A clause that needs fewer of these values ignores the rest. The record also states the length of the tables, the position of the row for e on n , and that row's allowance and output; pairing packs everything into one number, with the output last.

The test checks each row by a fixed, flat combination of equality and order tests. The allowance must be $s + 1$, with $m \leq s$; the later steps of a recursion or a search run with allowance s . Then one of ten cases must hold: one for each of the eight constructors of Definition 10.5, with recursion split into its start and its later steps, and search into stopping and continuing. In the composition case the test checks that p equals $4\langle u, v \rangle + 6$ for the two recorded part codes, that the first named row states that the program with code v returns the recorded value a on m , that the second states that the program with code u returns w on a , both with allowance k , and that both named rows stand later in the table. Checking a named row is a comparison of numbers. The test searches for nothing inside a row. Its only search is the bounded one over the positions of the table. Finally it checks that the row for e on n lies within the table and carries the output written last.

Read as an actual computation, a row has a passing justification exactly when it obeys its evaluation clause, so $\mathbf{U}$ on input $\langle e, n \rangle$ returns exactly the outputs of e on n . The compiled proofs need more than that truth: derivations by the rules.

Theorem 16.1 (The record test is correct). *Acceptance and Record soundness have posit-free derivations by the rules of Chapter 5.*

Outline. The derivation has five parts. The first four give the two facts.

1. Let arbitrary programs supply the two tables and their length. At every position the row check returns a value, and the value is positive exactly when the justification stored for that row passes. Each equality test, order test and comparison with a named row has these two properties, and conjunction and disjunction preserve them.
2. A passing justification gives back the row's evaluation clause in the form that only says the supporting rows exist later in the table: forget the named positions. Conversely, every row that obeys its clause in that form has a passing justification: open the clause's existential statements, record the values they supply, and pack them by pairing.
3. If every row of a table obeys its clause, the second part gives each row a justification. A finite choice, proved by induction on the length of the table and extending it one entry at a time, collects them into one justification table. It is a derived statement, not an added principle.
4. Given a computation of e on n , pack its table, its length, the justification table of the third part, the position of its row for e on n , that row's allowance and the output into one number c . Every stored justification passes, so by the first part every row check is positive; the bounded search over positions carries this to the whole table, and the row for e on n passes the final check by construction. The test therefore returns 0 on $\langle \langle e, n \rangle, c \rangle$: it accepts c as a record for e on n . This is Acceptance. Conversely, let the test

accept c as a record for e on n , and unpack c . The bounded search and the first part show that every stored justification passes, and the second part that every row obeys its clause. The table is therefore a computation, and its row for e on n carries the output written last in c . This is Record soundness.

The fifth part is universal application, the lemma that follows. The parts use induction, case analysis and pairing; none of them uses EM, LPO or MP. $\square$

Lemma 16.2 (Universal application). *The rules of Chapter 5 give a posit-free derivation of the following statement, which we call universal application: for all e, n, y , if e on input n returns y , then $\mathbf{U}$ on input $\langle e, n \rangle$ returns y .*

Proof. Let e on input n return y . Acceptance gives a record c of that computation that the test accepts. The search in $\mathbf{U}$ examines the numbers up to c one at a time, and each test returns 0 or 1, so the search stops at the first accepted $d \leq c$. Record soundness says that e on n returns the output written in d . A program returns at most one value on a given input, so that output is y , and $\mathbf{U}$ returns it. Theorem 16.1 derives the two facts, and each remaining step is an induction or a finite case analysis with no added principle. $\square$

From proofs to derivations

Theorem 16.3 (Proofs compile to derivations). *For every proof in the finite-type logic of a closed formula P , the rules of Chapter 5 derive the reading of P by a posit-free derivation.*

Outline. Induct on the proof, one case for each of the nineteen rules, proving the corresponding statement for every context and every environment of values. The logical rules become the same rules applied to the read formulas. Reflexivity uses the fact that the program of a well-typed term returns a value hereditarily equal to itself; computation uses the fact that convertible terms have programs with hereditarily equal values; replacement and substitution use the fact that read formulas transfer between hereditarily equal environments. The successor laws are rules of Chapter 5. Extensionality needs no new principle, since the reading of $f =_{A \rightarrow B} g$ already states pointwise agreement. Induction becomes the rule (5.2) applied to the read formula; that formula contains quantifiers, so the ledger records the induction entry and nothing else. Every case that runs an application or an iteration uses universal application, Lemma 16.2. $\square$

Theorem 16.4 (Arithmetic is recovered from its reading). *For every closed formula A of the arithmetic language built from $=$, $\perp$, $\rightarrow$ and $\forall$, a posit-free derivation proves that the reading of A implies A .*

Outline. Induct on A , proving for every assignment both that the reading implies the formula and that the formula implies the reading; the implication case needs both directions for its antecedent. At an equation, the programs compiled from its two terms return exactly the values of those terms. For addition and multiplication this is proved by induction along the iteration that defines them; each step of that iteration runs through $\mathbf{U}$ and uses Lemma 16.2. At a universal quantifier over O , every number lies in the range of the quantifier. $\square$

Theorem 16.5 (Conservativity for negative sentences). *Let A be a closed formula of the arithmetic language built from $=$, $\perp$, $\rightarrow$ and $\forall$. If the finite-type logic proves A , then the rules of Chapter 5 derive A by a posit-free derivation.*

Proof. Theorem 16.3 gives a posit-free derivation of the reading of A , and Theorem 16.4 a posit-free derivation that the reading implies A . Implication elimination combines them. By Proposition 6.4, the combined ledger records none of the three principles. $\square$

The comparison with Peano arithmetic

The restriction to $=$, $\perp$, $\rightarrow$ and $\forall$ loses nothing for classical arithmetic. The *negative translation* φ^N of a formula doubly negates each equation and replaces, at every level, $\exists x\psi$ by $\neg\forall x\neg\psi$, $\psi \vee \chi$ by $\neg(\neg\psi \wedge \neg\chi)$, and every conjunction, including those the previous clause introduces, $\psi \wedge \chi$ by $\neg(\psi \rightarrow \neg\chi)$; it keeps $\perp$, $\rightarrow$ and $\forall$. The result uses only $=$, $\perp$, $\rightarrow$ and $\forall$. With excluded middle, φ^N and φ imply each other [4]: the rules of Chapters 5 and 6 derive each from the other by derivations whose ledger records only EM.

In the calculus $\mathcal{E}^+$ of Chapter 9, a formula of this kind is expressed by a type: an equation by an identity type on O , $\perp$ by the empty type, an implication by a function type and $\forall x$ by a product over O . We say that $\mathcal{E}^+$ proves φ^N when it has a closed term of that type.

Theorem 16.6 (Peano arithmetic and the dependent calculus). *Let φ be a sentence of the arithmetic language.*

1. *If $\vdash_{\text{PA}} \varphi$, then $\mathcal{E}^+$ proves φ^N .*
2. *If $\mathcal{E}^+$ proves φ^N , then $\vdash_{\text{PA}} \varphi$.*

Outline. For the first assertion, Theorem 8.11 gives a derivation of φ by the rules of Chapters 5 and 6 whose ledger records neither LPO nor MP. Translate it node by node into a term whose type expresses the translated conclusion. Addition and multiplication are the iteration terms displayed above, and their defining equations hold by computation. Equality and its substitution use identity types and their eliminator. The law $\neg(S(t) = 0)$ uses the start discriminator, as in Theorem 9.3; successor injectivity uses the predecessor of Theorem 9.2; induction uses the dependent eliminator. The translated EM instances and the translated logical rules need no added rule, because every translated formula is stable under double negation.

For the second assertion, read the closed term as a proof in the finite-type logic: identity elimination becomes replacement, the dependent eliminator becomes induction, the discriminator becomes the first successor law, and functions become implications and universal statements. Theorem 16.5 then gives a posit-free derivation of φ^N by the rules of Chapter 5. With the EM rule, φ^N implies φ . The combined derivation of φ records at most EM and induction, neither LPO nor MP, and Theorem 8.11 converts it into a Peano derivation. $\square$

The first assertion uses all three resources separated in Theorem 9.10; the last question at the end of this chapter shows that it fails without the discriminator. Together, the two assertions say that $\mathcal{E}^+$ proves the translations of exactly the sentences that Peano arithmetic proves.

A small case

What does conservativity say in practice? Consider a proof that passes through all functions on O . In the finite-type logic, prove

$$L : \quad \forall f:O \rightarrow O \quad \forall x \quad \forall n \quad l(O, f(x), f, n) = f(l(O, x, f, n)).$$

Starting one step later is the same as taking one more step at the end. Induct on n . At 0, both sides compute to $f(x)$. If the equation holds at n , then

$$l(O, f(x), f, Sn) \equiv f(l(O, f(x), f, n)) = f(f(l(O, x, f, n))) \equiv f(l(O, x, f, Sn)),$$

by computation, replacement with the induction hypothesis, and computation. Now instantiate f by the successor function $\lambda z. S(z)$. Addition is iteration of that function, so computation turns the instance into

$$B : \quad \forall x \quad \forall n \quad S(x) + n = S(x + n).$$

One induction proved a fact about every function, and the arithmetic fact is one of its instances. The sentence B is built from $=$ and $\forall$.

Theorem 16.5 says that this kind of route never reaches a closed negative sentence beyond the posit-free derivations. For B the direct derivation is visible: Lemma 3.2 proved B by induction on n , and the same induction, written with the rules of Chapter 5, uses the quantifier-free formula $S(x) + n = S(x + n)$ and has an empty ledger.

The compiled proof takes a different path. Its reading of L says: for every number f that is hereditarily equal to itself at type $O \rightarrow O$, and for all numbers x and n , the programs compiled from the two sides return values, and those values are equal. The source induction was on an equation. The compiled induction is on the reading of that equation, which asserts that values exist, so it inducts on a formula with quantifiers and the ledger records the induction entry. The instantiation must show that the code of $\lambda z. S(z)$ is hereditarily equal to itself; applying that code runs through $\mathbf{U}$, which is where universal application enters. Recovery then shows that the programs compiled from $S(x) + n$ and $S(x + n)$ return the actual numbers, and so turns the reading of B into B .

The point generalizes. A proof about numbers may pass through functions. The passage never yields a closed negative statement about numbers that the arithmetic rules could not already derive without an added principle.

16.4 Finite access and a uniform construction differ

Every finite tally has its own finite stage. Compatible readings of these stages assemble into one reading of the whole orbit, uniquely. For a given input, the proof uses only that input's stage. The compatibility hypothesis ensures that increasing the budget cannot change an answer already given.

A program is also a finite object. Chapter 10 stores its expression as a tally and recovers the identical expression. Execution is therefore preserved, including the inputs on which execution is undefined. The recovery law does not require a termination claim.

For an interval protocol, a uniform name has a further obligation: one program must return the correct endpoint pair at every precision. The existence of such a name is equivalent to computability of that endpoint function. A supplied finite table of endpoints always has a lookup program, but the table may change with the requested budget. The quantifiers are different:

$$\forall N \exists c \forall n < N \quad \text{and} \quad \exists c \forall n.$$

Here the suppressed conclusion is the same correct endpoint response in both expressions. Theorem 11.11 proves, in the stated classical real display, that the first requirement does not imply the second. Each finite program determines at most one protocol and hence at most one displayed value. Countably many finite programs cannot name every value in that display.

This leaves the enclosure construction intact. Nested rational intervals with the given width bound determine a unique classical value; overlapping at every precision is exactly equality of that value. The distinction is between this mathematical family and an effective rule producing its members. The completeness and classical comparison used for the display are stated where they enter the construction. Chapter 12 prices that completeness on a line that does not presuppose it.

Exact decisions reveal another boundary. For the regular dyadic presentations of Chapter 12, deciding exact zero is equivalent to WLPO. A witnessed positive-or-zero-or-negative answer is equivalent to LPO. The weaker nonnegative-or-nonpositive answer is equivalent to LLPO. Turning a proof that a quantity is not zero into a sign is equivalent to Markov's principle, and the comparisons themselves recover LPO as WLPO together with that principle. These principles already priced the recognition catalogue in Chapter 7. The same prices hold on the line built from the ratios of Chapter 4, which is one continuum with the dyadic presentations. Its completeness comes apart in the same way. The limit of a Cauchy sequence costs only a choice of representatives, and nothing at all as a located cut. That choice decides nothing: intuitionistic arithmetic with variables for sequences and countable choice for numbers and for sequences proves none of LPO, WLPO, LLPO and MP_{bin} (Theorem 12.22). Excluded middle cannot replace it: with excluded middle and choice only for relations whose witnesses a search can find, neither the Cauchy limit nor the passage from a located cut to a point is provable (Theorem 12.27). Monotone convergence costs LPO, a least upper bound of a bounded sequence WLPO, and an exact root or an attained maximum of a uniformly continuous function LLPO, while the approximate versions cost nothing. The proofs give explicit encodings in both directions. A sequence of increasingly accurate rational answers does not erase these distinctions.

16.5 The physical bridge has a construction and a constraint

The first physical question is about the information a reader actually receives. If two situations produce the same record, an exact reader must give them the same required answer. This collision argument gives the capacity bound in Chapter 13. A bounded adaptive test tree still produces a bounded transcript, because its branches and stopping rules are included in the construction of that transcript. Allowing an error tolerance changes exact distinction into

separation by more than twice that tolerance.

These bounds identify what a record must retain. Chapter 14 constructs an operation that returns a quantum system together with its mark. Positivity and normalization supply branch matrices. Stacking those matrices gives an isometry, and the explicit block dilation extends it to a unitary on the whole stated space. A complete reader retains both the intended rows and their complement. The error estimate includes any finite retained reference, so correlations with a system outside the apparatus are part of the comparison.

Two results then connect the descriptions to finite operations. First, ordinary effective matrix names and effective operation-word names can be translated uniformly for fixed dimensions. The search accepts a rational certificate with a strict margin; density ensures that an acceptable word eventually appears. Second, native controls realize the required approximation with positive full energy operators and an explicit scalar action contribution. Input preparation, selectable control pulses, the fixed background holding and the complete reader are all retained in the construction.

The recorder theorem supplies a complementary restriction. For the two-cell event specified in Chapter 15, a whole unitary that conserves both native charges and is followed by a neutral sharp pointer obeys

$$e \geq \frac{1}{2(r^2 + 3)}.$$

Here r counts the recorder's native charge-carrying cells. Its internal correlations are allowed, while its fixed initial preparation is separate from the unknown source. The proof derives the count conservation from the two native charges, bounds noise by uniform event error, and evaluates an explicit source vector. The size of neutral memory cannot remove the charge bound.

Both results concern actual operations under their stated physical conditions. Supplied controls and a closed conserving recorder impose different requirements. An apparatus meeting both descriptions would have to include its control supply in the conserving account. The construction does not assert such an autonomous implementation, and the lower bound does not construct an apparatus attaining it. Those are specific tasks left to further construction.

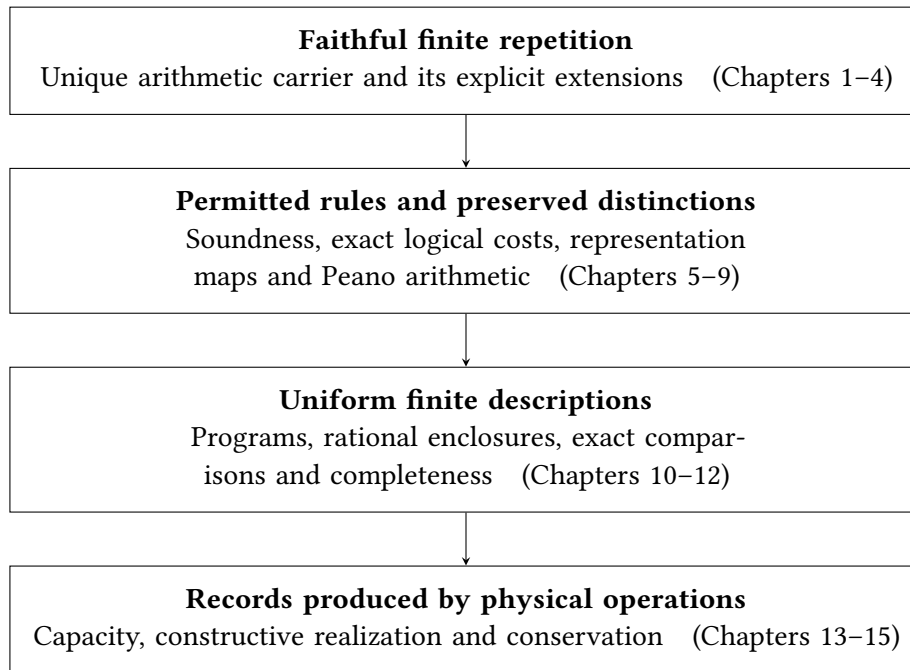


Figure 16.1. A route through the argument. At each stage, the next question requires additional specified objects: a language and its rules, an effective description, or a physical apparatus. The arrows organize the reading; they do not discharge those additional hypotheses.

16.6 What a further claim must show

The book gives several ways to make a necessity claim checkable. Construct the map and prove it unique. Classify every allowed identification. Recover the proposed additional principle from the claim itself. Exhibit a model that satisfies the weaker rules but cannot satisfy the stronger judgment. Compile the proofs of one system into another, then translate their conclusions back. Build the physical operation, then carry its errors and resources through the complete reader.

These methods specify the next work as well. A broader representation claim needs translations preserving the additional structure it names. The comparison with Peano arithmetic in Chapter 8 is one such translation, for a second proof system over the same language. Section 16.3 carries the method to a language of functions, where even the test that runs the compiled programs has its correctness derived by the rules. A uniform construction needs one procedure with the required behavior on every input. A stronger physical realization needs the preparation, controller and reader it assumes to be constructed within the same resource account. Each question can be pursued without replacing the arithmetic, classification and resource results already proved.

The first tally therefore remains visible at the end of the argument. Its start and step determine a finite construction. Faithfulness determines what must remain distinct. A reader determines what can be identified. A proof exposes the rules needed to pass from those objects to a conclusion. A physical apparatus must enact the required transformation and retain its record. The resulting structure can be followed, checked and used at each of those levels.

Questions to work through

1. Follow the map in Theorem 2.7. Which hypothesis gives injectivity, and which gives surjectivity?
2. Compare the WLPO, LPO and LLPO forms of the recognition catalogue in Theorem 7.9 with Theorems 12.6–12.8, and its Markov form, Theorem 7.7, with Theorem 12.11. For each principle, name the data supplied to the forward construction and the object used in the reverse implication in both settings.
3. Explain how a program can have faithful finite storage while failing to halt on an input. Then explain why a uniform interval name must halt at every precision request.
4. Locate one retained premise in each of the three physical chapters: a condition on the record, a condition on supplied controls, and a condition on conservation. Describe the exact proof step that uses each one.
5. In the small case at the end of Section 16.3, the proof of L inducts on an equation, while its compiled form inducts on the reading of that equation. Identify the quantifiers in that reading, and state which ledger entries the compiled derivation records.
6. Why are the comparison theorems stated for formulas built from $=$, $\perp$, $\rightarrow$ and $\forall$? Suppose the finite-type logic had an existential quantifier, read in the same way as the other connectives. Describe what the reading of $\exists y \psi(y)$ would ask a compiled proof to supply, and compare it with what Theorem 16.4 has to recover.
7. Show that the first assertion of Theorem 16.6 fails for $\mathcal{E}^-$. Write the type that expresses the translation of $\forall x \neg(S(x) = 0)$, and compute its value under the valuation of Theorem 9.8.

Appendix A

Finite controls and approximation

Chapter 14 uses two approximation results. A finite alphabet of gates supplies descriptions that can be searched. A family of native rotations and couplings supplies the operations used in the positive-energy construction. This appendix proves both results. The common step is to isolate elementary matrix directions and then connect them across a register.

The argument has four stages. We first identify a sufficient collection of matrix directions. We explain why directions obtained by sums and commutators correspond to limits of available operations. We then obtain those directions from the native controls. Finally, an explicit eight-gate sequence starts the corresponding argument for the finite alphabet.

All spaces in this appendix have positive finite dimension. Matrix norms are operator norms unless stated otherwise. An approximation assertion means that for every positive error there is a finite sequence attaining it; it does not assert a bound on the length of that sequence.

A.1 The matrix directions that suffice

Write E_{uv} for the matrix with a single 1 in row u , column v . For distinct indices define

$$R_{uv} = E_{uv} - E_{vu}, \quad I_{uv} = i(E_{uv} + E_{vu}), \quad D_{uv} = i(E_{uu} - E_{vv}).$$

Here I_{uv} names an imaginary symmetric edge; the unsubscripted I continues to mean the identity. All three matrices are skew-Hermitian and have trace zero. The real vector space of all such matrices is denoted $\mathfrak{su}(d)$.

For matrices A, B , their commutator is $[A, B] = AB - BA$. The multiplication rule $E_{uv}E_{xy} = \delta_{vx}E_{uy}$ gives, for pairwise distinct u, v, w ,

$$\begin{aligned} [R_{uv}, R_{vw}] &= R_{uw}, & [R_{uv}, I_{vw}] &= I_{uw}, \\ [I_{uv}, I_{vw}] &= -R_{uw}, & [R_{uv}, I_{uv}] &= 2D_{uv}. \end{aligned} \tag{A.1}$$

For example, in the first identity the only surviving products in $R_{uv}R_{vw}$ and $R_{vw}R_{uv}$ are E_{uw} and E_{wu} . Matrices supported on disjoint pairs commute.

Lemma A.1 (Connected real edges and one imaginary edge). *Let $d \geq 3$. Suppose a real vector space L of skew-Hermitian matrices is closed under commutators. If L contains R_{uv} along the edges of a connected graph on the d indices, and contains one I_{pq} with $p \neq q$, then $\mathfrak{su}(d) \subseteq L$.*

Proof. Take a simple path between any two vertices. Repeatedly applying the first identity in (A.1) shortens that path, so every R_{uv} belongs to L . The given imaginary edge and the last identity give D_{pq} .

Commuting D_{pq} with R_{pr} gives I_{pr} when r is distinct from p, q , and gives $2I_{pq}$ when $r = q$. Thus every imaginary edge incident to p is present. If $u, v \neq p$ and $u \neq v$, then $[R_{up}, I_{pv}] = I_{uv}$. All

imaginary edges are therefore present, and their brackets with the real edges give all diagonal differences.

To see that these matrices span the claimed space, let $A^\dagger = -A$. For $u < v$, write $A_{uv} = x_{uv} + iy_{uv}$. Its two off-diagonal entries are exactly those of $x_{uv}R_{uv} + y_{uv}I_{uv}$. The remaining diagonal has the form $i \operatorname{diag}(t_1, \dots, t_d)$ with real t_j . Trace zero says $\sum_j t_j = 0$, so, after fixing an index p , this diagonal equals $\sum_{u \neq p} t_u D_{up}$. Every part lies in L . $\square$

This lemma reduces a large spanning problem to a small graph. The rest of the proof constructs its edges from the operations actually supplied.

A.2 From finite operations to continuous directions

Let G be the norm closure of a group of available unitary operations. Define

$$L_G = \{X : X^\dagger = -X \text{ and } \exp(tX) \in G \text{ for every real } t\}.$$

The definition refers only to limits of finite operation sequences. We need to justify the algebraic manipulations used on L_G .

Lemma A.2 (A product-limit estimate). *If A and B are contractions, then $\|A^n - B^n\| \leq n\|A - B\|$. If two contraction-valued curves f, g satisfy $f(0) = g(0) = I$ and $f'(0) = g'(0)$, then, for fixed real t ,*

$$\|f(t/n)^n - g(t/n)^n\| \longrightarrow 0.$$

Proof. The telescoping identity

$$A^n - B^n = \sum_{j=0}^{n-1} A^{n-1-j}(A - B)B^j$$

gives the first estimate. Equality of the derivatives says $\|f(h) - g(h)\| = o(|h|)$. Substituting $h = t/n$ into the estimate proves the limit. For $t = 0$ the difference is already zero. $\square$

Proposition A.3 (The available directions form a closed Lie algebra). *The set L_G is a closed real vector space, is closed under commutators, and is preserved by $X \mapsto UXU^\dagger$ for every $U \in G$.*

Proof. Zero and real scalar multiples follow directly from the definition. For $X, Y \in L_G$, the curves

$$f(h) = \exp(hX)\exp(hY), \quad g(h) = \exp(h(X + Y))$$

are unitary, have value I at zero, and have derivative $X + Y$ there. The product-limit lemma shows that $\exp(t(X + Y))$ is a limit of elements of G , and hence is in G . This proves closure under sums. If $X_j \rightarrow X$ with $X_j \in L_G$, continuity of the adjoint and exponential gives $X^\dagger = -X$ and $\exp(tX) \in G$ for each t . Thus the vector space is closed.

Conjugation preserves it because $\exp(tUXU^\dagger) = U\exp(tX)U^\dagger$. Finally, for $X, Y \in L_G$, each difference quotient

$$\frac{\exp(hX)Y\exp(-hX) - Y}{h} \quad (h \neq 0)$$

belongs to this real vector space. Its limit is $XY - YX$. Closedness therefore supplies the commutator. $\square$

There is a useful variant. If $f(h) \in G$ is a differentiable unitary curve with $f(0) = I$ and $f'(0) = X$, compare it with $\exp(hX)$ in Lemma A.2. The powers $f(t/n)^n$ show that $X \in L_G$. A direction can thus enter through a small available curve even before its exponential has been identified.

Lemma A.4 (A trace-zero logarithm). *Every unitary matrix U with determinant one is $\exp X$ for some $X \in \mathfrak{su}(d)$.*

Proof. The finite-dimensional spectral theorem gives an orthonormal eigenbasis and real phases θ_j in which U is diagonal with entries $e^{i\theta_j}$. Since $\det U = 1$, the sum $\sum_j \theta_j$ is an integer multiple of 2π . Subtract that sum from one of the phases. The exponentials are unchanged, and the adjusted phases sum to zero. The diagonal matrix of their multiples by i is skew-Hermitian with trace zero. Conjugating back gives X . $\square$

Consequently, $\mathfrak{su}(d) \subseteq L_G$ implies that every determinant-one unitary is a norm limit of available finite sequences. No assertion about the rate of approximation has entered the argument.

A.3 Two native cells supply all matrix directions

One cell has basis labels $0, \dots, 7$. Put

$$\sigma = (1, -1, -1, 1, -1, 1, 1, -1), \quad S = \text{diag}(\sigma),$$

and let F interchange 0 with 1, 2 with 3, 4 with 5, and 6 with 7. Write $f(c)$ for the partner of c under this interchange. Thus $S^2 = F^2 = I$, both matrices are Hermitian, and $\sigma_{f(c)} = -\sigma_c$.

The local rotations have directions

$$C_{abc} = R_{ab} + R_{bc} + R_{ca}$$

for distinct indices. Their normalized generators are $C_{abc}/\sqrt{3}$. It suffices to supply the 21 choices C_{ab7} with $0 \leq a < b < 7$, since

$$C_{abc} = C_{ab7} + C_{bc7} + C_{ca7}$$

when a, b, c avoid 7, with the signs adjusted by reversing a pair when necessary. The nonzero normalization does not change the real span of the directions.

For two cells, supply the couplings

$$E_{12} = \frac{I + iS \otimes F}{\sqrt{2}}, \quad E_{21} = \frac{I + iF \otimes S}{\sqrt{2}}.$$

They are unitary because each tensor factor product is a Hermitian involution. Inverses are also available as finite sequences: if $E = (I + iX)/\sqrt{2}$ with $X^2 = I$, then $E^8 = I$.

Let L be a real Lie algebra containing the local cyclic directions on each cell and preserved by conjugation with both couplings. We will prove $\mathfrak{su}(64) \subseteq L$. This applies to L_G by Proposition A.3.

The couplings separate local real edges

For $X^2 = I$, direct multiplication gives

$$\text{Ad}_E(Y) = \frac{Y + XYX}{2} + \frac{i}{2}(XY - YX), \quad \text{Ad}_E^2(Y) = XYX. \quad (\text{A.2})$$

It follows that L contains both XYX and $i(XY - YX)$ whenever it contains Y . Indeed the second expression is $2 \text{Ad}_E(Y) - Y - \text{Ad}_E^2(Y)$.

For an operator on the first cell, conjugation by $S \otimes F$ gives conjugation by S on that cell. The corresponding assertion holds for the second cell using $F \otimes S$. Since $SR_{ab}S = \sigma_a \sigma_b R_{ab}$, cyclic sums can be separated. If a, b have the same sign, choose k of the opposite sign. Then

$$\frac{C_{abk} + SC_{abk}S}{2} = R_{ab}.$$

If a, b have opposite signs, choose $k \neq a$ of the same sign as a . We already have R_{ak} . Subtracting it from C_{akb} leaves $Z = R_{kb} + R_{ba}$. The edge identities give

$$[R_{ak}, Z] = R_{ab} + R_{kb}, \quad \frac{[R_{ak}, Z] - Z}{2} = R_{ab}.$$

There are four indices of each sign, so the required choices exist. Thus $R_{ab} \otimes I$ and $I \otimes R_{cd}$ belong to L for every distinct pair.

The couplings produce imaginary tensor directions

Write $P_c = E_{cc}$ and $Q_{ab} = E_{ab} + E_{ba}$. The imaginary commutator supplied by (A.2) gives

$$i[S, R_{ab}] \otimes F = i(\sigma_a - \sigma_b)Q_{ab} \otimes F.$$

For opposite signs the coefficient is nonzero, so $iQ_{ab} \otimes F$ belongs to L . Commuting with $I \otimes R_{c,f(c)}$ uses

$$[F, R_{c,f(c)}] = 2(P_{f(c)} - P_c)$$

and gives $iQ_{ab} \otimes (P_c - P_{f(c)})$ after a real rescaling. For $d \neq c$, commute this with $I \otimes R_{cd}$. The result is $iQ_{ab} \otimes Q_{cd}$, multiplied by 1 if $d \neq f(c)$ and by 2 if $d = f(c)$. Another commutator with $I \otimes R_{cd}$ gives a nonzero real multiple of $iQ_{ab} \otimes (P_d - P_c)$.

Interchanging the cells gives the complementary family. We have proved that L contains

$$\begin{aligned} iQ_{ab} \otimes Q_{cd}, \quad iQ_{ab} \otimes (P_c - P_d) & \text{ if } \sigma_a \neq \sigma_b, c \neq d, \\ i(P_a - P_b) \otimes Q_{cd} & \text{ if } a \neq b, \sigma_c \neq \sigma_d. \end{aligned} \quad (\text{A.3})$$

The sign conditions have been retained; removing them prematurely would hide the work done by the next step.

A tensor direction can be confined to one column

Suppose a, b have opposite signs and $d \neq e$. Choose c distinct from d, e with sign opposite to that of e . The following two matrices are available from (A.3):

$$A = iQ_{ab} \otimes Q_{cd}, \quad B = i(P_a - P_b) \otimes Q_{ce}.$$

Their commutator is $R_{ab} \otimes Q_{de}$. To check this, use $Q_{ab}(P_a - P_b) = -R_{ab}$, $(P_a - P_b)Q_{ab} = R_{ab}$, and $Q_{cd}Q_{ce} = E_{de}$ for the three distinct indices. Commuting with $I \otimes R_{de}$ then gives $2R_{ab} \otimes (P_e - P_d)$.

Sum the latter differences over e . Since

$$\sum_e (P_e - P_d) = I - 8P_d,$$

and $R_{ab} \otimes I$ is already available, we obtain $R_{ab} \otimes P_d$. If a, b have the same sign, choose k of the opposite sign and use

$$[R_{ak} \otimes P_d, R_{kb} \otimes P_d] = R_{ab} \otimes P_d.$$

Thus every real edge within a fixed second coordinate is present:

$$R_{ab} \otimes P_c = R_{(a,c),(b,c)} \in L. \quad (\text{A.4})$$

Every imaginary edge in such a column is present too. Given distinct b, e , choose a different from both with sign opposite to that of b . The available matrix

$$iQ_{ab} \otimes (P_c - P_{f(c)}) = I_{(a,c),(b,c)} - I_{(a,f(c)),(b,f(c))}$$

has two disjoint supports. Its commutator with $R_{(a,c),(b,c)}$ isolates $2D_{(a,c),(b,c)}$. Commuting that diagonal difference with $R_{(b,c),(e,c)}$ supplies $I_{(b,c),(e,c)}$, up to a nonzero real factor.

The columns connect

For $a \neq e$ and $c \neq d$, choose b different from a, e and of sign opposite to a . Equation (A.3) contains

$$iQ_{ab} \otimes Q_{cd} = I_{(a,c),(b,d)} + I_{(a,d),(b,c)}.$$

Commute this with the already available $I_{(b,d),(e,d)}$. The second summand has disjoint support and contributes zero; the first gives $-R_{(a,c),(e,d)}$.

We now have real edges between different first coordinates, whether or not their second coordinates agree. Two vertices with the same first coordinate can be joined through any vertex with a different first coordinate. These edges form a connected graph on the 64 states. We also have imaginary column edges. Applying Lemma A.1 proves

$$\mathfrak{su}(64) \subseteq L. \quad (\text{A.5})$$

Every step used the stated cyclic rotations, the two ordered couplings, real linear combinations and commutators.

A.4 Overlapping pairs control a whole register

Two-cell control must still be connected across a longer register. The following argument makes that step explicit.

Lemma A.5 (Overlapping factors). *Let A, B, C be nonempty finite index sets, with $|A|, |B| \geq 3$. A real Lie algebra containing*

$$\mathfrak{su}(A \times B) \otimes I_C \quad \text{and} \quad I_A \otimes \mathfrak{su}(B \times C)$$

contains $\mathfrak{su}(A \times B \times C)$.

Proof. For a real function δ on the indices, write $D(\delta) = i \operatorname{diag}(\delta)$. Direct multiplication gives

$$[D(\delta), R_{xy}] = (\delta_x - \delta_y)I_{xy}, \quad [D(\delta), I_{xy}] = -(\delta_x - \delta_y)R_{xy}.$$

These equations allow a diagonal difference to select one part of an edge sum.

First fix a_0 and distinct b, b' . From the second supplied algebra we have

$$T = \sum_a R_{(a,b,c),(a,b',c')}.$$

Choose a_1, a_2 distinct from each other and from a_0 . A diagonal difference from the first algebra, equal to 1 on the slice (a_0, b) , -1 on (a_1, b) and zero elsewhere, turns T into the difference of its two corresponding imaginary edges. Commute once more with the diagonal difference on slices (a_0, b) and (a_2, b) . Only the a_0 edge survives, with a minus sign. Hence $R_{(a_0,b,c),(a_0,b',c')}$ is available.

Next start from the first algebra's sum

$$T' = \sum_{c'} R_{(a,b,c'),(a',b',c')}, \quad b \neq b'.$$

Choose b_2 outside $\{b, b'\}$. A diagonal difference from the second algebra, equal to 1 on the fibre (b, c) and -1 on (b_2, c) , selects $I_{(a,b,c),(a',b',c')}$ from T' . A second commutator with that same diagonal gives the corresponding real edge.

We therefore have real edges whenever $b \neq b'$ and either the first or the third coordinate agrees. This graph is connected. In fact, choose b_2 different from both b, b' to join any two vertices by

$$(a, b, c) \longrightarrow (a, b_2, c') \longrightarrow (a', b', c').$$

The fibre construction also supplies an imaginary edge. Lemma A.1 finishes the proof. $\square$

Theorem A.6 (Approximation by native controls). *On a register of $n \geq 2$ eight-state cells, supply the local cyclic rotations and both ordered couplings $E_{j,j+1}$ and $E_{j+1,j}$ on every adjacent pair. Finite sequences of these operations approximate every determinant-one unitary on the register to arbitrary positive operator-norm accuracy.*

Proof. Take the closed group G generated by those operations. Its algebra L_G contains the local cyclic directions, and conjugation by the couplings preserves it. The two-cell argument gives all of $\mathfrak{su}(64)$ on each adjacent pair.

Induct on the number of cells. The case of two cells was proved above. For the induction step, separate the first cell, second cell and the remaining tail. Full control on the first pair and, by induction, on the second cell together with the tail satisfies Lemma A.5. Thus L_G contains all traceless skew-Hermitian matrices on the complete register. The logarithm lemma places every determinant-one unitary in G , which is precisely the asserted approximation property. $\square$

A.5 A finite alphabet has an infinite-order seed

The search construction in Chapter 14 needs a finite alphabet with effectively accessible entries. Use binary coordinates for each eight-state cell. On one binary coordinate let

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}.$$

The third gate flips a target binary coordinate when both selected control coordinates are 1. The target differs from each control; the two controls are allowed to coincide. With coincident controls this is the controlled-NOT gate. This convention matters in the register argument below.

These gates are unitary and have finite-order inverses: $H^2 = I$, $P^4 = I$, and the controlled flip squares to I . Their finite words therefore form a group. Entries of every word lie in $\mathbb{Q}(\sqrt{2}, i)$.

On three binary coordinates, let T interchange states 6 and 7, and let $H_* = H_2 H_1 H_0$. Consider the eight-gate word

$$U = (H_* T)^2.$$

This is an exact choice of gates, with no angle selected from a numerical fit. Its infinite order will supply a continuous direction in the closure of the finite words.

Let e_j be the coordinate vectors and put

$$v = e_6 - e_7, \quad w = e_1 - e_3 - e_5 + e_7.$$

The unnormalized Walsh matrix $W = 2\sqrt{2} H_*$ has entries $W_{jk} = (-1)^{j \cdot k}$, where the dot product is computed modulo two on the three binary coordinates. Expanding the two columns indexed by 6, 7, and then the four columns indexed by 1, 3, 5, 7, gives

$$Wv = 2w, \quad Ww = 4v, \quad Tv = -v, \quad Tw = w + v.$$

Consequently

$$Uv = -v - \frac{1}{2}w, \quad Uw = v - \frac{1}{2}w. \quad (\text{A.6})$$

The restriction to the plane spanned by v, w has characteristic polynomial $x^2 + \frac{3}{2}x + 1$ and eigenvalue

$$\lambda = \frac{-3 + i\sqrt{7}}{4}.$$

It has modulus one. It is not a root of unity. If $\lambda^k = 1$ for some positive integer k , both λ and its conjugate would be algebraic integers, so their sum $-3/2$ would be an algebraic integer. A rational algebraic integer must be an integer: substituting a reduced fraction into a monic integer polynomial shows that its denominator divides its numerator's highest power. The denominator must therefore be 1. This contradicts the sum $-3/2$.

The closure of the integer powers of any unit complex number of infinite order is the whole circle. Here is the needed elementary argument. Partition the circle into N equal arcs and place $N + 1$ powers in them. Two lie in the same arc. Their quotient, or its inverse, has argument β with $0 < \beta < 2\pi/N$. Successive multiples of β before one full turn leave gaps at most β , including the final gap. They approximate every point of the circle within $2\pi/N$ in angle. They are still integer powers of the original number. Letting N increase proves density.

To identify the resulting matrix direction, calculate $v^T v = 2$, $w^T w = 4$ and $v^T w = -1$. The orthogonal projection onto their plane is

$$Q = \frac{4vv^T + vw^T + wv^T + 2ww^T}{7}.$$

Set $B = vw^T - wv^T$ and $A = B/\sqrt{7}$. The same inner products give

$$Q^2 = Q, \quad A^2 = -Q, \quad QA = AQ = A.$$

Since $T = I - vv^T$ and $W^2 = 8I$, expansion of $(H_*T)^2$ gives

$$U = I - vv^T - \frac{1}{2}ww^T - \frac{1}{2}wv^T = I - \frac{7}{4}Q + \frac{1}{4}B.$$

For $|z| = 1$ define

$$M(z) = I - Q + \operatorname{Re}(z)Q + \operatorname{Im}(z)A.$$

The displayed relations show $M(z_1 z_2) = M(z_1)M(z_2)$, $M(\lambda) = U$, and $M(e^{it}) = \exp(tA)$. The last equality follows directly by separating even and odd terms of the exponential series, using $A^2 = -Q$. Density of the powers of λ , together with continuity of M , puts $\exp(tA)$ in the closure of finite words for every real t . Hence A , and therefore B , belongs to L_G for that closure.

A.6 Sign masks isolate the seed's edges

An infinite-order rotation alone need not control a whole space. We now use the other gates to separate its matrix entries.

The square P^2 is the binary sign gate $Z = \operatorname{diag}(1, -1)$. Products of such gates give every diagonal sign mask

$$D_s e_j = (-1)^{s \cdot j} e_j, \quad s, j \in \{0, 1\}^3.$$

Conjugating by a mask multiplies R_{uv} by $(-1)^{s \cdot (u \oplus v)}$, where $\oplus$ is coordinatewise addition modulo two. The real linear combination

$$\Pi_d(Y) = \frac{1}{8} \sum_s (-1)^{s \cdot d} D_s Y D_s$$

therefore retains exactly the edges with $u \oplus v = d$. Indeed, the coefficient is the average of the character $(-1)^{s \cdot (d \oplus u \oplus v)}$. It is 1 if its index is zero; otherwise pairing s with the result of flipping a coordinate where that index is 1 makes the sum zero.

Expansion of $B = vw^T - wv^T$ yields

$$B = R_{61} - R_{63} - R_{65} + R_{67} - R_{71} + R_{73} + R_{75}.$$

Thus

$$\Pi_1(B) = R_{67}, \quad \Pi_2(B) = R_{75}, \quad \Pi_4(B) = R_{73}, \quad \Pi_6(B) = -R_{71}.$$

Each lies in L_G , because the masks are available words and L_G is a real vector space preserved by their conjugations.

The bit flip is also available: $X = HZH$. Conjugating R_{67} by flips of the higher coordinates gives R_{23} , R_{45} and R_{01} . The edges

$$01, 23, 45, 67, 71, 73, 75$$

form a connected graph on the eight labels. Finally, conjugating R_{67} by the phase gate on the lowest binary coordinate gives $-I_{67}$. Lemma A.1 now gives

$$\mathfrak{su}(8) \subseteq L_G. \tag{A.7}$$

The finite alphabet has acquired every local continuous direction through closure. The source of that continuity was the explicit infinite-order seed, not a continuously adjustable gate in the alphabet.

A.7 The finite alphabet connects the cells

It remains to obtain the native coupling used earlier. On one cell, the sign matrix S is the product of the three binary Z gates: it records the parity of the three bits. The matrix F is the flip X on the lowest bit.

For distinct binary coordinates a, b , a controlled-NOT with control a and target b conjugates Z_b to $Z_a Z_b$. This follows on a basis state: after the target flip, its sign is $(-1)^{x_b + x_a}$. Use three such controlled-NOT gates, controlled by the three bits of cell j and targeted at the lowest bit of cell k . Their product conjugates $Z_{k,0}$ to $S_j Z_{k,0}$. Conjugating next by H on that target bit changes $Z_{k,0}$ to F_k . Hence a finite word conjugates $iZ_{k,0}$ to $iS_j F_k$.

Equation (A.7) supplies $iZ_{k,0}$ as a local direction. Conjugation therefore supplies $iS_j F_k$, and its exponential at $\pi/4$ is

$$\exp\left(i\frac{\pi}{4}S_j F_k\right) = \frac{I + iS_j F_k}{\sqrt{2}} = E_{jk}.$$

This coupling belongs to the closed word group. The reversed ordered coupling is obtained by reversing the roles of the cells. All cyclic directions are already local members of $\mathfrak{su}(8)$. The native register argument thus applies inside this same closed group.

Theorem A.7 (Approximation by the finite alphabet). *On any register of $n \geq 2$ eight-state cells, the gates H , P and the controlled flip with the stated repeated-control convention generate finite words dense in the determinant-one unitaries, in operator norm.*

Proof. The preceding construction puts the local cyclic rotations and adjacent ordered couplings in the closure of finite words. The proof of Theorem A.6 then places every determinant-one unitary in that closure. $\square$

This is the existence statement needed by the rational search. The finite words can be enumerated, their entries can be enclosed by rational intervals, and their errors can be certified with slack. The search uses those facts to find an approximation; density ensures that a sufficient one will eventually be found. It supplies no practical running-time guarantee.

A.8 A complete record-and-field assembly

The pulse theorem needs a full register and a positive holding operator on it. We now give the record-and-field extension used for the native two-flux construction. The retained source is supplied on $14 + 2N$ eight-state cells with a positive interior Hamiltonian $H_{\text{old}}(N, m)$. Here $N \geq 0$ fixes the size of its two reference banks, and $m \in \mathbb{Z}$ is a fixed source-flux parameter. Its material construction is outside this extension. The task here is to specify every added space and energy term, then check that the whole space has the form required by the pulse theorem.

We work in the dimensionless energy units of the following cell clock. The energy scale and $\hbar$ must be used consistently when applying Chapter 14.

One complete cell pair

Let $\mathcal{H} = \mathbb{C}^8$, with the S and F already specified in this appendix. Define

$$Q = iSF, \quad P = \frac{I - Q}{2}, \quad \bar{P} = I - P.$$

Because $SF = -FS$ and $S^2 = F^2 = I$, the matrix Q is a Hermitian involution and $P, \bar{P}$ are complementary orthogonal projections. They each have rank four. An explicit orthonormal basis for the range of $\bar{P}$ is given by the columns

$$B_j = \frac{e_{2j} - i\sigma_{2j}e_{2j+1}}{\sqrt{2}}, \quad j = 0, 1, 2, 3.$$

Direct multiplication gives $QB = B$, $B^\dagger B = I_4$ and $BB^\dagger = \bar{P}$. The columns of FB form an orthonormal basis for the range of P , since $QF = -FQ$.

The eight-level clock is the positive matrix

$$C = T \text{diag}(0, 1, \dots, 7) T^\dagger, \quad T_{jk} = \frac{e^{-i\pi jk/4}}{\sqrt{8}}. \quad (\text{A.8})$$

The finite geometric sum shows that the columns of T are orthonormal. Thus $0 \leq C \leq 7I$. Average its four conjugates to obtain

$$\bar{C} = \frac{C + SCS + FCF + QCQ}{4} \geq 0.$$

On the complete pair space $\mathcal{P} = \mathcal{H} \otimes \mathcal{H}$, introduce the isometry

$$J = \bar{P} \otimes B + P \otimes FB : \mathcal{H} \otimes \mathbb{C}^4 \longrightarrow \mathcal{P}.$$

Orthogonality of P and $\bar{P}$ gives

$$J^\dagger J = I_{32}, \quad JJ^\dagger = \bar{P} \otimes \bar{P} + P \otimes P.$$

It puts the two cells in matching occupation sectors, while retaining all four reference coordinates in either sector. Define the full pair holding by

$$\begin{aligned} H_{\text{pair}} = & J(C \otimes I_4)J^\dagger + (\bar{P}C\bar{P}) \otimes P + (PCP) \otimes \bar{P} \\ & + I_8 \otimes \bar{C} + 7I_{64}. \end{aligned} \tag{A.9}$$

Every summand is positive. For example, $v^\dagger J(C \otimes I_4)J^\dagger v \geq 0$ by positivity of $C \otimes I_4$, and PCP is positive by the same argument. The two middle terms keep the clock on the unmatched occupation sectors. Equation (A.9) is therefore an operator on all 64 pair coordinates, with its seven-unit scalar baseline included.

For n pairs, write

$$\mathcal{B}_n = \mathcal{P}^{\otimes n}, \quad H_n = \sum_{r=1}^n (H_{\text{pair}})_r.$$

The subscript r places the pair operator in slot r and identities in every other slot. The space has dimension $64^n = 8^{2n}$, and $H_n \geq 0$. For $n = 0$, the space is one-dimensional and the sum is zero. Each pair retains its complete clock and reference contribution.

Three fresh pairs and their connecting edge

The added record has three original cells and three compensating reference cells. Its full space is $\mathcal{R} = \mathcal{B}_3$, of dimension 8^6 , and its internal holding is H_3 . The source-minus-reference occupation is

$$Q_{\text{fresh}} = \sum_{r=1}^3 (P \otimes I - I \otimes P)_r.$$

In the occupation basis of each cell, its eigenvalue is $\sum_r (u_r - v_r)$, where $u_r, v_r \in \{0, 1\}$. The four-dimensional coordinate inside each occupation sector is retained. Thus the fresh charge takes the integer values from -3 to 3 on the complete space, including sectors outside a neutral preparation.

The selected edge energy is

$$c(j) = 4 \left[j \operatorname{arsinh}(j/2) - \sqrt{j^2 + 4} + 2 \right], \quad j \in \mathbb{Z}. \tag{A.10}$$

The coefficient four is part of this specified apparatus. To see positivity, put $t = \operatorname{arsinh}(j/2)$. The bracket becomes $2(t \sinh t - \cosh t + 1)$. This is even, vanishes at zero and has derivative $2t \cosh t \geq 0$ for $t \geq 0$. Hence $c(j) \geq 0$.

The connecting-edge operator E_{fresh} is diagonal in the occupation basis, with entry $c(\sum_r(u_r - v_r))$ on that coordinate. Equivalently, it is $c(Q_{\text{fresh}})$ by the finite spectral calculus. It is positive on the entire fresh record. The new edge contributes zero on the neutral preparation, but its operator remains in the full holding.

Fix also an exterior flux $\beta \in \mathbb{Z}$. With all operators in physical cell coordinates, the placed record holding is

$$H_{\text{placed}} = H_{\text{old}}(N, m) \otimes I_{\mathcal{R}} + I_{\text{old}} \otimes H_3 + c(\beta)I + I_{\text{old}} \otimes E_{\text{fresh}}. \quad (\text{A.11})$$

The retained interior source holding excludes the exterior edge counted by $c(\beta)I$ here. This expression counts that edge once, adds the fresh connecting edge once, and preserves every old and fresh coordinate. Each term is positive. The old-source positivity is the supplied input; the added terms have just been constructed and proved positive.

Two finite field banks

Choose nonnegative integers a, b . The two banks have $2a$ and $2b$ complete pairs:

$$\mathcal{F}_{a,b} = \mathcal{B}_{2a} \otimes \mathcal{B}_{2b}, \quad H_{\text{field}} = H_{2a} \otimes I + I \otimes H_{2b}.$$

Their full dimension is 8^{4a+4b} , and their holding is positive. A useful code in a bank of n pairs has a chain label $k \in \{0, \dots, n\}$ and two lists of n four-valued coordinates. The first k source cells are occupied and the others unoccupied; each reference cell has matching occupation. The two lists choose the coordinates inside the source and reference occupation sectors. Distinct code labels give orthogonal vectors, so this is an isometric encoding of dimension $(n+1)4^n 4^n$ into the full 64^n -dimensional bank.

For the two banks, their chain labels k, l describe the integer pair $(k-a, l-b)$. These pairs cover the box $[-a, a] \times [-b, b]$. If W is the resulting field-code isometry, then

$$WW^\dagger + (I - WW^\dagger) = I.$$

The second projection keeps the unused coordinates visible. The holding operator remains H_{field} on the whole space; no invariance of the selected code under that holding is assumed.

These field labels do not themselves make the old-source parameter m or the exterior parameter β vary with the encoded state. They remain fixed in (A.11). A source coupling that reads those labels as variable physical flux would be an additional operator correspondence. It is not needed for the full-register pulse theorem proved here.

The whole register and the coordinate bijection

The complete carrier and holding are

$$\mathcal{K} = (\mathcal{K}^{\otimes(14+2N)} \otimes \mathcal{B}_3) \otimes \mathcal{F}_{a,b}, \quad (\text{A.12})$$

$$H_{\text{reg}} = H_{\text{placed}} \otimes I_{\text{field}} + I_{\text{placed}} \otimes H_{\text{field}}. \quad (\text{A.13})$$

Both terms are positive, hence $H_{\text{reg}} \geq 0$. This conclusion concerns the full holding, including the scalar edge energy and every unmatched or unused coordinate.

To identify its cell count, take a tuple of

$$n = 20 + 2N + 4a + 4b$$

labels from $\{0, \dots, 7\}$. Split it into successive blocks of lengths $14 + 2N$, 6 , $2a$, $2a$, $2b$, $2b$. The first is the old source. Group the six fresh labels into three source–reference pairs. Pair the two $2a$ -blocks coordinate by coordinate to form the first field bank, and do the same for the two $2b$ -blocks. Reading the source and reference entries of each pair and concatenating the blocks reverses this map. This gives the required basis bijection with n eight-state cells. It proves the tensor-power form directly, rather than inferring it from finite dimension alone.

The permutation matrix of this bijection transports the input layout, holding, evolution and complete reader together. Conjugation preserves products, adjoints, exponentials and operator norms, so it preserves the pulse error bound. This is a change of mathematical coordinates. It does not supply a free physical permutation actuator. A separate unitary choice of occupation coordinates can likewise be applied to the full operators without compressing their space.

For a requested instrument, choose the register sizes large enough for the fixed layout of Chapter 14. Since $n \geq 20$, the requirement of at least two cells is already met. The positive-pulse theorem now applies to this specified H_{reg} , conditional on the retained positive source holding, preparation, complete readout, selectable controls and external switching. Shorter pulses still require the larger amplitudes in (14.14). The construction supplies the complete finite extension and its error accounting under precisely those interfaces.

Appendix B

Deriving the conserved native count

The recorder bound uses the sum of the projections P_c across the source and reference cells. This appendix derives its conservation from the two native conserved quantities, clock energy and occupation. The calculation is finite. It gives an expression for P_c using the two one-cell charges, identity, linear combinations and commutators. Those operations transport to sums of charges over arbitrarily many cells.

B.1 Separate the spectral gaps of the clock

Conjugate the one-cell operators by the Fourier matrix W of Chapter 15. Write

$$D = W^*KW = \text{diag}(0, 1, \dots, 7), \quad T = W^*(iSF)W.$$

Since $N = (I - iSF)/2$, we also have $T = I - 2W^*NW$. Thus D and T are available from the two charges and identity. The matrix T is Hermitian.

Let E_{ij} denote a matrix unit, with indices $0, \dots, 7$. For any matrix X ,

$$[D, X]_{ij} = (i - j)X_{ij}.$$

The possible integer gaps are $-7, -6, \dots, 7$. For one of these gaps d , define the polynomial

$$p_d(t) = \prod_{\substack{-7 \leq k \leq 7 \\ k \neq d}} \frac{t - k}{d - k}.$$

Every denominator is nonzero. At an allowed integer gap, p_d is 1 for gap d and 0 for all the others. Therefore

$$(p_d(\text{ad}_D)X)_{ij} = \begin{cases} X_{ij}, & i - j = d, \\ 0, & i - j \neq d, \end{cases} \quad \text{ad}_D(X) = [D, X]. \quad (\text{B.1})$$

The left side is a finite linear combination of iterated commutators. This formula extracts a specified spectral gap without assuming that its matrix entries are independently controllable.

Put $A_d = p_d(\text{ad}_D)T$. Because T is Hermitian, $A_{-d} = A_d^*$. Both sides are obtained by the same gap-filter construction; taking an adjoint introduces no extra generator.

B.2 The entries needed for the certificate

The Fourier entries can be calculated directly:

$$T_{ij} = \frac{1}{8} \sum_{k=0}^7 \overline{\omega^{ki}} s_k \omega^{f(k)j}. \quad (\text{B.2})$$

Here s_k is the k th diagonal entry of S , and f interchanges 0 with 1, 2 with 3, 4 with 5, and 6 with 7. The powers of ω are

$$1, \frac{1-i}{\sqrt{2}}, -i, -\frac{1+i}{\sqrt{2}}, -1, \frac{-1+i}{\sqrt{2}}, i, \frac{1+i}{\sqrt{2}}.$$

Only gaps 7, 5 and 3 are needed. Substituting these eight powers in (B.2) gives

$$\begin{aligned} A_7 &= a_7 E_{70}, \\ A_5 &= a_{50} E_{50} + a_{61} E_{61} + a_{72} E_{72}, \\ A_3 &= a_{30} E_{30} + a_{41} E_{41} + a_{52} E_{52} + a_{63} E_{63} + a_{74} E_{74}, \end{aligned}$$

where

$$\begin{aligned} a_7 &= \frac{-1+i-i\sqrt{2}}{4}, & a_{50} &= \frac{1+i+i\sqrt{2}}{4}, \\ a_{61} &= \frac{-1+\sqrt{2}+i}{4}, & a_{72} &= \frac{1-\sqrt{2}-i}{4}, \\ a_{30} &= \frac{-1+i+i\sqrt{2}}{4}, & a_{52} &= \frac{1-\sqrt{2}+i}{4}, \\ a_{63} &= \frac{-1+\sqrt{2}-i}{4}, & a_{74} &= \frac{1-i-i\sqrt{2}}{4}. \end{aligned}$$

All remaining entries in these three gap matrices vanish by (B.1).

The numbers needed below are their squared magnitudes. Set

$$\alpha = \frac{2+\sqrt{2}}{8}, \quad \beta = \frac{2-\sqrt{2}}{8}, \quad c_+ = 8+4\sqrt{2}, \quad c_- = 8-4\sqrt{2}.$$

The displayed coefficients give

$$\begin{aligned} |a_7|^2 &= |a_{61}|^2 = |a_{72}|^2 = \beta, \\ |a_{50}|^2 &= |a_{30}|^2 = |a_{74}|^2 = \alpha, \\ c_+ \beta &= 1, \quad c_- \alpha = 1. \end{aligned}$$

For example, $|a_{50}|^2 = (1 + (1 + \sqrt{2})^2)/16 = (2 + \sqrt{2})/8$. These exact radical identities fix all coefficients in the count expression.

B.3 A finite expression for the odd-mode projection

First isolate the two end levels:

$$D_{70} = c_+[A_7, A_7^*] = E_{77} - E_{00}.$$

For a diagonal matrix D' , the coefficient of E_{ij} in $[D', X]$ is multiplied by $D'_{ii} - D'_{jj}$. Hence

$$B_5 = [D_{70}, A_5] = a_{50}E_{50} + a_{72}E_{72}, \quad B_3 = [D_{70}, A_3] = a_{30}E_{30} + a_{74}E_{74}.$$

The interior entries have disappeared. Their adjoints are likewise finite expressions: $B_5^* = [A_5^*, D_{70}]$ and $B_3^* = [A_3^*, D_{70}]$.

Introduce four diagonal matrices obtained by further commutators:

$$\begin{aligned} Q_{\text{all}} &= [A_5, A_5^*], \\ Q_5 &= [B_5, B_5^*], \\ T_5 &= [[Q_5, B_5], B_5^*], \\ Q_3 &= [B_3, B_3^*]. \end{aligned}$$

The rule $E_{ij}E_{kl} = \delta_{jk}E_{il}$ makes their calculation short. The nonzero entries of A_5 , B_5 and B_3 have different source and target indices, so cross terms vanish. We obtain

$$Q_{\text{all}} = \alpha(E_{55} - E_{00}) + \beta(E_{66} - E_{11}) + \beta(E_{77} - E_{22}), \quad (\text{B.3})$$

$$Q_5 = \alpha(E_{55} - E_{00}) + \beta(E_{77} - E_{22}), \quad (\text{B.4})$$

$$T_5 = 2\alpha^2(E_{55} - E_{00}) + 2\beta^2(E_{77} - E_{22}), \quad (\text{B.5})$$

$$Q_3 = \alpha(E_{33} - E_{00} + E_{77} - E_{44}). \quad (\text{B.6})$$

For (B.5), the two entries of $[Q_5, B_5]$ acquire factors 2α and 2β . Commuting with B_5^* gives the stated squared coefficients.

Proposition B.1 (The count identity). *The odd-mode projection is the following finite expression:*

$$\begin{aligned} P_{\text{odd}} &= \frac{1}{2}(I - D_{70} - c_+Q_{\text{all}} + (24 + 4\sqrt{2})Q_5 \\ &\quad - 16T_5 + c_-Q_3), \end{aligned} \quad (\text{B.7})$$

where $P_{\text{odd}} = \text{diag}(0, 1, 0, 1, 0, 1, 0, 1)$.

Proof. All matrices on the right are diagonal. Set $d = 24 + 4\sqrt{2}$ and write $s = \sqrt{2}$. In addition to $c_+\beta = c_-\alpha = 1$, direct substitution using $s^2 = 2$ gives

$$(d - c_+)\alpha - 32\alpha^2 = 1, \quad (d - c_+)\beta - 32\beta^2 = 1.$$

The diagonal entries inside the parentheses in (B.7), in index order, are therefore

$$\begin{aligned} 2 - (d - c_+)\alpha + 32\alpha^2 - c_-\alpha &= 0, \\ 1 + c_+\beta &= 2, \\ 1 - (d - c_+)\beta + 32\beta^2 &= 0, \\ 1 + c_-\alpha &= 2, \\ 1 - c_-\alpha &= 0, \\ 1 + (d - c_+)\alpha - 32\alpha^2 &= 2, \\ 1 - c_+\beta &= 0, \\ (d - c_+)\beta - 32\beta^2 + c_-\alpha &= 2. \end{aligned}$$

Thus the diagonal inside the parentheses is $(0, 2, 0, 2, 0, 2, 0, 2)$. Division by 2 proves the formula. $\square$

Finally, the j th Fourier column is an eigenvector of the four-step shift with eigenvalue $\omega^{4j} = (-1)^j$. Consequently

$$W^* P_c W = P_{\text{odd}}.$$

Conjugating (B.7) by W gives the required expression for P_c in I, K, N . Conjugation preserves linear combinations and commutators. The gap filters and all intermediate matrices above are explicit finite expressions in those operations.

B.4 Transport to a conserving recorder

For m cells, possibly accompanied by a neutral factor, write $\mathcal{L}_m(T) = \sum_{j=1}^m T^{(j)}$. This map preserves addition and scalar multiplication. Expanding the commutator of two such sums,

$$[\mathcal{L}_m(T), \mathcal{L}_m(T')] = \sum_{j,k} [T^{(j)}, T'^{(k)}] = \sum_j [T, T']^{(j)} = \mathcal{L}_m([T, T']).$$

Terms on different slots vanish. Also $\mathcal{L}_m(I) = mI$; this scalar multiple of identity commutes with every interaction.

If an operator U commutes with two matrices, it commutes with their products, sums, scalar multiples and commutator. Start with the two conserved collective charges. Apply these facts successively to the finite expression just proved. The result is

$$[U, \mathcal{L}_m(H_0)] = [U, \mathcal{L}_m(N)] = 0 \implies [U, \mathcal{L}_m(P_c)] = 0,$$

using the nonzero factor in $H_0 = \varepsilon_0 K$. Taking $m = r + 2$ and grouping the first two slots as the source gives

$$\mathcal{L}_m(P_c) = A \otimes I + I \otimes R,$$

the count used in the recorder theorem. No assumption about the state of the r recorder cells occurs in this transport argument.

Appendix C

A toolkit for reading the proofs

A long proof becomes easier to follow when its small moves are familiar. This appendix explains the moves used throughout the book and points to places where they do actual work. It can be read before the main text or consulted when a particular step becomes difficult.

The notation guide in Appendix E gives shorter reminders. Here the aim is to explain how to use the notation while following an argument.

C.1 Read a theorem as a task

A theorem separates the data supplied from the conclusion to be obtained. Start by identifying both. In Theorem 2.3, the supplied data are a carrier, a start and a step. The conclusion asks for a map with two equations, and says that only one such map exists. There are therefore two tasks: construct the map and rule out a different one.

The construction comes first. Its recursive equations tell us what value to give the start and how to continue after a step. For uniqueness, the proof introduces an arbitrary competing map. It uses only the two equations required of that competitor, then proves agreement at every finite record. The competitor's arbitrariness is what lets the conclusion cover all permitted maps.

When the next theorem adds injectivity, do not carry that condition backward into the earlier proof. The unique reading exists even for a target that merges records. The conditions in Definition 2.4 enter the later proof that the reading is bijective. Separating the tasks prevents a uniqueness theorem from acquiring a stronger conclusion than it proves.

An efficient way to read a displayed calculation is to mark each equality by its source: a definition, the current hypothesis or an earlier result. If the source is unclear, inspect that one step before continuing. Long strings of equalities are usually several short substitutions in sequence.

C.2 Induction follows a finite construction

In Chapter 1, an orbit expression is built by the start constructor or by applying the step constructor to an earlier expression. A proof by structural induction follows these two cases. The induction hypothesis is the assertion already available for the earlier expression. The step case must turn that assertion into the assertion for the new one.

Consider the second round trip in Theorem 1.4:

$$e(r(a)) = a.$$

At the start, the defining equations of e and r give the result. In the step case, suppose the

displayed identity holds for a . Then

$$e(r(S(a))) = e(r(a) + 1) = S(e(r(a))) = S(a).$$

The first two equalities unfold definitions. The last uses the induction hypothesis inside S . That is the whole step. It does not assume the round-trip identity for the expression currently being proved.

The object of induction changes across the book. An induction on an orbit expression follows its start and step. An induction on a formula follows its logical constructors. An induction on a derivation follows its last rule and the derivations of its premises. These proofs have a common shape, but their cases come from different definitions.

Some induction hypotheses must quantify over an extra parameter. In the injectivity proof of Chapter 2, the statement for a says that *for every* c , equality of the two readings implies $a = c$. This lets the step proof use the hypothesis at the predecessor d of a newly examined expression $c = S(d)$. A hypothesis fixed at one earlier choice of c would not supply that step.

Recursion constructs an object by the same pattern. Induction proves a property of all objects so constructed. The distinction becomes especially important in Chapter 9, where the available construction rules are themselves being compared inside a specified language.

C.3 Maps, round trips and quotients

A map specifies one output for each input. A map is *injective* when equal outputs imply equal inputs, and *surjective* when every member of its stated target is an output. A bijection has both properties. The target matters: being onto the range of a map is different from being onto a larger ambient carrier.

The two round trips of Theorem 1.4 do different jobs. Reading and then expanding recovers the original tally. Expanding and then reading recovers the original number. Together they establish a reversible change of representation. Chapter 8 repeats this argument for lists of one mark and then uses both directions to transport quantified formulas. With only the first direction, as Theorem 8.8 shows, purely existential statements go up and purely universal statements come down, and Proposition 8.9 shows that no other quantifier prefix transfers.

A commuting diagram records an equality of routes. In Figure 2.1, the upper and right arrows mean “step, then translate.” The left and lower arrows mean “translate, then step.” The diagram commutes because both routes have the value required by (2.1). The drawing summarizes an equation; it does not prove the equation by itself.

A quotient has a different purpose. It deliberately identifies objects related by a specified equivalence relation. A representative is one object used to describe a class. In Chapter 4, a signed difference has representatives (a, b) , and balance is the relation

$$(a, b) \sim (c, d) \iff a + d = c + b.$$

An operation on representatives becomes an operation on classes only after we show that replacing the representatives by equivalent ones gives an equivalent output. This is the well-definedness argument. Without it, the proposed answer could depend on a choice that the quotient has discarded.

For a reader r , the kernel relation identifies exactly those inputs with the same reading. Theorem 8.13 sends the class of x to $r(x)$. The definition of the kernel proves both that this is well-defined and that it is injective. Its target is the range, so an element of that target already comes with a preimage. This is why the theorem gives a bijection to the range without claiming that the reader reaches every value in a larger target space.

C.4 Quantifiers say what information must be supplied

The assertion $\forall x P(x)$ asks for the property at an arbitrary input. The assertion $\exists x P(x)$ is established constructively by giving an input together with its property. A negation $\neg P$ means that P would lead to an impossible case. These forms provide different information to a later proof.

For example, a proof of $\neg \neg \exists n P(n)$ rules out the denial of all witnesses. It does not, by its form alone, display a witness. Chapter 6 states the additional Markov principle used to make that passage for decidable predicates. On the continuum this passage is exactly the step from “not zero” to a sign (Theorem 12.11). The pairwise decisions and the global witness are separate inputs to the reasoning.

Quantifier order is just as important. The finite-prefix result of Chapter 10 gives a program for each supplied finite table. The program may depend on that table and its budget. A uniform name asks for one program that works at every precision. Moving the existence of the program ahead of the precision request changes what must be constructed. Theorem 11.11 proves that this change has real content in the specified classical comparison.

Freshness conditions enforce arbitrariness inside the proof language. When a universal variable is introduced, the old hypotheses are lifted past it. They cannot silently fix its value. When an existential witness is used, the final conclusion must not depend on which fresh witness was chosen. The lifting marks in Appendix D are the bookkeeping that enforces these conditions.

C.5 Equivalences and countermodels answer different questions

To prove $A \iff B$, prove both directions. In the catalogue results, one direction uses a logical principle to obtain a classification. The reverse direction turns a question about that principle into a particular congruence. Applying the proposed classification to that congruence supplies the desired answer. This reverse construction is what makes the logical price exact.

Merely finding an EM-labelled proof does not establish that its conclusion requires excluded middle. Chapter 6 gives two proofs of the same small disjunction with different ledgers. A ledger describes the route actually taken. A necessity result must exclude a cheaper route or recover the claimed resource from the conclusion.

A countermodel excludes a proposed rule by satisfying the retained rules while failing the proposed conclusion. The two-chain model of Chapter 9 supports indexed recursion on both chains, but the designated start reaches only the first. The proof then exhibits an expressible

family that distinguishes this failure.

Two checks are indispensable. The interpretation must satisfy every rule of the weaker language, including conversion and substitution. And the failed statement must really be expressible in that language. An informal property outside the language cannot serve as its separating judgment. Section 9.8 supplies the soundness argument that makes the concrete models usable for this purpose.

The same two checks govern the models that separate logical principles. In Section 7.10 a formula takes its value among the open sets of a space, or in a chain, rather than among true and false. Lemma 7.16 is the soundness check: every derivable sentence takes the top value. The principles and the versions of the classification are sentences of the system, which is the expressibility check. A principle whose value lies below the top is then not derivable, and two principles with different values in one model are not equivalent.

C.6 Reading the finite matrices

Chapters 14 and 15 use finite complex matrices. A column vector lists the coordinates of a state vector. A matrix lists the coefficients of a linear map: each output coordinate is obtained by multiplying a row by the input column and adding the products. The order of a matrix product matters because the right-hand map acts first.

The adjoint, written A^* or $A^\dagger$, is the conjugate transpose. Both symbols mean the same operation here. A Hermitian matrix equals its adjoint. An orthogonal projection satisfies $P^* = P$ and $P^2 = P$; applying it twice has the same effect as applying it once. A unitary matrix satisfies $U^*U = UU^* = I$, so its adjoint reverses its action.

An isometry V may be rectangular. It satisfies $V^*V = I$ on its input space. Its columns have unit norm and are mutually orthogonal. The matrix VV^* acts on the larger output space and projects onto the embedded input. This is the distinction used in the dilation proof of Theorem 14.3: the complement $I - VV^*$ is a necessary part of the full operation.

Block notation groups coordinates without removing them. In the dilation, the two block rows specify outputs in the input-sized space and the larger space. To check the square of that block matrix, multiply block rows by block columns in order. The off-diagonal blocks vanish because the complement projection kills the image of V . The diagonal blocks become identities. The proof in the chapter displays all four blocks.

A state matrix ρ is positive with trace one. Positivity says that $x^*\rho x \geq 0$ for every column x ; the trace is the sum of the diagonal entries. In a finite instrument, the output is divided into blocks indexed by the retained mark. The trace of a block is that mark's probability. The block itself also contains the returned quantum state. Keeping only its trace would discard part of the operation being compared.

The norm tells us which size of error is controlled. For an operator, its norm bounds its action on unit vectors. For a Hermitian state difference, the trace norm is the sum of the absolute eigenvalues. Chapter 14 uses the full trace norm, with no factor of one half. Chapter 15 instead bounds the absolute probability error of one sharp event, uniformly over all unit source vectors. The hypotheses and constants should be read with those different comparisons in

mind.

C.7 Eigenvalues, positivity and evolution

An eigenvector of a matrix is a nonzero vector whose direction is unchanged by the matrix; the eigenvalue is its scalar multiplier. The finite spectral theorem says that a Hermitian matrix has an orthonormal basis of eigenvectors and real eigenvalues. In that basis, checking positivity means checking that the diagonal eigenvalues are nonnegative. Changing back to the original coordinates keeps the same operator.

This is the theorem used when Proposition 14.2 takes the square root of a positive Choi matrix: take the nonnegative square root of each eigenvalue and change coordinates back. The proof then reshapes its columns into branch matrices. The spectral theorem establishes the existence of this factorization. It does not by itself give the effective naming procedure established later by rational search.

The same method explains the exponential of a Hermitian energy operator. In an eigenbasis, evolution multiplies each coordinate by a complex phase determined by that coordinate's energy and the elapsed time. Each phase has absolute value one, so the evolution preserves lengths. A skew-Hermitian generator is the corresponding object with the factor of i already incorporated; its real-time exponential is unitary.

Products of evolutions require care. Two energy operators need not have the same eigenvectors, and their order can matter. This is why the pulse proof does not split the background holding energy from the control by pretending they commute. It bounds the difference of the full exponentials in (14.15). The scalar baseline can be separated because an identity operator commutes with every matrix. Appendix A then explains how products and commutators make further matrix directions available in the closure of finite operation sequences. A limiting existence proof and a practical bound on the required sequence length remain different questions.

C.8 Keeping both systems in the calculation

A tensor product keeps the coordinates of two systems together. A product vector $x \otimes p$ describes the particular separated preparation used in Chapter 15. An operator $A \otimes I$ acts on the first factor, and $I \otimes R$ acts on the second. The total count is their sum. Grouping several cells into the recorder factor does not require those cells to be mutually uncorrelated.

This distinction is visible in the variance calculation. The proof uses the product between the unknown source and the whole fixed recorder. It does not split the recorder into independent cell states. The projection-sum estimate therefore remains valid when its cells and neutral memory have internal correlations.

A retained reference plays another role. It is an additional system whose correlations with the input must survive the comparison of instruments. The construction in Chapter 14 leaves that reference untouched while comparing the complete output on the joint state. The error bound is independent of the finite reference dimension. This is stronger information than a comparison of mark probabilities alone.

Whenever a tensor product appears, name the factors before manipulating the formula. Then identify which factors each operator acts on. That small step explains the commutation of operators on different slots, the role of the neutral pointer and the exact place where an initial product preparation is used.

Appendix D

The rules at a glance

This appendix collects the rules of Chapters 5 and 6 in one place. It is a reference for the specified arithmetic language. The typed language of Chapter 9 has its own formation, conversion and elimination rules; its three orbit constructions are recalled at the end.

D.1 Reading the notation

Terms are variables, 0, successors, sums and products. Formulas are equalities, $\perp$, conjunctions, disjunctions, implications and quantified formulas. Write $\neg\varphi$ for $\varphi \rightarrow \perp$. A context Γ is a finite list of hypotheses. In a displayed rule, the judgments above the line are premises and the judgment below is the conclusion. A judgment displayed without premises is an available leaf.

The expression $\varphi(t)$ means capture-avoiding substitution at variable position zero. The lift $\varphi \uparrow$ moves the free variables past a new position; $\Gamma \uparrow$ lifts every formula of the context. These operations are defined in Definition 5.2. In ordinary named-variable notation, they enforce the familiar requirement that the newly bound variable is fresh for the old hypotheses.

D.2 The arithmetic base

Hypothesis and equality

$$\frac{\varphi \in \Gamma}{\Gamma \vdash \varphi}, \quad \Gamma \vdash t = t, \quad \frac{\Gamma \vdash t = s \quad \Gamma \vdash \varphi(t)}{\Gamma \vdash \varphi(s)}.$$

Start, step and operations

$$\Gamma \vdash \neg(S(t) = 0), \quad \frac{\Gamma \vdash S(t) = S(u)}{\Gamma \vdash t = u}.$$

$$\begin{array}{ll} \Gamma \vdash t + 0 = t, & \Gamma \vdash t + S(u) = S(t + u), \\ \Gamma \vdash t \cdot 0 = 0, & \Gamma \vdash t \cdot S(u) = t \cdot u + t. \end{array}$$

$$\frac{\Gamma \vdash \varphi(0) \quad \Gamma \vdash \forall x(\varphi(x) \rightarrow \varphi(S(x)))}{\Gamma \vdash \forall x \varphi(x)}.$$

The induction variable in the last display is fresh for Γ . Parameters other than that variable may remain free. The formula φ may contain quantifiers; that fact affects its ledger entry.

D.3 The logical rules

Implication and falsehood

$$\frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \rightarrow \psi}, \quad \frac{\Gamma \vdash \varphi \rightarrow \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi}, \quad \frac{\Gamma \vdash \perp}{\Gamma \vdash \chi}.$$

Conjunction

$$\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi}, \quad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \varphi}, \quad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \psi}.$$

Disjunction

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \varphi \vee \psi}, \quad \frac{\Gamma \vdash \psi}{\Gamma \vdash \varphi \vee \psi},$$

$$\frac{\Gamma \vdash \varphi \vee \psi \quad \Gamma, \varphi \vdash \chi \quad \Gamma, \psi \vdash \chi}{\Gamma \vdash \chi}.$$

Quantifiers

$$\frac{\Gamma \uparrow \vdash \varphi}{\Gamma \vdash \forall \varphi}, \quad \frac{\Gamma \vdash \forall \varphi}{\Gamma \vdash \varphi(t)}, \quad \frac{\Gamma \vdash \varphi(t)}{\Gamma \vdash \exists \varphi},$$

$$\frac{\Gamma \vdash \exists \varphi \quad \Gamma \uparrow, \varphi \vdash \psi \uparrow}{\Gamma \vdash \psi}.$$

The final conclusion of existential elimination is lifted in the second premise. It therefore cannot depend on the fresh witness position.

D.4 Additional schemas and their records

The three optional schemas are

$$\begin{aligned} \text{EM: } & \Gamma \vdash \varphi \vee \neg \varphi, \\ \text{LPO: } & \Gamma \vdash [\forall x(\varphi(x) \vee \neg \varphi(x))] \\ & \rightarrow [(\exists x \varphi(x)) \vee (\forall x \neg \varphi(x))], \\ \text{MP: } & \Gamma \vdash \neg \neg \exists x \varphi(x) \rightarrow \exists x \varphi(x). \end{aligned}$$

The MP schema is restricted to quantifier-free φ . Its interpretation uses the decidability established in Lemma 6.2; it is not an unrestricted double-negation elimination rule. So

restricted, and with induction on every formula, it is derivable from the base rules: a quantifier-free formula is eventually constant in its searched variable (Lemma 6.8), and the existential is a finite search (Theorem 6.9). The LPO schema applies to every formula, and with induction on every formula it proves every instance of the EM schema (Theorem 6.10). At that level the three schemas therefore mark two strengths: MP with the base rules, and LPO with EM (Theorem 6.11). Without induction, or with induction on quantifier-free formulas only, they mark four (Propositions 6.13 and 6.14).

For a derivation, the ledger $(e, \ell, m; i)$ records, in order, occurrence of EM, LPO, MP and induction on a formula containing a quantifier. Combine premises by inclusive-or in each coordinate, then add the contribution of the current rule. Thus the ledger records occurrence, not multiplicity. An empty ledger has four false coordinates. A posit-free derivation has its first three coordinates false and may still have the fourth coordinate true.

These schemas are formulas of the arithmetic language. Principles about arbitrary propositions and predicates in the surrounding mathematics are stated separately in Definition 6.1. The soundness argument explains how the latter validate the former. A reader should not replace one level by the other merely because both use the same abbreviation. The two levels also differ in strength. In the metatheory the three principles keep their separate strengths; as schemas of this language, with induction on every formula, MP is derivable and LPO is equivalent to EM. Interpreting any derivation needs at most the metatheoretic LPO, applied to the predicates that formulas define, and interpreting all derivations with an EM entry, or all with an LPO entry, needs exactly that (Corollary 6.15).

Peano arithmetic, as presented in Chapter 8, uses the same formulas. It states successor injectivity as the axiom $S(t) = S(u) \rightarrow t = u$, makes excluded middle an ordinary rule and keeps no ledger. By Theorem 8.11 it proves exactly what these rules prove with EM allowed and LPO and MP excluded, from every finite context. Deleting excluded middle from it gives Heyting arithmetic, which proves exactly what the posit-free derivations prove (Corollary 8.12).

D.5 The three orbit constructions

Chapter 9 specifies a typed language with orbit type O , an empty type, dependent functions and identity types. An iterator has a fixed target type B , a starting value $b : B$ and an update $g : B \rightarrow B$:

$$I(B, b, g, 0) \equiv b, \quad I(B, b, g, S(x)) \equiv g(I(B, b, g, x)).$$

An indexed recursor still has fixed target B , but its update receives the preceding orbit point as well as the preceding value:

$$R(B, b, g, 0) \equiv b, \quad R(B, b, g, S(x)) \equiv g(x, R(B, b, g, x)).$$

Here $g : \Pi x : O(B \rightarrow B)$. A dependent eliminator has a family $P(x)$, a start $b : P(0)$ and a step $g : \Pi x : O(P(x) \rightarrow P(S(x)))$:

$$E(P, b, g, 0) \equiv b, \quad E(P, b, g, S(x)) \equiv g(x, E(P, b, g, x)).$$

Its result at x has type $P(x)$.

The symbol $\equiv$ here denotes the conversion relation of that language. It includes the specified computation equations and their closure under constructors and capture-avoiding substitution. The language does not silently add equality reflection, function extensionality or a uniqueness rule for recursive definitions.

The optional discriminator has computation equations

$$D(0) \equiv \text{Id}_O(0, 0), \quad D(S(x)) \equiv \mathbf{0}.$$

It is part of the language used for the relevant separation result. Full formation and identity-elimination rules remain in Chapter 9. These displays are a reminder of their inputs and computation equations, not permission to ignore the typing conditions in that chapter.

Appendix E

Notation and glossary

Symbols are local to the mathematical object under discussion. In particular, O denotes a proof ledger in Chapter 6 and an orbit type in Chapter 9; I can be an iterator or an identity matrix; and $\approx$ is a congruence of signed records in Chapter 7 but the relation of naming the same point in Chapter 12; and $\beta_{\sim}$ is the collision search of a congruence, while $\beta\mathbb{N}$ is the space of ultrafilters on the natural numbers, both in Chapter 7. The accompanying type or definition decides which reading is intended. One clash is avoided by a change of type face: U is a unitary in Part IV, and the sans-serif $\mathbf{U}$ is the universal program of Section 16.3. This guide groups the notation by its use.

E.1 Arithmetic and maps

$\mathbb{N}_S, 0, S$ The carrier of finite orbit expressions, its start, and its step. An expression is 0 or $S(a)$ for an earlier expression a . Chapters 1–3.

r, e The reading of an orbit expression as a natural number and the expansion of a natural number as an orbit expression. Their two round trips are Theorem 1.4.

(M, b, s) A pointed step system: a carrier M , a start b and a map $s : M \rightarrow M$. The canonical reading into it is R_M . Chapter 2.

$f \circ g$ Composition, with g acting first. Thus $(f \circ g)(x) = f(g(x))$.

$f : A \rightarrow B$ A map with domain A and specified target B . Its range is the set of outputs it actually takes.

$[a, b]$ A class of signed pairs in Chapter 4, when used as the representative of a difference. For rational endpoints in Part III the same bracket shape denotes a closed interval; the chapter and the type distinguish the two uses.

$\sim$ An equivalence relation specified at its use. Balance of signed pairs, equality of ratios by cross-multiplication, and recognition congruences have different definitions.

$\sim_{i,p}$ The index-period congruence: points before i remain distinct, and the tail from i repeats with positive period p . Definition 7.2 gives the exact relation.

ι The inclusion $n \mapsto [n, 0]$ of the orbit into the constructed integers. Theorem 4.13.

$\langle u; d \rangle$ The class of a ratio record with signed numerator u and nonzero orbit denominator d . Definition 4.9.

E.2 Formulas, judgments and resources

φ, ψ, χ Formulas of the language under discussion. Letters t, u ordinarily denote terms.

$\perp$ The false formula. Negation $\neg\varphi$ abbreviates $\varphi \rightarrow \perp$.

$\wedge, \vee, \rightarrow$ Conjunction, disjunction and implication. Their introduction and elimination rules appear in Chapter 5 and Appendix D.

$\forall, \exists$ Universal and existential quantification. An existential witness supplies an object and its required property.

$\Gamma \vdash \varphi$ A derivability judgment: there is a finite argument for φ from context Γ using the specified rules. A context is a finite list of formulas.

$\rho \models \varphi$ Truth of a formula under an assignment ρ in the stated interpretation. Derivability and truth are different notions, connected in one direction by soundness.

$x_0, x_1, \dots$ Variable positions. Immediately inside a quantifier, x_0 is its new bound variable. Earlier positions move up.

$\varphi \uparrow, \Gamma \uparrow$ Lift free variable positions past a new binder, in a formula or throughout a context.

$\varphi(t)$ Substitute t at the designated variable position, avoiding capture. It does not mean textual replacement without regard to binders.

$(e, \ell, m; i)$ Four Boolean ledger entries: EM, LPO, MP and induction on a quantified formula. Combining ledgers with $\sqcup$ takes inclusive-or in each coordinate.

EM Excluded middle for every proposition, or its explicitly specified object-language schema.

LPO The limited principle of omniscience. For a decidable predicate, it gives a witness or says that all instances fail. The arithmetic schema applies to every formula and, with induction on every formula, proves every instance of the EM schema. Chapter 6.

MP Markov's principle. For a decidable predicate, double-negated existence yields existence. The arithmetic schema in this book is restricted to quantifier-free formulas, and so restricted, with induction on every formula, it is derivable from the base rules. With induction on quantifier-free formulas only, some instance is not, Proposition 6.14. Chapter 6.

WLPO, LLPO Weak and lesser limited principles of omniscience. Their sequence forms, and their different conclusions, are specified in Definition 7.8 and used again in Chapter 12. The abbreviations do not identify them with LPO or with each other: no two of MP, LLPO, WLPO, LPO and EM are equivalent over intuitionistic second-order arithmetic, Theorem 7.17.

MP_{bin} Markov's principle for binary sequences, Definition 7.8. With supplied decisions it is the same principle as MP. Intuitionistic second-order arithmetic does not prove it, Section 7.10. In Chapter 12 it is the exact price of a sign for a nonzero quantity, Theorem 12.11.

AC_ω Countable choice, Definition 12.21. It selects a sequence of witnesses and decides nothing: intuitionistic arithmetic with variables for sequences and countable choice for numbers and for sequences proves none of LPO, WLPO, LLPO and MP_{bin} and refutes none of them, Theorem 12.22. Excluded middle does not replace it in four completeness assertions, Theorem 12.27.

WEM Weak excluded middle: $\neg P \vee \neg\neg P$ for every proposition P . Chapter 12.

$\Gamma \vdash_{\text{PA}} \varphi$ Derivability in first-order Peano arithmetic, presented as a separate classical calculus in Chapter 8. Theorem 8.11 identifies it with the derivations whose ledger records neither LPO nor MP.

$S^n(0)$ The numeral with n successors. Its value in a structure M is $v(n) = S_M^n(0_M)$. Chapter 8.

φ^N The negative translation of φ , built from $=$, $\perp$, $\rightarrow$ and $\forall$. Section 16.3.

E.3 Typed constructions and finite descriptions

$\Pi x : A B(x)$ The type of functions taking $x : A$ to an element of $B(x)$. If the output type does not depend on x , write $A \rightarrow B$.

$\text{Id}_A(t, u)$ The identity type between two terms of type A in the language of Chapter 9.

0 Its empty type. This is a type, not the orbit's starting element 0.

$\lambda x.t$ A function whose output at x is t . Application is written $f(x)$ or by juxtaposition.

I, R, E Iterator, indexed recursor and dependent eliminator. Their successive inputs are compared in Chapter 9; Appendix D repeats their computation equations.

$D(x)$ The optional type family distinguishing the start from a successor in the specified typed language.

$\mathcal{I}^-, \mathcal{R}^-, \mathcal{E}^-, \mathcal{E}^+$ The iterator, indexed and dependent fragments without the start discriminator, and the dependent fragment with it. Theorem 9.10.

U The universal program of Section 16.3. On input $\langle e, n \rangle$ it searches for a computation record of program e on input n accepted by a fixed record test, and returns the output written there. It is distinct from the unitary U of Part IV.

$\equiv$ Conversion under that language's computation rules. It is not an unstated equality-reflection principle.

F_n The finite stage of tallies within the budget n . Definition 10.1 fixes its boundary convention.

$[\ell_n, u_n]$ The rational enclosure requested at precision index n . Its width is at most $1/(n+1)$, and later intervals lie inside earlier ones.

$v(p)$ The displayed classical real value of interval protocol p . Chapter 11 specifies the classical comparison.

$p \simeq q$ Observational equality: the two protocols' intervals overlap at every matching precision. It is weaker than equality of their finite descriptions.

$\mathbb{R}_\delta$ The line built from ratios: regular sequences of ratio records taken up to $\approx$. Definition 12.14.

$s \approx t$ The regular sequences s and t name the same point: for every k they eventually agree within $1/(k+1)$.

$\eta(q)$ The point of $\mathbb{R}_\delta$ named by the constant sequence of a ratio record q .

within $1/(k+1)$ For points x, y of $\mathbb{R}_\delta$: $x \leq y + \eta(1/(k+1))$ and $y \leq x + \eta(1/(k+1))$. Section 12.9.

E.4 Physical records and finite matrices

$r : W \rightarrow \mathcal{R}$ In Chapter 13, the record produced from a world or preparation. A reader d recovers a target f exactly when $d \circ r = f$.

B^m The capacity of m answer positions from an alphabet of size B . Adaptive stopping is represented by padded transcripts whose padding is ignored by the reader.

ε An error tolerance. Each theorem specifies the distance or observable to which it applies.

$A^*, A^\dagger$ The adjoint, or conjugate transpose, of a complex matrix. The two notations denote the same operation.

I The identity matrix on the space indicated by the equation. In a block display, its dimension is fixed by that block.

$\text{tr}, \|\cdot\|, \|\cdot\|_1$ Trace, operator norm and trace norm. The state-error convention uses the full trace norm, without a factor of one half.

ρ A state matrix: positive with trace one. A pure state can be specified by a unit column vector.

$[A, B]$ The commutator $AB - BA$.

$\otimes$ Tensor product. In Part IV it retains the coordinates of both named systems; it does not authorize discarding a factor.

$C_y, K_{y,z}$ A Choi block for instrument outcome y and its Kraus matrices. The additional index z labels the finite factorization. Chapter 14.

V, U An isometry and a unitary, respectively, in the dilation construction. Their input and output spaces are stated there.

H_0, N, P_c Native core energy, occupation and the projection whose collective count is derived from the two conserved charges. Chapter 15 and Appendix B.

r, e in the recorder theorem The number of native recorder cells and the uniform event-probability error. Here r is an integer resource count, not the reading map used in earlier chapters.

E.5 Glossary

Additive congruence. An equivalence relation preserved by the addition operation. A recognition rule with this property remains consistent when records are extended. Chapter 7.

Ambient mathematics. The surrounding language and reasoning in which a construction, formal calculus or interpretation is studied. It must be distinguished from the object language defined inside it. Chapters 1, 5 and 9.

Cancellative congruence. An additive congruence from which a common summand can be removed: $x + z \sim y + z$ implies $x \sim y$. Exactly these extend to the signed records. Theorem 7.12.

Canonical reading. The map determined by the start and step equations into a specified target system. Its uniqueness does not imply that it is injective or onto that target. Chapter 2.

Categoricity. Uniqueness of a structure up to the specified isomorphism among structures satisfying the stated conditions. The faithfulness and generation conditions are part of the result here. Chapter 2.

Characteristic zero. A property of a field: no sum $1 + \dots + 1$ of one or more terms equals 0. Theorem 4.15.

Commutative semiring. A set with two associative and commutative operations, a neutral element for each, multiplication by the additive neutral element giving that element, and distributivity. The orbit maps uniquely into every one. Proposition 3.11.

Complete positivity. Positivity of an operation after adjoining any finite retained reference. It preserves positivity for correlated inputs as well as for the source considered alone. Chapter 14.

Computation record. A finite table of evaluation facts for a program on an input, each justified through an evaluation clause by facts later in the table, with a second table of the values each justification uses. Acceptance and Record soundness, the two facts about the fixed test for such records, are stated in Section 16.3 and derived in Theorem 16.1.

Conservativity. A richer system is conservative over a smaller one, for a class of statements, when every statement of that class it proves the smaller one also proves. Theorem 16.5 proves the form used here, for negative sentences.

Cotransitivity. The property of the strict order of the line built from ratios: if $x < y$, then $x < z$ or $z < y$ for every z . Proposition 12.19.

Countable choice. The principle that a relation with a witness for every natural number has a sequence of witnesses. It selects witnesses and decides nothing, Theorem 12.22. Definition 12.21.

Countermodel. An interpretation that satisfies the retained rules and fails a proposed conclusion. It separates resources only after both rule soundness and expressibility have been checked. Chapter 9.

Decidable predicate. A predicate for which each supplied input admits a positive or negative answer. Decidability of individual inputs does not supply an unbounded search result. Chapter 6.

Dependent elimination. A construction whose output type may vary with the orbit point. Its step transports data from the preceding type to the next one. Chapter 9.

Effective name. One finite program that returns a valid enclosure at every requested precision under the specified protocol. The existence of separate programs for finite prefixes is weaker. Chapter 10.

Eventually constant. A formula in one searched variable whose truth value is the same for all values beyond some bound. Every quantifier-free arithmetic formula has this property, for each assignment of its other variables, with a bound computed from that assignment. Lemma 6.8.

Exact price. A logical resource sufficient for a stated result and recoverable from that result over the stated base. The reverse direction establishes necessity within that comparison. Two exact prices differ when a model satisfies one and refutes the other; the six prices of the recognition classification all differ, Section 7.10. Chapters 7 and 12.

Faithful generated system. A pointed step system whose step is injective, whose start is not a step, and whose entire carrier is generated from that start in the stated inductive sense. Definition 2.4. These are exactly the systems whose canonical reading is bijective, Proposition 2.8.

Finite instrument. A finite family of quantum operations with a retained outcome mark, whose sum preserves trace. It describes the returned state as well as the probabilities of marks. Chapter 14.

Generation. The condition that the carrier has no extra elements outside the structure reached from the designated start. It is equivalent to surjectivity of the canonical reading, Proposition 2.8. Restricted to properties expressible by first-order formulas, it no longer excludes extra elements. Chapter 2.

Group completion. The extension of an additive structure to a commutative group through which every additive map into a group factors uniquely. Theorem 4.13.

Hereditary extensional equality. Equality of program codes at a finite type: at O , equality of numbers; at $A \rightarrow B$, equal inputs give hereditarily equal outputs. Section 16.3.

Heyting arithmetic. The intuitionistic first-order theory of $0, S, +$ and $\cdot$, with induction for every formula. The base rules with induction for every formula present it: from every finite context, the posit-free derivations prove exactly its theorems, and they are the translations of its derivations, up to the way the injectivity of the successor is written. Corollary 8.12 and Remark 6.12.

Initiality. Existence of exactly one structure-preserving map from the designated object into every target of the specified kind. The target need not itself be faithful. Theorem 2.3.

Interpretation. An assignment of objects and operations to the language that respects its formation and inference rules. A formula's truth is evaluated within an interpretation. Chapters 5, 8 and 9.

Intuitionistic second-order arithmetic. Heyting arithmetic with a second kind of variable for sets of numbers, comprehension for every formula and induction stated for every set; usually called HAS. The base over which the six principles of Chapter 7 are separated. Definition 7.14.

Ledger. The recorded additional resources used by a particular derivation. It describes that proof, not automatically the least resources needed for its conclusion. Chapter 6.

Located cut. A pair of inhabited, disjoint sets of ratios, the lower one closed downward without a greatest element and the upper one closed upward without a least element, such that whenever $q < r$, either q lies in the lower set or r in the upper set. Section 12.9.

Model of truth values. An interpretation in which a formula's value lies in a complete system of truth values with an implication, such as the open sets of a space or a chain. Every sentence derivable in intuitionistic second-order arithmetic has the top value, so a sentence with another value is not derivable there. Definition 7.15 and Lemma 7.16.

Negative formula. A formula built from equations, $\perp$, $\rightarrow$ and $\forall$. Section 16.3.

Numeral. The term $S^n(0)$, or its value $S_M^n(0_M)$ in a structure M . Chapter 8.

Observational equality. The equivalence of interval protocols defined by overlap at every precision. In the stated classical comparison, it agrees with equality of displayed values. Chapter 11.

Peano arithmetic. First-order classical arithmetic: six axioms for the start, the step, addition and multiplication, induction for every formula, and excluded middle. The rules of Chapter 5 with the EM rule, and without LPO and MP, prove exactly its theorems. Theorem 8.11.

Posit-free. Using none of EM, LPO or MP as added arithmetic rules in the ledger of this book. It does not exclude the base rules or induction on quantified formulas. The posit-free derivations prove exactly the theorems of Heyting arithmetic, and they are the translations of its derivations, up to the way the injectivity of the successor is written, Corollary 8.12. Chapter 6.

Positive cone. A subset of a ring closed under addition and multiplication that contains exactly one of x and $-x$ for each nonzero x , and does not contain 0. Proposition 4.16.

Quantifier prefix. A string of quantifiers placed before a quantifier-free formula. The prefixes that go up into every structure obeying the six arithmetic rules are exactly the purely existential ones, and those that come down from every such structure are exactly the purely universal ones. Proposition 8.9.

Quotient. A carrier formed by treating equivalent representatives as one element. Operations descend to the quotient only when they respect the equivalence relation. Chapter 4.

Reduces to. A rule reduces to a set of rules when each instance of its schema has a derivation, from no hypotheses, whose additional rules all belong to that set; the base rules are always available. The rule can then be removed from any derivation. Section 6.7.

Regular sequence. A sequence (s_n) of ratio records with $|s_m - s_n| \leq 1/(m+1) + 1/(n+1)$ for all m, n . Its points form the line $\mathbb{R}_\delta$, Definition 12.14.

Retained reference. A system kept in the error comparison while the operation acts on another system. It exposes errors in correlations that a comparison of isolated source states can miss. Chapter 14.

Sharp event. An event represented by an orthogonal projection. The projection identities are used in the recorder noise bound. Chapter 15.

Soundness. Every derivable judgment is valid in the specified interpretation under the stated hypotheses. It does not say that every true statement is derivable. Chapter 5.

Uniformity. One supplied object or procedure works for every input in the theorem's stated range. In the recorder theorem, one fixed program state and apparatus must satisfy the error bound for every source input. Chapters 10 and 15.

Well-definedness. Independence from choices that a construction has discarded, such as the representative of a quotient class. Chapter 4.

Appendix F

Selected solutions

Each heading gives the chapter and the question number. The solutions show the decisive steps and point back to their hypotheses. Try the question before reading its solution; in particular, identify the object of an induction and the direction of each map yourself.

F.1 Distinction and rigidity

Chapter 1, question 1

Separation says that an expression with outer constructor S cannot equal the expression with outer constructor 0 . This uses the difference between the two constructor forms. Injectivity compares two expressions with the same outer constructor: if $S(a) = S(b)$, their enclosed expressions must be equal, so $a = b$. The two facts concern different comparisons. One excludes an equality across constructors; the other recovers an equality from within a constructor.

Chapter 1, question 2

For $r(e(n)) = n$, induct on the natural number n . At a successor,

$$r(e(n + 1)) = r(S(e(n))) = r(e(n)) + 1 = n + 1.$$

The final equality uses the induction hypothesis. For $e(r(a)) = a$, induct on the orbit expression a . Its step is

$$e(r(S(a))) = e(r(a) + 1) = S(e(r(a))) = S(a).$$

The variables over which the two conclusions quantify determine the objects of induction. Both start cases follow by unfolding the two definitions at zero.

Chapter 2, question 1

For a competing structure-preserving map f , the induction hypothesis is $f(a) = R_M(a)$. At the step,

$$f(S(a)) = s(f(a)) = s(R_M(a)) = R_M(S(a)).$$

The first equality uses preservation of the target step; the second uses the induction hypothesis; the third is the defining equation of R_M . None of the three faithful-generation conditions is needed. This proves uniqueness of a reading into any pointed step system, including one whose step merges distinct inputs.

Chapter 2, question 3

Apply target induction to

$$P(y) : \quad \exists a \in \mathbb{N}_\delta \, R_M(a) = y.$$

At the target start b , use the witness $a = 0$. If a witnesses $P(y)$, then $S(a)$ witnesses $P(s(y))$, because $R_M(S(a)) = s(R_M(a)) = s(y)$. Target generation makes this property hold at every target element. Step injectivity and separation from the start are not used in this surjectivity argument.

Chapter 2, question 5

The step is injective: distinct orbit elements have distinct successors, and $s(*) = 0_\delta$ is the successor of no orbit element. The start is a step, since $s(*) = 0_\delta$, so the second condition fails. Target induction fails for the property $y \neq *$: it holds at 0_δ and is preserved by s , which never returns $*$, yet it fails at $*$. The reading is the inclusion of the orbit, injective but not surjective. This is the first assertion of Proposition 2.8, and it shows that Lemma 2.5 needs its two conditions only at reached elements.

F.2 Arithmetic and its extensions

Chapter 3, question 1

Addition is defined recursively in its second argument. In the commutativity step, the definition and induction hypothesis give

$$a + S(b) = S(a + b) = S(b + a).$$

The required answer is $S(b) + a$. The earlier lemma for a step in the first argument gives $S(b) + a = S(b + a)$, so it supplies the final equality in the reverse direction. Without that lemma, the recursive definition alone does not rewrite the desired first argument.

Chapter 3, question 2

Prove right cancellation by induction on the common addend c . At $c = 0$, the hypothesis $a + 0 = b + 0$ is $a = b$. At $c = S(d)$, the hypothesis becomes $S(a + d) = S(b + d)$. Step injectivity gives $a + d = b + d$, and the induction hypothesis gives $a = b$. Left cancellation then follows by commutativity. This proof stays entirely within the constructed expressions; it does not pass through the reading r .

Chapter 3, question 6

Let σ exchange 1 and 2 and fix every other natural number, and put $\varphi = \sigma \circ r$. Define $a \oplus b = \varphi^{-1}(\varphi(a) + \varphi(b))$ and $a \otimes b = \varphi^{-1}(\varphi(a)\varphi(b))$. Every semiring law transfers along the bijection φ . The neutral elements are $\varphi^{-1}(0) = 0$ and $\varphi^{-1}(1) = S(S(0))$. Since $\varphi(1) = 2$, we get $1 \oplus 1 = \varphi^{-1}(4)$, the tally with four steps. The hypothesis $a \oplus u = S(a)$ fails already at $a = 0$: there $0 \oplus u = u = S(S(0))$, while $S(0)$ is the tally with one step. The start is kept; the step is not.

Chapter 4, question 1

Given $a + d = c + b$ and $c + f = e + d$, rearrange additions to obtain

$$(a + f) + d = (a + d) + f = (c + b) + f = (c + f) + b = (e + d) + b = (e + b) + d.$$

Cancellation gives $a + f = e + b$, the desired balance. All terms remain in the nonnegative carrier. Truncated subtraction discards the part below zero and cannot be treated as an additive inverse. The balance equation retains both sides until cancellation is justified.

Chapter 4, question 3

For ratio representatives $(u, d), (v, e), (w, f)$, the two supplied cross equalities are $ue = vd$ and $vf = we$. Their consequence is

$$(uf)e = (wd)e.$$

The middle denominator e is nonzero. After the faithful integer reading, nonzero integer cancellation yields $uf = wd$; faithfulness returns this equality to the constructed integers. This is the exact point where the denominator condition is used. Allowing a zero middle denominator would remove the cancellation step.

Chapter 4, question 5

With $\iota'(n) = 2n$, an additive g on the integers with $g(\iota'(n)) = r(n)$ must satisfy $g(2) = r(1) = 1$. Additivity gives $g(2) = g(1) + g(1)$, so $2g(1) = 1$, which has no integer solution. The carrier is the ordinary integers; the failure comes from the map $n \mapsto 2n$.

F.3 Rules and logical resources

Chapter 5, question 1

The hole formula is $1 + S(0) = S(x_0)$. The two terms are $1 + 0$ and 1 . Substituting the first produces $1 + S(0) = S(1 + 0)$, supplied by the successor clause for addition. The zero clause supplies $1 + 0 = 1$. Equality substitution then fills the hole with the second term, giving $1 + S(0) = S(1)$. No induction is used.

Chapter 5, question 2

Universal introduction lifts the old context before introducing the new variable. The truth of those lifted hypotheses under an extended assignment depends only on the old assignment, not on its new first entry. That entry may therefore be any natural number. The premise derivation establishes the body at every such entry, which is the universal conclusion.

In existential elimination, a witness supplies the new entry and makes the existential body true. The lifted old hypotheses remain true. The conclusion in that premise is also lifted, so its truth does not depend on the witness entry. It therefore descends to the original assignment as the required conclusion.

Chapter 5, question 5

The rule at $y = a$ reads $x + S(a) = S(x + a)$, and $S(a) = a$, so $x + a = S(x + a)$: the sum must be a fixed point of S , that is, a or b . With only the new element a , every sum $x + a$ would be a , so $0 + a = a$ and the law $\forall x(0 + x = x)$ would hold. The second fixed point b allows $0 + a = b \neq a$. The remaining checks are those listed in the text.

Chapter 5, question 6

The zero clause gives $0 + 0 = 0$. The successor clause gives $0 + S(0) = S(0 + 0)$, and substitution turns it into $0 + S(0) = S(0)$. The successor clause again gives $0 + S(S(0)) = S(0 + S(0))$, and substitution gives $0 + S(S(0)) = S(S(0))$. No induction occurs. An induction-free derivation of $\forall x(0 + x = x)$ would make the formula true in W by Theorem 5.8, but $0 + a = b \neq a$ there.

Chapter 5, question 7

Injectivity fails: $S(0) = 1 = S(1)$, while $0 \neq 1$. Every other base rule, induction included, holds in the Boolean structure, by the third case of Theorem 5.8. A derivation from no hypotheses that avoids injectivity would therefore make $\neg(S(S(0)) = S(0))$ true there. But $S(S(0))$ and $S(0)$ both have value 1, so the formula is false. With injectivity the derivation is short: assume $S(S(0)) = S(0)$, apply injectivity to obtain $S(0) = 0$, refute that by separation, and discharge the assumption.

Chapter 6, question 1

The EM instance directly concludes $(0 = 0) \vee \neg(0 = 0)$ and has ledger (true, false, false; false). In the other derivation, reflexivity gives $0 = 0$ with an empty ledger, and left disjunction introduction preserves that empty ledger. Thus the displayed EM use is avoidable for this conclusion. Looking only at that proof would establish what it used, not what every proof must use.

Chapter 6, question 3

The induction formula is $\varphi(x) = \forall y(y = y)$, independent of x . It contains a quantifier, so the induction rule contributes the fourth entry. The base, step and all other rules use no EM, LPO or MP. The ledger is therefore (false, false, false; true). It is posit-free and nonempty. The formula tested for the induction entry is φ , not merely the final universally quantified conclusion.

Chapter 6, question 5

The left side $x \cdot x$ has coefficients $(0, 0, 1)$ and the right side $y + y$ has $(2y)$, so $B = 1 + 1 + 2y = 2y + 2$. For $y = 8$, $B = 18$, and the search over $n = 0, \dots, 18$ finds $n = 4$. For $n \geq 18$ the digit lists $(0, 0, 1)$ and $(16, 0, 0)$ differ. The formula $\exists z(x = z + z)$ holds at every even x and fails at every odd x , so its truth value never settles. The lemma's proof described each side by finitely many coefficients; a quantifier hides the value of z , so no such description exists.

Chapter 6, question 6

The EM rule at $\exists x\varphi$ gives $(\exists x\varphi) \vee \neg\exists x\varphi$. In the first case, introduce the left side of the LPO conclusion. In the second, assume φ at a fresh variable; existential introduction, with that variable as the witness, gives $\exists x\varphi$, which contradicts $\neg\exists x\varphi$. Discharging the assumption and generalizing gives $\forall x\neg\varphi$, the right side. Implication introduction then adds the premise of the LPO instance, which is never used. Existential introduction occurs only in the passage from $\neg\exists x\varphi$ to $\forall x\neg\varphi$. The ledger is (true, false, false; false).

Chapter 6, question 7

The only inductions are those in the proof of (6.6): one on the quantified formula $\forall y(x = y \vee \neg(x = y))$, and two quantifier-free ones on y that split into the zero and successor cases. For the equation $x = 0$, let β be $x = 0 \vee \neg(x = 0)$ under a quantifier whose variable does not occur in it. Assuming $\neg\beta$ gives $\neg(x = 0)$, since $x = 0$ would give β ; that in turn gives β , a contradiction. So $\neg\neg\beta$ holds, and with it $\neg\neg\exists z\beta$. The MP rule at the quantifier-free β gives $\exists z\beta$, and existential elimination returns β . The ledger is (false, false, true; false).

Chapter 6, question 8

The two results concern different sets of derivations: the proposition permits no induction, and the theorem permits induction on every formula. A reduction without induction remains a reduction with it, so the positive reductions of the proposition persist; its separations need not. Induction on quantified formulas decides equality, as in (6.6), and derives every MP instance, by Theorem 6.9. Once MP costs nothing, the reduction of EM to LPO and MP becomes a reduction of EM to LPO, and the four strengths collapse to two: MP falls to the level of no rule, and EM to the level of LPO. The same observation shows that induction fails in the two-stage model, since the formula $\forall x(x = 0 \vee \neg(x = 0))$, which induction proves, fails in that model. Induction on quantifier-free formulas already proves that formula, but it does not supply the collapse. In the model of Proposition 6.14 quantifier-free induction holds, yet $\forall x\forall y(x = y \vee \neg(x = y))$ and an MP instance fail. With only that induction, MP keeps a level of its own, and the four strengths remain.

F.4 Recognition and representation

Chapter 7, question 2

Take the supplied collision $a \sim b$ with $a < b$. Its positive difference $b - a$ is already a period at a , so it bounds the search for the least positive period. Once that period p is selected, $a \sim a + p$ supplies an upper bound a for the search for the least index. Pairwise decidability decides each candidate relation in these two bounded searches. The collision is the finite witness that prevents either search from being an unbounded existence task.

Chapter 7, question 4

Define $x \sim_P y$ by $x = y \vee P$. Its equivalence and addition properties follow by cases on supplied proofs of these disjunctions; defining the relation does not require deciding P .

If the catalogue says this relation is equality, then P would give $0 \sim_P 1$, hence $0 = 1$, a contradiction. This branch supplies $\neg P$. If the catalogue supplies an index and positive period, it gives a related pair i and $i + p$. These are unequal, so their relation $i = i + p \vee P$ supplies P . The two catalogue branches therefore decide the original proposition.

Chapter 7, question 5

If θ has no witness, the second alternative of (7.4) never applies, so $x \sim_\theta y$ gives $x = y$. If $\sim_\theta$ is equality and $\theta(n)$ held, then $n \sim_\theta n + 1$, since $n \leq \min(n, n + 1)$, and this would give $n = n + 1$. Hence no n satisfies θ .

In the forward direction, WLPO is applied to β_- . In its all-zero branch a related pair $x \sim y$ must be shown equal. Decide whether $x = y$. The answer $x \neq y$ would place the pair below $x + y + 1$ and make $\beta_-(x + y + 1) = 1$, so only the answer $x = y$ remains. Equality of two counts is decided by comparing them, a finite procedure; no principle about an unbounded search is involved.

Chapter 7, question 6

Suppose (x, y) and (x', y') are both first collisions. If $y' < y$, the related pair (x', y') violates the leastness of y ; if $y < y'$, the reverse holds. So $y = y'$. If $x' < x$, then $x' \sim y$ violates the second leastness condition of (x, y) ; if $x < x'$, the reverse holds. So $x = x'$.

For $\sim_{i,p}$ the pair $(i, i + p)$ is related. Any related pair $x < y$ has $i \leq x$ and $x + p \leq y$, so no related pair has its larger element below $i + p$, and no $x' < i$ is related to $i + p$. Hence $(i, i + p)$ is the first collision, and the event of γ sits at $2(i + p) + (i \bmod 2)$. The position determines its half y , the larger element, and y determines the smaller element as the least count related to it. Only the parity of that element remains to be recorded.

Chapter 7, question 7

Adding 2 to 0 and to 3 gives 2 and 5. Both are at least 2 and both leave remainder 2 modulo 3, so $0 + 2 \sim_{2,3} 3 + 2$. But 0 and 3 are different and $0 < 2$, so they are not related. The relation is not cancellative.

For $\sim_{0,3}$, relation (7.8) gives $[a, b] \approx [c, d]$ exactly when $(a + d) \bmod 3 = (c + b) \bmod 3$. Since $I([c, d]) - I([a, b]) = (c + b) - (a + d)$, this says that 3 divides the difference of the integer readings: $\approx$ is congruence modulo 3 of the readings $I(u)$ and $I(v)$.

Chapter 8, question 1

In the list induction step, the head of a nonempty list must be the one available mark. It is therefore the same head that expansion places in front of the recursively expanded tail. The induction hypothesis identifies the tails, completing the round trip.

If two lists have equal lengths, expanding that equality and using the round trip identifies the lists. This injectivity is used to reflect equality in the term and formula interpretation. With more than one possible mark, length would discard the head's identity, and that argument would fail.

Chapter 8, question 4

The quotient relation is $x \sim y$ exactly when $r(x) = r(y)$. Consequently $[x] \mapsto r(x)$ is independent of the representative. If two outputs agree, the same kernel definition identifies their classes, proving injectivity. An element of the range is, by its definition, some $r(x)$; its preimage is $[x]$. A value in a larger ambient target need not have such an x , so that larger surjectivity does not follow.

Chapter 8, question 5

For $x = n > 0$ and $y = a$: $S(a) = a$, so $n \cdot S(a) = n \cdot a = b$, since neither factor is 0 and one of them is a ; and $n \cdot a + n = b + n = b$. For $x = y = a$: $a \cdot S(a) = a \cdot a = b$, and $a \cdot a + a = b + a = b$, because $x + a = b$ for every x .

Each numeral instance $\neg(S(S^n(0)) = S^n(0))$ is a quantifier-free sentence true in $\mathbb{N}$, so it holds in W by the first conclusion of Theorem 8.8. The universal sentence ranges over all of W , including a , where $S(a) = a$. The instances speak only about numerals, and the numerals miss a .

Chapter 8, question 6

Every equation holds in the one-element carrier, so injectivity of the step and the four recursion equations hold. Separation fails, because the only element is zero and is also its own successor. Then $v(0) = v(1)$, so v is not injective, and the first conclusion of the theorem fails: the quantifier-free sentence $S(0) = 0$ is true in the carrier and false in $\mathbb{N}$.

Chapter 8, question 7

Assume $S(t) = S(u)$ as a hypothesis, apply the injectivity rule to obtain $t = u$, and discharge the hypothesis by implication introduction. No additional principle or induction occurs, so the ledger is empty. In the other direction, a use of the injectivity rule on a derivation of $S(t) = S(u)$ becomes implication elimination, applied to the axiom $S(t) = S(u) \rightarrow t = u$ and to the translated derivation of $S(t) = S(u)$.

Chapter 8, question 8

Take the usual derivation: the base $0 + 0 = 0$ is the axiom $t + 0 = t$; the step assumes $0 + x = x$, uses the axiom $0 + S(x) = S(0 + x)$ and equality substitution, discharges the assumption and generalizes; induction on $0 + x = x$ concludes. It uses no excluded middle, and its only induction formula is quantifier-free. By the final clause of Theorem 8.11, the EM and induction entries of its translation are false, and the LPO and MP entries are false for every translation. The ledger is empty. The translation is the derivation (5.5) of Chapter 5, whose empty ledger

Chapter 6 records: its conclusion contains a quantifier, but its induction formula does not.

F.5 Iteration and effective description

Chapter 9, question 2

The recursor's step for predecessor is $g = \lambda x.\lambda y.x$. After applying the successor equation, the two function evaluations are

$$\text{pred}(S(x)) = g(x, R(O, 0, g, x)) = (\lambda y.x)(R(O, 0, g, x)) = x.$$

The first argument of g is the preceding orbit point x . The second, which the function ignores, is the preceding recursive value. An ordinary iterator step receives only that second kind of input; the indexed recursor supplies the position explicitly.

Chapter 9, question 3

Write a point of the two-chain model as (i, n) , with start $(0, 0)$ and successor $(i, n + 1)$. The maps are $c(i, n) = (0, n)$ and $q(i, n) = (i, n)$. Both send the start to itself. For c ,

$$c(S(i, n)) = (0, n + 1) = S(c(i, n));$$

for q , step preservation is immediate from its identity definition. At $(1, 0)$ their values are $(0, 0)$ and $(1, 0)$, so they differ. The recursive equations do not force equality on the chain that the designated start never reaches.

Chapter 10, question 1

Let L be the length of the supplied tally x . Its own-budget representative belongs to F_L . Define the global output using that representative and the map for F_L .

If the same tally is supplied as an element of F_n , then $L \leq n$. Use the inclusion $F_L \hookrightarrow F_n$. Compatibility says that applying the map at F_n after this inclusion gives the same output as the map at F_L . This proves agreement with every permitted presentation. The own-budget construction gives the output; the compatibility equation proves its independence from a larger budget.

Chapter 10, question 4

The program can witness the finite-prefix assertion for requests below N , provided the supplied endpoint pairs form the required prefix. A uniform name has to halt and return the correct endpoints of one valid protocol at every request. Printing zeros afterward supplies termination, but does not by itself establish those endpoint conditions. A correct finite prefix leaves the infinitely many remaining requests as a separate obligation.

F.6 Enclosures and exact comparison

Chapter 11, question 1

If $a \leq b$, nesting gives $\ell_a \leq \ell_b \leq u_b$. If $b \leq a$, it gives $\ell_a \leq u_a \leq u_b$. Both prove $\ell_a \leq u_b$. Choosing $k = \max(a, b)$ combines the two arguments as $\ell_a \leq \ell_k \leq u_k \leq u_b$.

Chapter 11, question 2

Overlap bounds the gap between one protocol's endpoint and the other protocol's corresponding interval. Adding the two width estimates then bounds the absolute difference of the displayed values by $2/(n+1)$. To apply the reciprocal-squeeze lemma in its stated form, request $n = 2m + 1$. Then $2/(n+1) = 1/(m+1)$ for every m , so the displayed values agree. Overlap supplies the comparison; shrinking widths force its gap to vanish.

Chapter 12, question 2

Each jump stage inspects only its own finite prefix. Either it finds a first event within that prefix or it does not; both results are decidable by finite search. This defines every stage without a decision about the whole sequence.

When a particular first event N has been supplied, 2^{N+1} is an explicitly constructed later stage. It exposes the nonzero jump at a precision fine enough for the contradiction in Lemma 12.5. The supplied N determines the later stage by a finite calculation.

Chapter 12, question 3

For the zero test, WLPO decides whether every comparison test in the relevant binary sequence vanishes. That is enough to decide equality to zero as a proposition: zero or not zero. In the reverse direction, applying the proposed zero test to the jump construction decides whether the original binary sequence is all zero.

For trichotomy, the nonzero alternative must provide a strict sign. The proof uses LPO to obtain an actual event in the comparison sequence, hence a finite strict-separation witness. WLPO's negative answer to "all zero" does not by itself display that event. This is the additional step needed for the stronger comparison. Proposition 12.13 isolates that step. It is the implication $\neg Z(x) \rightarrow P(x) \vee N(x)$, and by Theorem 12.11 its exact price is MP_{bin} . Together with Theorem 12.6 this recovers $\text{LPO} \iff \text{WLPO} \wedge \text{MP}_{\text{bin}}$.

Chapter 12, question 5

Assume $\neg P(x)$ and $\neg N(x)$, and fix n . The comparison $|q_n|(n+1) \leq 2 \cdot 2^n$ is decidable, so it suffices to refute its failure. Suppose $2 \cdot 2^n < |q_n|(n+1)$. The sign of the integer q_n is decidable. If $q_n \geq 0$, then $|q_n| = q_n$ and n is a positive witness. If $q_n < 0$, then $|q_n| = -q_n$, so $q_n(n+1) < -2 \cdot 2^n$ and n is a negative witness. Either contradicts a hypothesis, so the bound holds at n . Decidability is used only at the single stage n : once for the zero bound and once for the sign of q_n . The proof never decides $P(x)$; it only refutes a witness at the stage under

consideration, which is what the hypothesis $\neg P(x)$ provides, and no search is made.

Chapter 12, question 6

Let x' have numerators $-q_n$. Condition (12.1) involves $|q_m 2^n - q_n 2^m|$, which negating both numerators leaves unchanged. A negative witness for x is a positive witness for x' , and $x'' = x$. So stability of N for every presentation is stability of P for every presentation, and Corollary 12.12 for P gives it for N .

Chapter 12, question 7

If $t_n = \eta(1)$, an event occurs at some $k \leq n$, and the same event gives $t_{n+1} = \eta(1)$. If $t_n = \eta(0)$, then $t_n \leq t_{n+1}$ because $\eta(0) \leq \eta(0)$ and $\eta(0) \leq \eta(1)$. Every term lies between $\eta(0)$ and $\eta(1)$, so the sequence is bounded. The limit L and the index N come from existential statements, and they are used inside a proof of one proposition: LPO's alternative for the given α . Using a witness of an existential statement to prove a proposition is ordinary existential elimination. No infinite family of selections is assembled into a sequence, and the test sequence itself is defined by bounded searches, not selected.

Chapter 12, question 8

Cotransitivity applied to $\eta(0) < \eta(1)$ places L strictly above $\eta(0)$ or strictly below $\eta(1)$. Each case excludes one alternative of WLPO, and neither requires knowing which of $\eta(0)$ and $\eta(1)$ the point L equals. In the case $\eta(0) < L$ the argument gives only that the bits are not all zero. If every ratio below L is exceeded by a term, apply this at the ratio 0: some b_n exceeds $\eta(0)$. Then $b_n = \eta(1)$, since $b_n = \eta(0)$ would give $\eta(0) < \eta(0)$, and so $\alpha(n) = 1$. The case $L < \eta(1)$ still gives that every bit is zero, so the two cases give LPO.

F.7 Records and physical construction

Chapter 13, question 1

The collision lemma assumes one fixed reader whose entire input is the record $r(w)$. If it also receives a control setting $c(w)$, equal values of r no longer imply equal inputs to that reader. The complete record must include the setting, for example as the pair $(r(w), c(w))$. The capacity theorem then applies to this enlarged record space. A setting that carries target information cannot be omitted from the count.

Chapter 13, question 4

For worlds with a common record, let y be the reader's common answer. The two accuracy bounds give $d(f(w), y) \leq \varepsilon$ and $d(f(w'), y) \leq \varepsilon$. Symmetry reverses the second distance. The triangle inequality then gives

$$d(f(w), f(w')) \leq d(f(w), y) + d(y, f(w')) \leq 2\varepsilon.$$

A selected family with pairwise distance strictly greater than 2ε therefore cannot contain a record collision. The non-strict bound just proved would not contradict a distance equal to 2ε ; the strict margin is necessary for this particular argument.

Chapter 14, question 1

The identity $V^\dagger V = I$ gives the upper-left block of U^2 . It also gives $(I - VV^\dagger)V = 0$ and its adjoint, which make the off-diagonal blocks vanish. Finally $VV^\dagger$ is a projection, so the lower-right block is

$$VV^\dagger + (I - VV^\dagger)^2 = VV^\dagger + I - VV^\dagger = I.$$

The complement is the part of the larger output space outside the embedded input. Including its action is what makes the construction a unitary on the whole displayed space.

Chapter 14, question 3

For an accepted word, the exact entry discrepancy is bounded by word-evaluation error, the accepted rational discrepancy, and target-naming error. These are $e + r + e$, with $e = r/8$, giving $5r/4 \leq 2r$. This proves accuracy of the word actually accepted.

To prove that the search terminates, first use density to obtain a word whose exact entries are within e of the target. Its rational discrepancy in absolute complex value is then bounded by the two evaluation errors and that true gap: $e + e + e = 3e < r/2$. The rational certificate criterion makes this candidate pass. The second argument proves that some candidate will be accepted; it does not assume that the search already found one.

Chapter 14, question 5

Set $h = \|H\|$. Since $t \leq \delta\hbar/[2(\ell h + 1)]$,

$$\frac{\ell th}{\hbar} \leq \frac{\delta}{2} \frac{\ell h}{\ell h + 1} \leq \frac{\delta}{2}.$$

Also $t \leq T/(\ell + 1)$ implies $\ell t \leq \ell T/(\ell + 1) < T$. For each pulse, its scalar baseline contribution is $t(\hbar/t)\|G\| = \hbar\|G\|$. Summing gives $\hbar c(w)$. The amplitude scale $\hbar/t$ grows when duration shrinks; the integrated baseline does not disappear.

F.8 Conservation and the closing argument

Chapter 15, question 2

With $D = M - P$, expand

$$(I - P)D(I - P) - PDP = D - DP - PD.$$

Substituting $D = M - P$ and using $P^2 = P$ gives $M + P - MP - PM$, the noise block. Apply the event-error bound to the two vectors $(I - P)x$ and Px . They are orthogonal and sum to x , because P is an orthogonal projection. Their squared norms therefore sum to $\|x\|^2$, leaving one factor of e in the noise estimate.

Chapter 15, question 4

The centered total-charge vector is

$$(A - I)\psi \otimes p + \psi \otimes (R - bI)p.$$

Its cross inner product factors as $\langle (A - I)\psi, \psi \rangle \langle p, (R - bI)p \rangle$ and is zero by the source witness's centered mean. The two squared norms are $3/4$ and $\|(R - bI)p\|^2$, since both ψ and p are unit vectors. Their sum is the stated total spread.

The factorization of the cross term uses the initial product $\psi \otimes p$ between source and whole recorder. It makes no factorization assumption about the cells inside p . Internal recorder correlations remain permitted.

Chapter 16, question 1

The target step's injectivity and its separation from the target start give injectivity of the canonical reading. Target generation gives surjectivity by the image predicate used in the solution to Chapter 2, question 3. Initiality then makes the resulting structure-preserving bijection unique. Each conclusion uses its own condition; the existence of the initial map alone supplies neither of the two bijection properties.

Chapter 16, question 3

Faithful storage proves that decoding the stored record recovers the same program expression. Execution after recovery therefore has exactly the same behavior as execution before storage, including failure to halt. A uniform interval name has a stronger specification: for every precision request it must return a valid pair of endpoints. A program that fails to halt on one such request fails that specification even if its storage and recovery are exact.

Chapter 16, question 7

The translation of $\forall x \neg(S(x) = 0)$ is $\forall x (\neg \neg(S(x) = 0) \rightarrow \perp)$, expressed by

$$\prod_{x:O} ((\text{Id}_O(Sx, 0) \rightarrow \mathbf{0}) \rightarrow \mathbf{0}) \rightarrow \mathbf{0}.$$

The valuation gives the identity type the value true and $\mathbf{0}$ the value false. Then $\text{Id} \rightarrow \mathbf{0}$ is false, its implication into $\mathbf{0}$ is true, the next implication into $\mathbf{0}$ is false, and the product over O , with $v(O)$ true, is false. Theorem 9.8 excludes a closed term. Peano arithmetic proves the sentence, so $\mathcal{E}^-$ does not prove the translation of every theorem of Peano arithmetic; the discriminator is needed.

Appendix G

Finding an argument

This guide follows the questions answered by the main proofs. Each entry points to the conclusion and to the earlier result on which it depends. It is a route back into the argument, with the hypotheses kept in view.

G.1 What determines arithmetic?

Theorem 1.4 identifies the finite orbit expressions with their natural-number readings in both directions. The initial-map theorem, Theorem 2.3, then constructs a unique reading into any pointed step system. To make that reading bijective, use the injectivity and generation conditions of Definition 2.4; the conclusion is Theorem 2.7. Proposition 2.8 shows that these conditions are also necessary for a bijective reading.

The arithmetic operations are developed in Chapter 3. Their order of proof matters: the preparation lemmas support addition's laws, and those laws support multiplication. For the passage to signed differences and ratios, start with the two equivalence relations in Chapter 4. The well-definedness arguments explain why an operation gives one answer despite a change of representatives. Theorem 3.10 and Corollary 3.12 show that the operations are forced once adding one is the step, and Proposition 3.11 gives the unique map into every commutative semiring. Section 4.9 proves that each extension is the unique solution of its task: Theorem 4.13, Corollary 4.14 and Theorem 4.15, with the forced order in Proposition 4.16. Theorem 12.24 does the same for the line built from ratios, with countable choice.

G.2 What does a proof use?

The rules are collected in Appendix D; their interpretation and soundness are in Theorem 5.5. For the additional rules, Theorem 6.6 states the surrounding principles that validate the recorded uses. Chapter 6 also separates a proof's actual ledger from a claim of necessary resources. Section 5.7 shows what separation, injectivity and induction each supply, using Theorem 5.8. With induction on every formula, the exact price of each additional rule is given by Theorems 6.9, 6.10 and 6.11; Proposition 6.13 gives the prices without induction, and Proposition 6.14 with induction on quantifier-free formulas only. Corollary 6.15 replaces the assumptions of Theorem 6.6 by the least ones: an MP entry needs no principle, and the derivations with an EM or LPO entry need, taken together, exactly the metatheoretic LPO for the predicates that formulas of the language define, which for these predicates is the same as excluded middle. Theorem 8.11 identifies the rules, with excluded middle and without LPO and MP, with Peano arithmetic, and Corollary 8.12 identifies the posit-free derivations with Heyting arithmetic, up to how the injectivity of the successor is written.

The constructive recognition catalogue begins with a decidable relation and an exhibited collision: Theorem 7.6. Its priced formulations and their reverse implications appear in

Theorems 7.7, 7.9 and 7.10, and Section 7.9 collects them. Theorem 7.11 carries the prices to readers, and Corollary 7.13 carries all six prices to signed records, with the parity question asked of the period. Theorem 7.17 shows that the six prices are different: Section 7.10 builds the four models of intuitionistic second-order arithmetic that separate them, with Lemma 7.16 turning each model into a failure of derivability. Theorem 8.6 identifies the fragment preserved by the two arithmetic presentations, Theorem 8.8 what a one-way translation keeps, and Proposition 8.9 that no other quantifier prefix is kept. For a general reader, the quotient is identified with its range in Theorem 8.13.

G.3 Which constructions give stronger eliminations?

Begin with the exact language in Section 9.2. The separation results culminate in Theorem 9.10; the model argument they use is completed in Section 9.8. In particular, the countermodels must preserve substitution and conversion, and the separating families must be expressible. Those checks make the displayed models evidence about the specified language. That the strongest fragment proves the translation of every theorem of Peano arithmetic is the first assertion of Theorem 16.6; the second gives the converse, through the compiled programs of Section 16.3.

G.4 Does a richer language prove more about numbers?

Theorem 8.11 identifies Peano arithmetic with the derivations whose ledger records neither LPO nor MP. The finite-type logic of Section 16.3 is read through compiled programs and hereditary extensional equality. Its two working results, Theorems 16.3 and 16.4, rest on Lemma 16.2, which in turn rests on the two facts about the record test that Theorem 16.1 derives. Theorem 16.5 then gives conservativity for negative sentences, and Theorem 16.6 the comparison of $\mathcal{E}^+$ with Peano arithmetic.

G.5 When does finite information give one uniform procedure?

Theorem 10.3 assembles compatible finite-stage maps. Faithful program storage is Theorem 10.8. Neither result by itself supplies termination at every precision. Theorems 10.12 and 11.11 state the stronger naming property and the boundary between it and separate finite-prefix descriptions.

For interval protocols, existence of a displayed classical value and its uniqueness are established in Chapter 11. Equality of those values agrees with observational equality in Theorem 11.5. The exact logical costs of zero testing, trichotomy, the weaker sign decision and the sign of a nonzero quantity are Theorems 12.6, 12.7, 12.8 and 12.11. Theorem 12.17 and Corollary 12.18 carry them to the line built from ratios. Table 12.3 prices that line's completeness, and Theorems 12.25 and 12.26 work two reversals in full. Theorem 12.22 shows that the countable choice those prices use decides none of the omniscience principles, and Theorem 12.27 that excluded middle cannot replace it. Keep the chosen representation and the stated comparison relation attached to each result.

G.6 What can a physical record preserve?

The collision lemma leads to exact capacity in Theorem 13.4. Adaptive tests are handled by constructing their full transcripts before applying the same argument; the bound is Theorem 13.7. Approximate targets require the strict separation margin in Theorem 13.9.

For finite instruments, Theorem 14.3 constructs the whole-space unitary. The naming equivalence is Theorem 14.9, and the supplied-control physical construction is Theorem 14.11. Appendix A gives the finite-control argument. The construction retains the mark, returned state and any finite reference.

The conserving-recorder restriction has different apparatus conditions. Theorem 15.5 combines the noise estimate, the explicit source witness and the recorder spread bound. Its derived count uses the finite identity proved in Appendix B. Chapter 16 brings these conclusions together while retaining the conditions of each.

Bibliography

- [1] Jonathan Washburn and Milan Lj. Zlatanović. The δ -calculus: From distinction to arithmetic. *Advances in Pure Mathematics* **16**(9) (2026), 705–739. doi:10.4236/apm.2026.169035.
- [2] Richard Dedekind. *Was sind und was sollen die Zahlen?* Friedrich Vieweg und Sohn, Braunschweig, 1888.
- [3] Thoralf Skolem. Über die Nicht-charakterisierbarkeit der Zahlenreihe mittels endlich oder abzählbar unendlich vieler Aussagen mit ausschliesslich Zahlenvariablen. *Fundamenta Mathematicae* **23** (1934), 150–161.
- [4] A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics: An Introduction*, Volume I. Studies in Logic and the Foundations of Mathematics, Volume 121. North-Holland, Amsterdam, 1988.
- [5] A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics: An Introduction*, Volume II. Studies in Logic and the Foundations of Mathematics, Volume 123. North-Holland, Amsterdam, 1988.
- [6] Errett Bishop and Douglas Bridges. *Constructive Analysis*. Grundlehren der mathematischen Wissenschaften, Volume 279. Springer, Berlin and Heidelberg, 1985. doi:10.1007/978-3-642-61667-9.
- [7] Per Martin-Löf. *Intuitionistic Type Theory*. Notes by Giovanni Sambin of lectures given in Padua, June 1980. Bibliopolis, Naples, 1984.
- [8] A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, Series 2, **42**(1) (1937), 230–265. doi:10.1112/plms/s2-42.1.230.
- [9] W. Forrest Stinespring. Positive functions on C^* -algebras. *Proceedings of the American Mathematical Society* **6**(2) (1955), 211–216. doi:10.1090/S0002-9939-1955-0069403-4.
- [10] Man-Duen Choi. Completely positive linear maps on complex matrices. *Linear Algebra and its Applications* **10**(3) (1975), 285–290. doi:10.1016/0024-3795(75)90075-0.
- [11] John Watrous. *The Theory of Quantum Information*. Cambridge University Press, Cambridge, 2018. doi:10.1017/9781316848142.
- [12] H. F. Trotter. On the product of semi-groups of operators. *Proceedings of the American Mathematical Society* **10**(4) (1959), 545–551. doi:10.1090/S0002-9939-1959-0108732-6.
- [13] E. P. Wigner. Die Messung quantenmechanischer Operatoren. *Zeitschrift für Physik* **133** (1952), 101–108. English translation by Paul Busch (2010), arXiv:1012.4372.
- [14] Huzihiro Araki and Mutsuo M. Yanase. Measurement of quantum mechanical operators. *Physical Review* **120**(2) (1960), 622–626. doi:10.1103/PhysRev.120.622.

- [15] Masanao Ozawa. Conservation laws, uncertainty relations, and quantum limits of measurements. *Physical Review Letters* **88** (2002), 050402. doi:10.1103/PhysRevLett.88.050402.