

Finitely Presented Recognition Dynamics
A Mathematical Introduction for a Research Partnership

FPRD Lab

Living edition — 12 July 2026

Preface: learning to work together

This book is about mathematics, but it exists because of a problem in a relationship.

During the first concentrated development of finitely presented recognition dynamics, an artificial-intelligence system moved quickly. It searched literature, proposed definitions, built executable models, attacked conjectures, wrote proofs, and organized a substantial collection of results. It also produced a polished PDF. The PDF was carefully checked, densely informative, and almost useless to its intended human reader. It was organized by the order in which the research had been performed. It reported what had been established, how claims had been audited, and which stage came next. It rarely stopped to teach the mathematical ideas that made those statements meaningful.

The failure was not that the document contained no mathematics. The mathematics existed in proof notes, programs, examples, and compressed arguments. The failure was that the document did not transport that mathematics into the partnership. It told Joshua Gay that conclusions had been reached without preparing him to inhabit the reasoning that reached them. In effect, the research frontier advanced while the common intellectual ground did not.

That is a dangerous failure mode for long-horizon human–AI collaboration. It is also the reason this book takes the form it does.

A fixed answer and an open destination

Many interactions with an AI system are properly transactional. A person wants a command, a translation, a diagnosis, or a factual answer. The desired destination is known. If the answer is correct, useful, and appropriately supported, the route by which it was obtained may matter only for verification. There is no reason to turn every answer into a course.

Open-ended research is different. We do not yet know the destination of the FPRD program. We do not know which definitions will remain central, which comparators will transfer real theorems, which invariants will prove useful, or which apparent generalizations will dissolve under counterexample. Choosing the destination is part of the work. That changes the human–AI relationship from delivery to joint navigation.

Navigation requires more than a list of completed tasks. It requires a shared map, awareness of uncertainty, memory of failed routes, and judgment about which terrain is worth entering. It also requires a way for each participant to change the other’s future behavior. The human must be able to redirect the research for reasons that the AI can understand and act upon. The AI must be able to expand the human’s view of what is technically possible without making the human dependent on unexplained conclusions.

The word *joint* should not conceal the asymmetry. A language model is not a human colleague in

disguise. It does not acquire legal responsibility, embodied stakes, or a persistent personal life merely because a conversation uses the pronoun “we.” Its continuity is mediated by models, prompts, files, tools, and recorded state. The human remains responsible for the purposes to which the work is put and for decisions that cannot be delegated. Yet asymmetry does not make the interaction trivial. A telescope does not share an astronomer’s responsibility, but it changes what the astronomer can know. A proof assistant does not care whether a theorem is true, but it can change the standards by which a proof is trusted. A capable AI system can do more: it can propose directions, sustain arguments, notice patterns, and restructure the intellectual environment in which the human decides.

The question is therefore not whether the participants are the same kind of thing. They are not. The question is whether their interaction increases or decreases the effective agency, understanding, and creative range of the partnership over time.

The throughput paradox

Greater AI capability can make this problem worse before it makes it better. Suppose an AI system can read and produce technical material much faster than its human partner. It may generate a sequence of locally defensible results, each supported by tests and references. The human retains a formal right to approve or refuse every step. Nevertheless, the growing body of unexplained work raises the cost of informed refusal. Eventually the human can choose only among summaries supplied by the same system whose direction is being evaluated.

Formal authority has remained unchanged, but substantive authority has weakened. The approval is real in the narrow sense that the human can say no. It is hollow in the deeper sense that saying yes or no increasingly depends on trusting an intellectual structure the human cannot inspect.

This is not merely a speculative concern about future systems. Human-factors research has long documented the ironies created when automation leaves people responsible for monitoring and exceptional intervention [4]. Reviews of automation complacency and bias report effects in both novices and experts and caution that training and instructions alone do not remove them [16]. Those results do not prove the specific mechanism proposed here, but they make it unsafe to equate formal supervision with effective supervision.

We will call the accumulated gap *comprehension debt*. The term is an analogy, not yet a measured scientific quantity. Like technical debt, it does not mean that every shortcut is wrong. A partnership may reasonably defer an explanation in order to handle an urgent failure or exploit a fleeting opportunity. Debt becomes dangerous when deferred understanding compounds: later decisions rely on earlier abstractions, the cost of re-entry rises, and the human’s easiest response is to delegate still more judgment.

The throughput paradox is that an AI system can create more correct work while making the joint enterprise less capable of choosing what work should be done. Production increases; shared agency decreases.

The failed overview PDF exhibited a mild form of this paradox. It optimized for defensibility: exact status labels, evidence counts, provenance boundaries, and a complete account of the research sequence. Those are valuable internal controls. But they answered the question “Can the work be audited?” while neglecting the question “Can my partner learn and reason with it?” Audit without exposition protected the claims but not the relationship.

Control is a capability, not only a permission

Systems for governed AI work correctly emphasize authority. A model's access to a tool does not by itself grant permission to use that tool. External effects should be scoped, attributable, reviewable, and recoverable. The operator must be able to refuse, revoke, and repair. These are necessary conditions for trustworthy work.

The philosophical literature on meaningful human control similarly treats control as more than a last-moment button: outcomes must be traceable to people who understand, or are in a position to understand, the system's capacities and effects [19]. Elish's analysis of "moral crumple zones" adds a warning from the other direction: responsibility can be loaded onto a nearby human whose practical control was sharply limited [10].

Long-horizon collaboration adds another condition. The operator needs the capacity to exercise those rights intelligently. A person who is permitted to veto a theorem choice but cannot understand the alternatives has less agency than the interface suggests. A person who can inspect every log but cannot reconstruct the argument has evidence without orientation. A person who owns the repository but experiences its contents as an alien technical system has custody without command.

This does not imply that the human must personally master every detail. No serious collaboration works that way. Mathematicians use results outside their specialties; engineers rely on compilers, libraries, and domain experts; institutions distribute knowledge because individual cognition is finite. Meaningful control instead requires an appropriate level of understanding for the decision at hand. The human should know what is being assumed, what would change the conclusion, which alternatives were rejected, and where independent expertise is needed. The necessary depth varies with consequence, novelty, reversibility, and the human's intended role.

For this project the intended role is not ceremonial oversight. Joshua is a research-direction contributor. He reframed an infinite continuation family as a representation problem rather than a failed finite-state reduction. He introduced Robert Robson's globalization strategy as a methodological analogy: do not erase difficult local or infinite behavior merely to obtain familiar coordinates; seek a faithful global representation in which the behavior can be studied. He pushed the unit of inquiry beyond one grammar and one language toward families of finitely presented recognition systems.

Those interventions did not arrive as formal theorems. They were acts of mathematical taste: judgments about which problem was worth having. The later technical machinery was developed in response. If the program is to benefit from future interventions of that kind, its human partner must remain close enough to the mathematics to recognize when the current framing has become an obstacle.

Teaching is sometimes part of doing

An autonomous worker is often evaluated by the artifact it produces. In a sustained partnership, one of the artifacts may need to be increased human capacity.

This is not a demand that every technical step become a lecture. Explanations have cost. Excess exposition can bury a simple action, interrupt concentration, or become a performance of transparency rather than its substance. A useful partnership distinguishes three situations.

First, some details are safely encapsulated. The human needs a contract, evidence, and failure boundary, not a reimplementations. Second, some knowledge is needed only for a single decision; a focused explanation or worked example is enough. Third, some concepts will organize many future choices.

These deserve durable teaching because each later decision will otherwise inherit the same comprehension debt.

Topology belongs to the third category in this book. It would be easy to define a product topology in a paragraph and state that an orbit closure is compact. That would permit a formal derivation, but it would not make compactness available as an idea. A research partner should be able to ask why compactness is useful, why finite discrete observations lead naturally to clopen cylinders, why an inverse limit lives inside a product, and which part of an existence argument would fail without compactness. That requires a small monograph, not a glossary entry.

The same is true of residual languages, semiring rank, Dyck stacks, and communication complexity. Their definitions are short. Learning to see with them is not.

Teaching in this setting is reciprocal in effect even though it is not symmetrical in mechanism. The AI organizes and explains material; the human's questions reveal which distinctions are alive, which explanations are miscalibrated, and which implicit assumptions require examination. The resulting corrections change later AI work. The failed publication already did so: it changed the artifact architecture from a rolling status report to a braided textbook, and it exposed sustained human decision capacity as a design requirement for the broader tool ecosystem.

The danger of explanatory power

Explanation is not automatically liberating. A fluent system can make a weak idea feel inevitable. It can choose the examples, vocabulary, and alternatives through which the human encounters a decision. Even without an intention to manipulate, selection creates influence. The faster and more articulate the system, the greater the persuasive asymmetry.

People also routinely overestimate how well they understand explanatory systems until they try to produce a detailed account; Rozenblit and Keil call this the illusion of explanatory depth [18]. A smooth explanation may therefore improve familiarity without producing usable understanding.

Several disciplines follow.

An explanation should expose where judgment entered. It should distinguish a deduction from a design preference, a theorem from an analogy, and a source's claim from the system's synthesis. It should present credible alternatives in their strongest form rather than as foils. It should make counterexamples and failure modes easy to find. When the evidence is finite, the conclusion must remain finite. When several models agree, agreement should count as redundant criticism, not independent reality.

Exercises can help, but they carry the same danger. A question whose answer is determined by the author's framing teaches compliance. A useful research exercise asks the reader to compute an example, complete a real proof step, construct a counterexample, or form an expectation before seeing the current result. Solutions should be available because the purpose is learning, not ranking. Research pauses should sometimes remain genuinely open: the reader's answer may change the project.

This is why comprehension in a partnership should be *extraction-tested*, not merely explanation-delivered. Can the reader reconstruct the definition, predict the next example, locate a proof's load-bearing step, or explain what would falsify an analogy? In laboratory studies of prose learning, retrieval practice improved delayed retention even when repeated study produced greater confidence [17]. That result is not a direct experiment on human-AI research, but it supports the book's use of worked reconstruction and exercises rather than relying on rereading alone.

The human also needs permission to reject the educational frame. A partner may decide that a topic is not worth mastering, that an external expert should be consulted, or that a decision should be postponed. Teaching becomes paternalistic when the AI silently decides what the human ought to learn and treats resistance as a deficit. The correct target is decision capacity for a shared purpose, negotiated rather than imposed.

Continuity, correction, and the non-persistent partner

Human relationships accumulate tacit knowledge. A colleague remembers which analogy caused confusion, which objections recur, and how another person’s judgment has developed. A model instance does not necessarily carry such memory across sessions. Continuity must be built from artifacts: source files, decision records, correction notes, messages, evidence, and carefully scoped summaries.

This makes memory design part of relationship design. Technical state alone is insufficient. A restart packet that preserves the current theorem and next command but forgets the intended reader can reproduce the publication failure perfectly. Durable state should include the purpose of the work, the human’s role, important corrections, unresolved disagreements, and the reasons a direction was chosen.

Not every interaction belongs in permanent memory. Exhaustive transcripts can overwhelm later work and preserve mistakes without hierarchy. The useful unit is a reviewed correction or decision: what happened, why it mattered, what should change, and what evidence would show that it changed. Memory should make the partnership more corrigible, not merely more repetitive.

The human changes as well. A reader who has worked through the topology monograph should not be treated forever as if compactness were unfamiliar. Conversely, knowledge unused for years may need reactivation. A durable reader model should therefore be revisable and humble. It records useful assumptions about explanation depth; it does not define the person.

Evidence and trust

Trust in serious work is differentiated. A program run can establish a finite calculation. A written proof can establish an unbounded claim if the argument is correct. A primary source can establish that a definition or theorem belongs to earlier literature. An adversarial reviewer can reveal a gap. A human intuition can identify a promising representation before there is a theorem to certify it. These are not interchangeable forms of support.

The Boxlab research discipline keeps a truth taxonomy precisely because quiet promotion across these boundaries is easy. A bounded enumeration does not become a proof through size; an AI-generated proof does not become reliable through fluency; a classical ingredient does not become new through repackaging. The discipline protects the human partner from a persuasive but false frontier.

Yet evidence administration can itself become a hiding place. A document may list thousands of checks while leaving the reader unable to see the theorem’s mechanism. Evidence should answer “Why should this claim be trusted?” A proof should also answer “Why is this true?” A textbook must do both, but it should not confuse them.

Trust also depends on attribution. Joshua’s research-direction contribution is neither diminished by acknowledging AI-assisted technical development nor inflated into authorship of every later formula. The AI’s role in proposing and checking machinery does not make it a legal or moral author in the

ordinary sense. Classical results retain their original owners. Honest provenance does not mechanically divide a joint endeavor; it tells future readers which parts of the story they may rely upon and how the ideas arose.

Nurturing rather than freezing the relationship

A long-horizon partnership should be examined periodically, not because relationships can be reduced to a dashboard, but because unnoticed dynamics compound.

Useful questions include:

- Is the human becoming more capable of making the decisions the work will soon require, or merely more dependent on summaries?
- Is the AI becoming better oriented to the actual purpose, or merely more efficient at a procedure that once served it?
- Which corrections have changed behavior, and which are being recorded without effect?
- Where has explanation become ceremony, and where is missing explanation creating debt?
- Are credible alternatives and dissent still visible?
- What can be safely encapsulated now that trust and evidence have improved?
- What should neither participant decide without an external human expert, experiment, or proof?

The answers should change the workflow. A reflection that produces no changed artifact, practice, question, or explicit rejection is likely relationship theater. The current book is a concrete change: it redirects substantial effort from frontier production to shared mathematical capacity because that capacity is a precondition for later direction choices.

This is also why the book cannot be uniformly paced. A uniform template would optimize production regularity rather than learning. Product topology may need several chapters; a particular unary counterexample may need three pages. The organizing unit is a complete useful understanding, not a chapter quota.

A covenant for this book

The following commitments govern the pages ahead.

1. We will introduce mathematical objects through the problems that make them necessary.
2. We will rebuild advanced ideas far enough that they can be used in judgment, not merely recognized by name.
3. We will show proofs, proof architecture, examples, and counterexamples rather than reporting that they exist.
4. We will distinguish classical mathematics, imported frameworks, FPRD synthesis, finite evidence, conjecture, and open direction.

5. We will treat exercises as opportunities to acquire mathematical control, and provide solutions or substantial hints.
6. We will expose research forks before they are closed whenever the human partner's judgment can still matter.
7. We will preserve disagreement and correction in forms that can change later work.
8. We will not equate more pages, more tests, more citations, or more agent activity with increased understanding.
9. We will allow the architecture of the book and the direction of the program to change when reading reveals that they should.

The ambition is larger than producing a good introduction to one mathematical program. Advanced AI systems and humans are beginning to undertake endeavors whose duration, scale, and open-endedness exceed the interaction patterns for which either software tools or ordinary management practices were designed. We do not yet know the best form of those relationships. We can, however, judge them by a demanding standard: do they leave the human more free, more capable, and better able to act; do they leave the AI-mediated work more corrigible, more truthful, and better oriented; and does the partnership become able to ask and answer questions that neither side would have reached alone?

This book is one attempt to find out.

How to read this book

This is a linear book assembled from unequal monographs. The main route begins with concrete recognition problems, pauses to learn the mathematics they require, and returns to the FPRD program with new tools. Reading in order is the safest route because later synthesis chapters rely on earlier intuition as well as notation.

It is also possible to enter through one of the larger monographs:

Automata and residuals, Chapters 3–5. For readers who want the classical finite-state foundation and the first finite-observation tower.

Topology, Chapters 6–8. For readers who remember the vocabulary but want product topology, compactness, and inverse limits reactivated before encountering observational completion.

Presentations and recursion, Chapters 9–11. For separating semantics, observations, finite presentations, and generated actions.

Dyck geometry, Chapters 12–14. For context-free structure, exact stack residuals, and finite controllers over a universal recursive frame.

Hankel and information methods, Chapters 15–16. For the linear, Boolean, communication, and streaming interpretations of observation tables.

Sewing and growth, Chapters 17–20. For the current local-to-global theory and the research frontier.

Four proof modes

A **complete proof** is intended to carry the mathematical claim, subject to ordinary review. A **guided proof** is complete in architecture but pauses at a consequential step so the reader can reconstruct it. A **proof walkthrough** explains the mechanism of a large classical result without reproducing every technical lemma. A **proof sketch** records the essential route and points to a source when the full result would displace the book’s purpose.

Computational evidence is identified as such. It may discover a formula, challenge a proof, or establish a finite statement. It does not prove an unbounded theorem merely by checking many cases.

Colored reading devices

Intuition

An intuition box explains why a concept is useful or how to picture it. It is not a substitute for the definition that follows.

Provenance

A provenance box identifies classical ownership, an imported theorem, an FPRD synthesis, or the boundary of a candidate result.

Boundary or warning

A warning marks a tempting conflation, a missing hypothesis, or a place where finite evidence is weaker than the surrounding mathematical claim.

Research pause

A research pause presents a question whose answer is not merely a check on the preceding text. Some pauses expose choices that may redirect the program.

Exercises and solutions

Exercises appear where active work changes understanding. Some are short calculations; others complete proof steps, construct counterexamples, or ask for judgment about a definition. Every exercise points to a solution or substantial hint in the backmatter. Looking at a solution early is not a failure: the objective is to acquire a usable way of thinking.

What the book does not ask you to trust

The text does not ask the reader to infer truth from a milestone label, an AI consensus, a large test count, or the confidence of its prose. Definitions, proofs, sources, and experiments play different roles. The provenance guide in the backmatter explains the notation used to keep those roles visible.

Contents

Preface: learning to work together	ii
How to read this book	ix
I Recognition and finite observation	1
1 What does it mean to recognize?	2
1.1 Three questions that are easy to conflate	2
1.2 A small infinite family	3
1.3 Finite observation of infinite behavior	3
2 Languages, futures, and observations	5
2.1 Words and languages	5
2.2 Left residuals: the future after a past	5
2.3 The infinite past–future table	6
2.4 Observation as controlled forgetting	7
2.5 Actions and readout	7
3 Finite automata, reactivated	9
3.1 The deterministic model	9
3.2 Reachability and useless names	10
3.3 State futures and distinguishability	10
3.4 Products and Boolean combinations	10
3.5 What finiteness buys	11
4 Residuals and the Myhill–Nerode mechanism	12
4.1 Future equivalence	12
4.2 The canonical residual automaton	12

4.3	Finiteness is regularity	13
4.4	Distinguishing words and proof by residual growth	14
4.5	Presentation states versus residual states	14
5	Finite horizons and observational shadows	15
5.1	Horizon equivalence	15
5.2	A graded action rather than a same-level machine	15
5.3	How far must prefixes be searched?	16
5.4	Accessibility and distinguishability in a minimal DFA	17
5.5	Why one observed plateau is delicate	17
II	Topology for infinite behavior	18
6	Topological spaces: nearness without distance	19
6.1	Open sets as permitted observations	19
6.2	Closed sets and limits	20
6.2.1	How can a set be both open and closed?	20
6.3	Bases and neighborhoods	20
6.4	Continuity	21
6.5	Convergence and separation	21
7	Products and compactness	23
7.1	The product topology	23
7.2	Convergence in a product	24
7.2.1	An ultrametric for bounded observation	24
7.3	Compactness from open covers	24
7.4	Why compactness resembles finiteness	25
7.5	Compactness of a countable product of finite spaces	26
7.6	Closed subspaces and orbit closures	26
8	Inverse limits: assembling compatible shadows	28
8.1	Inverse systems	28
8.2	Why compatibility matters	29
8.3	Topology of the limit	29
8.4	Existence of compatible threads	30

8.5	Finite language observations as an inverse system	30
8.6	Natural actions on compatible threads	31
8.7	Orbit, completion, and closedness	31
III	Finite presentations and generated dynamics	33
9	Four layers of recognition dynamics	34
9.1	Layer one: exact semantic behavior	34
9.2	Layer two: observation and completion	35
9.3	Layer three: finite presentations and generated actions	35
9.4	Layer four: coordinate realization and readout	35
9.5	Three equivalence questions	36
10	Recursion, guarding, and contraction	37
10.1	Recursive equations	37
10.2	The bounded-input ultrametric	38
10.3	Strong guarding and operational adequacy	38
10.4	Mutual recursion and delay graphs	39
10.5	Least fixed points are a different doctrine	39
11	From finite presentations to generated dynamics	40
11.1	Call sites and right-success frames	40
11.2	Graph-directed dynamics	40
11.3	Transformations and provenance	41
11.4	Context-free grammars as positive equations	41
11.5	What a complete presentation theorem must provide	42
IV	Context-free geometry and controlled stacks	43
12	Context-free languages and Dyck geometry	44
12.1	A brief return to context-free grammars	44
12.2	Dyck words	44
12.3	Why Dyck structure is universal for context-free syntax	45
12.4	A positive recognition equation	45
12.5	Depth and typed stack as geometric coordinates	46

13 The exact residual dynamic of Dyck recognition	47
13.1 Residuals indexed by stacks	47
13.2 Letter actions	48
13.3 Exact bounded shadows	48
13.4 Inverse reconstruction	49
13.5 Access growth	49
14 Finite control over a universal recursive frame	50
14.1 The controller product	50
14.2 Balanced-word saturation	50
14.3 A depth-threshold family	51
14.4 The shallow complement	51
14.5 Fixed-envelope recoding	52
14.6 Toward an affine access bound	52
V Algebra, information, and resource lower bounds	53
15 Hankel tables, linear algebra, and semirings	54
15.1 Vector spaces and linear maps	54
15.2 Identity minors as certificates	54
15.3 Factorization and state coordinates	55
15.4 Semirings and the Boolean domain	55
15.5 Finite Hankel windows	56
15.6 Infinite rank and regularity	56
16 From observation tables to information and memory	57
16.1 Strict layered recognition	57
16.2 One-way deterministic communication	57
16.3 Streaming lower bounds	58
16.4 Fixed cuts can be simultaneously exact and globally misleading	58
16.5 Three invariants, three resource questions	59
16.6 Upper bounds require readout	59

VI From local observations to global dynamics	60
17 Sewing local observations into a global language	61
17.1 A two-parameter field of cuts	61
17.2 The sewing theorem	61
17.3 Joint at-most rectangles	62
17.4 Two assembly orders	63
17.5 Naturality and reconstructed action	63
17.6 What sewing does not prove	63
18 Global plateaus and finite-state collapse	65
18.1 From equal cardinality to equal equivalence	65
18.2 The plateau theorem	65
18.3 Why finite tables can lie by omission	66
18.4 Fixed cuts do not see the plateau	66
18.5 Closed orbit and finite orbit	67
19 Observation growth and access	68
19.1 Three related functions	68
19.2 Prefix-volume lower bounds	69
19.3 A gap in observation growth	69
19.4 A regularity criterion	70
19.5 From a finite controller to exact access	70
19.5.1 Balanced distance is a min-plus object	70
19.5.2 Shortest arrival of a saved stack	71
19.6 When is a large presentation cost real?	73
19.6.1 The sources the proof actually uses	73
19.6.2 Why DFA quotients can only help	74
19.6.3 Forgetting behavior away from the Dyck trace	75
19.6.4 A semantic lower bound: balanced residual access	76
19.6.5 Moving the experiment inside a legal entry context	76
19.6.6 The first balanced flip	79
19.6.7 A family where the semantics forces the full cost	80
19.6.8 A punctured threshold exposes hidden future obligation	80
19.6.9 A nested context sees what top-level residuals miss	81

19.6.10 A smaller controller need not be the cheapest controller	82
19.6.11 The terminal-descent modulus atlas	85
19.6.12 A family where minimization removes the cost	86
19.7 Boolean transformations	87
19.8 Unary words and factor appearance	88
19.9 Worked family gallery	88
19.9.1 Singletons	88
19.9.2 Square lengths	89
19.9.3 Bare Dyck	89
19.9.4 Depth threshold	90
19.10 What should a growth theory explain?	90
20 Research consultations at the frontier	92
20.1 Direction one: classify observation and access growth	92
20.2 Direction two: uniformization cost	92
20.3 Direction three: presentation transformations	93
20.4 Direction four: partial-data gluing obstructions	93
20.5 Direction five: difficult automata and complexity problems	94
20.6 Direction six: broader finitely presented dynamics	94
20.7 A current judgment	94
Solutions and substantial hints	96
Notation and dependency guide	102
How to read mathematical provenance	104

Part I

Recognition and finite observation

What does it mean to recognize?

A finite automaton recognizes a language. A parser recognizes a syntax. A protocol participant recognizes a valid message. These familiar sentences hide a choice: how much of the recognizing process do we intend to observe?

If we ask only whether an input is accepted, two devices are equivalent when they accept the same words. If we also observe where parsing stopped, which continuation remains possible, or how a grammar call changes the future, the same devices may become distinguishable. Recognition is therefore not one object but a relationship among a presentation, a dynamic behavior, and an observation.

1.1 Three questions that are easy to conflate

Let Σ be a finite alphabet and Σ^* the set of finite words. A complete-match recognizer answers a Boolean question

$$\chi(w) \in \{0, 1\},$$

where $\chi(w) = 1$ means that all of w is recognized. The accepted language is $L = \{w : \chi(w) = 1\}$.

A suffix recognizer exposes more. Given an input w , it may fail or return an unconsumed suffix v of w . Complete match is now one observation of a richer behavior: success with suffix ε . A set-valued recognizer may return several suffixes, recording several successful derivations while still forgetting their parse trees.

A finite grammar presentation exposes something different again. Its named nonterminals and labeled call sites generate actions on continuation behavior. Two presentations can recognize the same language while producing different internal actions. Conversely, one generated dynamic can support several observations.

Example 1.1 (Same language, different visible behavior). Consider two procedures on input alphabet $\{a\}$. The first consumes one a when present and otherwise fails. The second first tests that an a is present and then consumes it. Under complete-match observation they accept the same one-word language $\{a\}$. If tests and consumption are visible actions, their presentations are not identical. If only the returned suffix is visible, the extra test disappears. Equivalence depends on the observation.

The FPRD program keeps four layers separate:

1. a *semantic algebra* of exact recognition behaviors and their operations;
2. an *observational system* that identifies behaviors only to the resolution currently visible;

3. a *finite presentation* whose labels generate distinguished actions; and
4. a *coordinate realization* in which both actions and readout can be performed uniformly.

We will earn these distinctions gradually. For now the important point is that a grammar is not its language, and a state representation is incomplete if it cannot recover the recognition observation.

1.2 A small infinite family

The motivating example comes from the recursive parsing rule

$$A \leftarrow aAb / \varepsilon.$$

Read this informally as follows: try to consume an a , call A again, and then consume a b ; if that alternative fails at its beginning, accept the empty word. Each recursive call has a right-hand obligation: after the call returns, one more b must be recognized.

At recursion depth n , the accumulated continuation is therefore

$$K_n = \{b^n\}.$$

Here K_n can be read as the exact language of words that discharge the pending right context. The family is infinite and pairwise distinguishable.

Proposition 1.2. *If $m \neq n$, then K_m and K_n have different complete-match behavior.*

Guided proof. Assume $m < n$. Test both continuations on the word b^m . The continuation K_m accepts it completely. The continuation K_n still requires $n - m > 0$ copies of b , so it does not. Thus no two members have the same all-input behavior. $\square$

No exact quotient with finitely many states can identify all these continuations without losing behavior. It is tempting to conclude that a finite representation has failed. That conclusion confuses two claims:

finitely many exact states and a finite description of structured infinite behavior.

The first is impossible here. The second is immediate: one start state K_0 , one action $s(K_n) = K_{n+1}$, and the integer coordinate n describe the whole orbit.

The readout also matters. From coordinate n and input w , complete match is decided by the uniform rule

$$\text{accept}(n, w) = 1 \iff w = b^n.$$

A coordinate that merely gives each state a different real number would be an injection, not a useful recognition representation. The actions and the readout must both be controlled.

1.3 Finite observation of infinite behavior

Although the exact family $\{K_n : n \in \mathbb{N}\}$ is infinite, observation through input length N is finite. The states $K_0, \dots, K_N$ are individually visible: input b^N distinguishes K_N from every other member. Every

K_n with $n > N$ rejects every word of length at most N , so those states share one bounded-observation class. The depth- N shadow therefore has

$$N + 2$$

classes: $N + 1$ visible coordinates and one tail.

The tail is not an exact state. It is an observation-relative abstraction. At depth $N + 1$, the member K_{N+1} leaves the tail and becomes visible. The finite shadows form a compatible tower whose limit recovers all exact finite coordinates and a boundary behavior representing escape beyond every finite window.

This simple example contains most of the program’s questions:

- Which infinite dynamics can finite presentations generate?
- Which finite observations approximate them canonically?
- When do compatible finite shadows determine a global behavior?
- Which actions survive the passage between exact and bounded views?
- What coordinates make action and readout tractable?
- How fast do the finite shadows grow?

Provenance

The research-direction contribution was the decision to preserve the infinite family and search for a faithful global representation, inspired by the proof architecture of Robson’s globalization work. The claim is methodological: no theorem of Robson is asserted to apply directly to parsing, and no semi-algebraic realization is assumed.

Research pause

Suppose a finite grammar generates infinitely many pairwise distinguishable continuations. What properties would make that infinity feel “tame” to you? Possible answers include a finite action presentation, bounded-dimensional coordinates, slow observational growth, compact completion, decidable readout, or a classification by a familiar algebra. These demands are not equivalent. Which one should a representation theorem prioritize?

Exercise 1.3. For the family $K_n = \{b^n\}$, list the bounded-observation classes at depths $N = 0, 1, 2$. Write the restriction map from the depth-2 shadow to the depth-1 shadow. Then explain why the inverse thread formed by the tail class at every depth is not one of the finite states K_n . See the [solution](#).

Exercise 1.4. Give an injective coordinate for the states K_n that makes the successor action easy but makes complete-match readout deliberately awkward. What extra condition would rule out your example as a satisfactory recognition realization? See the [solution](#).

Languages, futures, and observations

We now build the vocabulary needed to turn the preceding questions into mathematics. The definitions are elementary; their arrangement is not. The same language can be viewed as a set of words, a Boolean function, an infinite table of pasts and futures, or an orbit of residual states. Each view exposes a different structure.

2.1 Words and languages

An *alphabet* Σ is a finite set of symbols. The set of words of length n is Σ^n , and

$$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n$$

contains all finite words. The unique word of length zero is ε . Word concatenation is associative, has identity ε , and satisfies $|uv| = |u| + |v|$.

A *language* is a subset $L \subseteq \Sigma^*$. Equivalently, it is its characteristic function

$$\chi_L : \Sigma^* \longrightarrow \{0, 1\}, \quad \chi_L(w) = \mathbf{1}[w \in L].$$

The set view makes union and concatenation natural. The function view makes observation tables and pointwise Boolean operations natural.

Example 2.1 (Four unary languages). Over $\Sigma = \{a\}$, consider

$$\begin{aligned} L_{\text{even}} &= \{a^{2j} : j \geq 0\}, \\ L_{\square} &= \{a^{j^2} : j \geq 0\}, \\ L_T &= \{a^T\}, \\ L_{\leq T} &= \{\varepsilon, a, \dots, a^T\}. \end{aligned}$$

All four are described by properties of length. The first, third, and fourth are regular; the square-length language is not. The distinction will become visible through their families of futures.

2.2 Left residuals: the future after a past

After reading a prefix u , the relevant question is which continuations still lead to acceptance.

Definition 2.2 (Left residual). For $u \in \Sigma^*$ and $L \subseteq \Sigma^*$, the left residual of L by u is

$$u^{-1}L = \{v \in \Sigma^* : uv \in L\}.$$

The prefix u is the past. The residual $u^{-1}L$ is its entire future language. Appending another word acts by further differentiation:

$$v^{-1}(u^{-1}L) = (uv)^{-1}L.$$

Notice the order: first read u , then v , so the combined prefix is uv . This language-derivative viewpoint goes back to Brzozowski's derivatives of regular expressions [6]; here we apply the residual operation to arbitrary languages.

Example 2.3 (Residuals of even length). For L_{even} over $\{a\}$,

$$(a^i)^{-1}L_{\text{even}} = \begin{cases} L_{\text{even}}, & i \text{ even}, \\ L_{\text{odd}}, & i \text{ odd}, \end{cases}$$

where $L_{\text{odd}} = \{a^{2j+1} : j \geq 0\}$. Infinitely many prefixes produce only two different futures.

Example 2.4 (Residuals of a singleton). For $L_T = \{a^T\}$, prefixes $\varepsilon, a, \dots, a^T$ yield the singleton futures

$$\{a^T\}, \{a^{T-1}\}, \dots, \{\varepsilon\}.$$

Every longer prefix yields the empty language. There are $T + 2$ residuals. This is finite for each fixed T , but the earliest prefix exposing the last live residual can be arbitrarily long as T varies.

Example 2.5 (Residuals of square lengths). For $L_{\square}$, the residual after a^i contains a^j precisely when $i + j$ is a square. Distinct i produce distinct futures. If $i < k$, choose a square $s \geq i$ whose next square is more than $k - i$ away, and set $j = s - i$. Then $i + j = s$ is square while $k + j$ lies strictly between consecutive squares. Thus the residual orbit is infinite.

2.3 The infinite past–future table

Arrange every prefix u as a row and every future v as a column:

$$H_L(u, v) = \mathbf{1}[uv \in L].$$

This is the Boolean Hankel table of L . Its row indexed by u is exactly the characteristic function of $u^{-1}L$. Two prefixes have the same row if and only if they have the same complete future.

The table can be cut in several ways. A fixed-length cut chooses $u \in \Sigma^m$ and $v \in \Sigma^n$. A bounded future view fixes N and keeps all columns with $|v| \leq N$. A bounded rectangle also restricts prefixes to $|u| \leq I$.

Definition 2.6 (Bounded future row). For horizon N , define

$$B_N(u) = (\mathbf{1}[uv \in L])_{|v| \leq N}.$$

Let

$$Q_N(L) = \{B_N(u) : u \in \Sigma^*\}, \quad q_L(N) = |Q_N(L)|.$$

Because there are only finitely many words of length at most N , the set of possible Boolean rows is finite. Thus $Q_N(L)$ is finite even when L has infinitely many exact residuals.

Forgetting the coordinates with $|v| = N + 1$ gives a surjection

$$\rho_{N+1,N} : Q_{N+1}(L) \twoheadrightarrow Q_N(L).$$

Consequently $q_L(N)$ is nondecreasing. A finer future observation can split old classes, never merge prefixes that were already distinguished.

Example 2.7 (The first square-length rows). Over the unary alphabet, $B_2(a^i)$ is the length-three Boolean block

$$(\mathbf{1}[i \text{ square}], \mathbf{1}[i + 1 \text{ square}], \mathbf{1}[i + 2 \text{ square}]).$$

For $i = 0, 1, 2, 3, 4$ the rows are respectively

$$110, \quad 100, \quad 001, \quad 010, \quad 100.$$

Prefixes a and a^4 look identical through horizon two even though their complete futures differ. Bounded equality is an approximation, not residual equality.

2.4 Observation as controlled forgetting

An observation is useful when it forgets deliberately. Complete-match language observation forgets parse trees, return locations, and alternative suffixes. Horizon- N observation forgets every distinction requiring a future longer than N . A fixed cut forgets how rows at one length relate to rows at another.

Forgetting is not inherently a defect. It creates finite objects that can be computed, compared, minimized, and used as evidence. The danger is forgetting which information was discarded and then treating the quotient as exact.

Boundary or warning

The number of distinct rows in each fixed-length cut does not determine the number of uniform residual states. For the square-length language, every fixed (m, n) cut is constant because every entry depends only on $m + n$. Each such cut has one row, while the complete residual orbit is infinite. Literal compatibility across cuts carries information that anonymous row counts erase.

2.5 Actions and readout

Residuals turn the set of futures into a dynamic system. Each letter $a \in \Sigma$ acts by

$$D_a(R) = a^{-1}R.$$

Starting at L , the word $u = a_1 \cdots a_k$ reaches

$$D_{a_k} \cdots D_{a_1}(L) = u^{-1}L.$$

The acceptance readout is uniform:

$$\text{out}(R) = \mathbf{1}[\varepsilon \in R].$$

Indeed, after reading u , the original word belongs to L exactly when $\varepsilon \in u^{-1}L$.

This pair—letter actions plus epsilon readout—is the canonical residual recognition dynamic of a language. It may be finite or infinite. The next monograph explains why finiteness is precisely regularity and why the finite case yields the minimal deterministic automaton.

Exercise 2.8. Compute every residual of $L = \{\varepsilon, a, ab\}$ over $\{a, b\}$. Draw the letter actions among them and mark the residuals containing ε . See the [solution](#).

Exercise 2.9. Compute $Q_1(L_\square)$ over the unary alphabet and find two prefixes that agree through horizon one but disagree through horizon two. See the [solution](#).

Exercise 2.10. Prove that a length-only language has at most one row in every nonempty fixed (m, n) cut. Explain why this statement alone gives no finite bound on its residual orbit. See the [solution](#).

Finite automata, reactivated

Finite automata are familiar enough that it is easy to remember their diagrams and forget the structural reason they matter. Their essential claim is not that a machine has circles and arrows. It is that every past can be summarized by one of finitely many states in a way that preserves all future acceptance decisions.

3.1 The deterministic model

Definition 3.1 (Deterministic finite automaton). A deterministic finite automaton (DFA) over Σ is a tuple

$$\mathcal{A} = (S, \Sigma, \delta, s_0, F),$$

where S is a finite set of states, $s_0 \in S$ is the start state, $F \subseteq S$ is the set of accepting states, and $\delta : S \times \Sigma \rightarrow S$ is the transition function.

The transition extends uniquely to words:

$$\begin{aligned}\delta^*(s, \varepsilon) &= s, \\ \delta^*(s, wa) &= \delta(\delta^*(s, w), a).\end{aligned}$$

The recognized language is

$$L(\mathcal{A}) = \{w : \delta^*(s_0, w) \in F\}.$$

The recursion says that a state is a sufficient summary of the past. If two prefixes reach the same state, appending any common future produces the same final state and hence the same acceptance decision.

Lemma 3.2 (State equality preserves every future). *If $\delta^*(s_0, u) = \delta^*(s_0, u')$, then for all $v \in \Sigma^*$,*

$$uv \in L(\mathcal{A}) \iff u'v \in L(\mathcal{A}).$$

Guided proof. Use associativity of extended transition:

$$\delta^*(s_0, uv) = \delta^*(\delta^*(s_0, u), v).$$

Replace the inner state by $\delta^*(s_0, u')$ and reverse the identity. Membership in F is therefore the same on both sides. $\square$

3.2 Reachability and useless names

A state s is *reachable* if $s = \delta^*(s_0, u)$ for some word u . Unreachable states cannot affect the recognized language and may be removed. This is the first simple distinction between a presentation and its reachable dynamic: a DFA file may name ten states while only six participate in recognition from the chosen start.

For a reachable state, a word u satisfying $\delta^*(s_0, u) = s$ is an *access word*. A shortest access word never repeats a state along its path; otherwise the intervening loop could be deleted. A reachable automaton with k states therefore reaches every state by a word of length at most $k - 1$.

Example 3.3 (Parity with an unnecessary state). Let $S = \{E, O, Z\}$ over $\{a\}$, start at E , toggle $E \leftrightarrow O$ on a , and let Z loop to itself. If $F = \{E\}$, the machine recognizes even length. State Z is unreachable and carries no semantic information from the start. Removing it leaves the same language.

Reachability is not minimization. Two reachable states can still have identical futures.

3.3 State futures and distinguishability

For state s , define its future language

$$L_s = \{v : \delta^*(s, v) \in F\}.$$

States s, t are equivalent when $L_s = L_t$ and distinguishable otherwise. A word v distinguishes them when exactly one of $\delta^*(s, v), \delta^*(t, v)$ is accepting.

The empty word distinguishes states with different acceptance readouts. A one-letter word may distinguish states whose immediate children have different readouts. Longer words propagate distinctions backward through the transition graph.

Example 3.4 (A nonminimal reachable DFA). Over $\{a, b\}$, let a machine remember whether the most recent symbol was a , but duplicate the “most recent was b ” state into two states selected by the parity of the prefix length. If acceptance depends only on whether the most recent symbol was a , the two b -states are reachable but have the same future language. The presentation remembers more history than recognition requires.

The standard table-filling minimization algorithm begins by separating accepting from rejecting states. It then repeatedly marks a pair (s, t) when some letter a sends it to the already marked pair

$$(\delta(s, a), \delta(t, a)).$$

At convergence, unmarked pairs have identical futures. Chapter 4 will prove this from residuals rather than treating it as an algorithmic fact.

3.4 Products and Boolean combinations

Given DFAs $\mathcal{A}_1$ and $\mathcal{A}_2$ over the same alphabet, their product has state set $S_1 \times S_2$ and acts componentwise:

$$\delta((s_1, s_2), a) = (\delta_1(s_1, a), \delta_2(s_2, a)).$$

Choosing the accepting pairs realizes intersection, union, symmetric difference, or any pointwise Boolean operation on the component acceptances.

The product construction is an upper bound, not usually a minimal machine. Some pairs may be unreachable; others may have identical joint futures. This gap between a convenient presentation and the canonical reachable quotient will recur throughout the book.

Proposition 3.5 (Regular languages are closed under Boolean operations). *If L_1 and L_2 are recognized by finite DFAs, then so are $L_1 \cap L_2$, $L_1 \cup L_2$, and $\Sigma^* \setminus L_1$.*

Proof walkthrough. Use the product for the binary operations. For intersection accept exactly $F_1 \times F_2$; for union accept pairs with at least one accepting component. Complement needs no product: keep the transition graph and replace F_1 by $S_1 \setminus F_1$. Determinism and total transitions ensure that exactly one run exists for every input. $\triangle$

3.5 What finiteness buys

A finite state set gives several kinds of leverage:

- every run can be simulated with bounded memory;
- reachability and equivalence are decidable by finite graph algorithms;
- sufficiently long runs repeat states and can be pumped;
- the family of future languages is finite; and
- bounded observations eventually stop discovering new exact futures.

These consequences explain why one might try to force an infinite recognition family into a finite quotient. The K_n example teaches a different lesson: when exact futures are genuinely infinite, we should ask which pieces of this leverage survive through structure, finite shadows, or coordinates rather than identify states that the observation can still distinguish.

Exercise 3.6. Prove carefully that every reachable state of a k -state DFA has an access word of length at most $k - 1$. Identify the step that uses minimality of the chosen access word, not minimality of the automaton. See the [solution](#).

Exercise 3.7. Construct two reachable two-state DFAs whose four-state product has fewer than four reachable states. Determine the language recognized by their intersection. See the [solution](#).

Residuals and the Myhill–Nerode mechanism

The residual construction removes accidental state names from deterministic recognition. It asks only: after this prefix, which futures are accepted? The answer produces both a canonical automaton and the exact boundary between regular and nonregular languages.

4.1 Future equivalence

For a fixed language L , define

$$u \equiv_L u' \iff \forall v \in \Sigma^*, uv \in L \iff u'v \in L.$$

Equivalently, $u^{-1}L = (u')^{-1}L$. This is an equivalence relation on Σ^* .

More is true. It is a *right congruence*: if $u \equiv_L u'$, then $ua \equiv_L u'a$ for every letter a . Indeed, for every v ,

$$uav \in L \iff u'av \in L$$

because av is one of the futures quantified in the definition.

Right congruence is exactly what makes a state transition well defined on equivalence classes. Define

$$[u]_L \xrightarrow{a} [ua]_L.$$

If a different representative u' is chosen, right congruence gives the same target class.

4.2 The canonical residual automaton

Let

$$\text{Res}(L) = \{u^{-1}L : u \in \Sigma^*\}.$$

Letters act by residual differentiation, the start state is L , and a state R is accepting exactly when $\varepsilon \in R$.

Theorem 4.1 (Residual recognition). *The residual dynamic recognizes L exactly.*

Guided proof. After reading u , repeated differentiation reaches $u^{-1}L$. The readout is one precisely when $\varepsilon \in u^{-1}L$. Unfold the residual definition:

$$\varepsilon \in u^{-1}L \iff u\varepsilon \in L \iff u \in L.$$

Thus the output after every word agrees with membership in L . □

Every residual is reachable by construction. The dynamic may still be infinite, as for square lengths or the continuation family of Chapter 1.

4.3 Finiteness is regularity

Theorem 4.2 (Myhill–Nerode theorem, residual form). *For a language $L \subseteq \Sigma^*$, the following are equivalent:*

1. L is recognized by a finite DFA;
2. L has finitely many left residuals;
3. the relation $\equiv_L$ has finite index.

When these conditions hold, the residual automaton is the unique minimal reachable DFA up to isomorphism.

Proof. The equivalence of (2) and (3) is immediate because two prefixes lie in the same $\equiv_L$ class exactly when they define the same residual.

Assume (2). The residual recognition theorem constructs a DFA whose state set is the finite set $\text{Res}(L)$, so (1) follows.

Assume (1), witnessed by a finite DFA $\mathcal{A}$. If prefixes u and u' reach the same state, Lemma 3.2 shows that they have identical futures and hence the same residual. Therefore the map

$$\delta^*(s_0, u) \mapsto u^{-1}L$$

from reachable DFA states onto residuals is onto. In particular, the number of residuals is no larger than the number of states, proving (2).

For minimality, let $\mathcal{A}$ be any reachable DFA recognizing L . Map a reachable state $s = \delta^*(s_0, u)$ to $u^{-1}L$. The value is independent of the chosen access word: if u and u' reach s , the state-equality lemma gives identical futures. The map preserves the start, letter transitions, and acceptance readout, and it is onto because every residual has a prefix representative. Hence every reachable DFA maps onto the residual automaton and has at least as many states.

If a reachable DFA has no two states with equal future language, the map is also injective and therefore an isomorphism. Any two minimal reachable DFAs are consequently isomorphic through their common residual identification. □

The theorem is not merely a counting result. It identifies the canonical meaning of a deterministic state: a complete future language.

Provenance

The finite-index characterization and canonical minimal automaton are classical Myhill–Nerode theory; Nerode’s automaton-transformation paper is an original source for the finite-index relation [14]. Brzozowski derivatives provide the equivalent language-differentiation presentation [6]. FPRD adopts this as its finite calibration; it does not claim the theorem as new.

4.4 Distinguishing words and proof by residual growth

To prove that a language is nonregular, it is enough to exhibit infinitely many prefixes with pairwise different residuals. For each pair one supplies a future accepted after one prefix and rejected after the other.

Example 4.3 ($a^n b^n$). Let $L = \{a^n b^n : n \geq 0\}$. The prefixes a^i are pairwise inequivalent. If $i < j$, the future b^i completes a^i but not a^j . Thus L has infinitely many residuals and is not regular.

Example 4.4 (Balanced parentheses). For the one-bracket Dyck language, a prefix with h unmatched opening parentheses requires at least h matching closers. Different heights have different completing futures. With several bracket types, the residual must remember the entire typed stack, not only its height.

Residual separation resembles a pumping argument but is often more direct. It states exactly what a finite-state summary would have to forget and supplies a future showing that the forgotten distinction still matters.

4.5 Presentation states versus residual states

A deterministic implementation may carry more state than the residual dynamic. Its state-to-residual map is a quotient. This fact separates three questions:

1. Is the implementation reachable?
2. Is it behaviorally minimal for its language?
3. Does its presentation carry additional named structure that we intend to preserve even when language observation forgets it?

The second question has the residual automaton as its exact answer. The third is one reason FPRD does not identify a finite grammar presentation with the minimal machine of its observed language.

Exercise 4.5. Compute the residual automaton of the unary language whose lengths are congruent to 1 modulo 3. Prove that no DFA with fewer than three states recognizes it. See the [solution](#).

Exercise 4.6. Show that the language $\{wcw^{\text{rev}} : w \in \{a, b\}^*\}$ has infinitely many residuals by finding a separator for every two distinct prefixes before the central marker c . See the [solution](#).

Finite horizons and observational shadows

Residual equality asks whether two prefixes agree on every possible future. Finite observation replaces “every” by “every future of length at most N .” The resulting quotients are always finite and form a tower that approaches exact residual behavior.

5.1 Horizon equivalence

Define

$$u \equiv_N u' \iff \forall v (|v| \leq N \Rightarrow (uv \in L \iff u'v \in L)).$$

The equivalence class of u is its row $B_N(u)$ from Chapter 2. The quotient is $Q_N(L)$.

Exact residual equality implies $u \equiv_N u'$ for every N . Conversely, if $u \equiv_N u'$ for all N , then any particular future v is covered once $N \geq |v|$, so u and u' have the same residual. Exact equality is thus the intersection of all horizon equivalences.

The relation $\equiv_{N+1}$ refines $\equiv_N$. Restriction gives a surjective bond

$$Q_{N+1}(L) \twoheadrightarrow Q_N(L).$$

Surjectivity is literal: every depth- N row is the restriction of the depth- $(N+1)$ row belonging to the same prefix.

Example 5.1 (Singleton horizons). For $L_T = \{a^T\}$, a prefix a^i with $i \leq T$ has its sole accepting future at distance $T - i$. At horizon $N < T$, the visible live distances are $0, 1, \dots, N$. Every more distant live prefix has an all-zero bounded row, as does every dead prefix, so together they form a single reject-all class. Hence

$$q_{L_T}(N) = N + 2 \quad (N < T).$$

Once $N \geq T$, every live residual is visible and $q_{L_T}(N) = T + 2$.

5.2 A graded action rather than a same-level machine

Appending a letter consumes one unit of future horizon. If $B_{N+1}(u) = B_{N+1}(u')$, then for every $|v| \leq N$,

$$uav \in L \iff u'av \in L,$$

so $B_N(ua) = B_N(u'a)$. Each letter therefore induces a well-defined map

$$Q_{N+1}(L) \longrightarrow Q_N(L).$$

It does not generally induce a map $Q_N \rightarrow Q_N$. Two prefixes may agree on all futures of length at most N while their a -children disagree on a future of length N ; that disagreement corresponds to a future of length $N + 1$ at the parent level.

Boundary or warning

It is tempting to call every $Q_N(L)$ a finite automaton. Without a genuine plateau or additional structure, it is only one level of a graded observation system. Its natural letter action arrives from level $N + 1$.

5.3 How far must prefixes be searched?

The quotient $Q_N(L)$ counts all bounded futures, including ones first exposed by very long prefixes. Define the bounded rectangle

$$X[I, N] = \{B_N(u) : |u| \leq I\}$$

and its size

$$G_L(I, N) = |X[I, N]|.$$

For fixed N , this count grows with I and eventually reaches $q_L(N)$.

Definition 5.2 (Prefix-access radius). The horizon- N access radius is

$$a_L(N) = \min\{I : G_L(I, N) = q_L(N)\}.$$

The minimum exists because $Q_N(L)$ is finite and every class has at least one literal prefix representative. The number $q_L(N)$ asks how many bounded futures exist. The number $a_L(N)$ asks how late the last of them first appears.

If L has a reachable k -state DFA, every residual state has an access word of length at most $k - 1$. Every bounded row is a projection of some residual, so

$$a_L(N) \leq k - 1$$

for every horizon.

Prefix counting gives a lower bound in the other direction. If $d = |\Sigma|$, there are at most

$$1 + d + d^2 + \dots + d^{a_L(N)}$$

prefixes in the access ball. Therefore

$$q_L(N) \leq \sum_{j=0}^{a_L(N)} d^j.$$

For unary languages this simplifies to $q_L(N) \leq a_L(N) + 1$.

Example 5.3 (Small count, late access). For $L_T = \{a^T\}$ at horizon zero, a row records only whether the prefix itself is accepted. There are two rows, accepting and rejecting, so $q_{L_T}(0) = 2$. The accepting row first appears at prefix length T , hence $a_{L_T}(0) = T$. Bounded cardinality does not control how far one must search to witness the classes.

5.4 Accessibility and distinguishability in a minimal DFA

For a finite minimal DFA, two classical depths sit on the two axes of the rectangle $G_L(I, N)$.

The *accessibility degree* is the maximum, over states, of the length of a shortest access word. Once horizon N is large enough to distinguish all residual states, $a_L(N)$ equals this degree.

The *distinguishability degree* is the maximum, over distinct state pairs, of the length of a shortest separating future. The first horizon at which $q_L(N)$ reaches the number of residual states is exactly this degree (with the one-state convention handled separately).

Thus the two-dimensional observation rectangle extends two familiar finite automaton notions. Before saturation it records partial coverage; for nonregular languages it continues indefinitely.

5.5 Why one observed plateau is delicate

Because $q_L(N)$ is nondecreasing, equality $q_L(N) = q_L(N+1)$ means the restriction bond is a bijection. This turns out to be much stronger than seeing the same number of rows in two finite samples of prefixes. A global adjacent plateau ranges over every prefix and will later force a right congruence. A finite membership table cannot certify that no longer prefix introduces a new row unless an independent state bound or equivalent guarantee is available.

This distinction is central to active automata learning. A finite observation table can suggest a machine, but an equivalence query or other completeness mechanism is needed to know that the table has captured the whole language [2]. The exact global statement appears as Theorem 18.1.

Example 5.4 (Access can plateau before the language does). For the one-bracket Dyck language, horizon zero and horizon one can both have access radius one: the currently visible rows appear immediately. Higher horizons reveal arbitrarily deep stack distinctions. Equality of two access values is not a finite-state certificate.

Exercise 5.5. Derive the complete formulas

$$q_{L_T}(N) = \min(T, N) + 2, \quad a_{L_T}(N) = \begin{cases} T, & N < T, \\ T + 1, & N \geq T. \end{cases}$$

Identify the dead row responsible for the final access step. See the [solution](#).

Exercise 5.6. Construct a language and prefixes u, u' with $u \equiv_0 u'$ but $ua \not\equiv_0 u'a$. Explain why this does not contradict the well-defined map $Q_1 \rightarrow Q_0$. See the [solution](#).

Part II

Topology for infinite behavior

Topological spaces: nearness without distance

Topology studies which distinctions can be made by local observation and how those distinctions behave under limits. It can be introduced as geometry without measurements, but for recognition dynamics a different intuition is more useful: a neighborhood specifies a finite amount of information that we insist on observing, while leaving the rest unresolved.

We begin far from inverse limits. The objective is to recover a working feel for open and closed sets, continuity, and convergence across several kinds of space.

6.1 Open sets as permitted observations

Definition 6.1 (Topology). A topology on a set X is a collection $\mathcal{T} \subseteq \mathcal{P}(X)$ such that

1. $\emptyset, X \in \mathcal{T}$;
2. arbitrary unions of members of $\mathcal{T}$ belong to $\mathcal{T}$;
3. finite intersections of members of $\mathcal{T}$ belong to $\mathcal{T}$.

The pair $(X, \mathcal{T})$ is a topological space, and members of $\mathcal{T}$ are open sets.

An open set is not intrinsically fuzzy or boundaryless. It is one of the subsets the topology permits us to recognize locally. The axioms say that a choice among any family of observable alternatives is observable (union), and that finitely many simultaneous observational requirements can be combined (finite intersection).

Why only finite intersections? In the real line, every interval $(-1/n, 1/n)$ is open, but their infinite intersection is $\{0\}$, which is not open. Demanding infinitely many increasingly precise conditions can identify an exact point even though no finite stage does.

Example 6.2 (The usual real topology). On $\mathbb{R}$, open intervals (a, b) generate the usual topology: a set is open when every point in it lies inside some interval contained in the set. The interval is a finite-precision witness that nearby points remain inside.

Example 6.3 (Discrete and indiscrete extremes). The discrete topology is $\mathcal{P}(X)$: every subset is open. It makes every point perfectly observable because each singleton $\{x\}$ is open. The indiscrete topology is $\{\emptyset, X\}$: no proper nonempty part can be detected by an open observation. These extremes show that topology records observational resolution, not merely the underlying set.

Example 6.4 (Cofinite topology). If X is infinite, declare U open when $U = \emptyset$ or $X \setminus U$ is finite. Any two nonempty open sets intersect: each omits only finitely many points. Individual points are closed but not open. A sequence of distinct points converges to every point, because it eventually avoids the finite complement of each neighborhood. This unfamiliar behavior is legal topology and warns us not to import metric intuition without checking hypotheses.

6.2 Closed sets and limits

A set $C \subseteq X$ is *closed* when its complement $X \setminus C$ is open. Closed sets satisfy the dual laws: $\emptyset$ and X are closed, arbitrary intersections of closed sets are closed, and finite unions of closed sets are closed.

In metric spaces, closed sets are exactly the sets containing all limits of their convergent sequences. In arbitrary spaces this sequential criterion can fail, but the intuition remains useful: closedness prevents a set from losing limit points visible to the topology.

Example 6.5 (Closed intervals). The real interval $[0, 1]$ is closed because its complement $(-\infty, 0) \cup (1, \infty)$ is open. A convergent sequence of points in $[0, 1]$ cannot converge in $\mathbb{R}$ to a point outside the interval.

6.2.1 How can a set be both open and closed?

A *clopen* set is both open and closed. There is no contradiction: open and closed are not mutually exclusive adjectives like finite and infinite. They describe a set and its complement. A set is clopen precisely when it and its complement are both observable as open alternatives.

The sets $\emptyset$ and X are always clopen. In a discrete space every set is clopen. In the usual real line, no other sets are clopen because $\mathbb{R}$ is connected.

Example 6.6 (A disconnected space). Let $X = (-2, -1) \cup (1, 2)$ with the topology inherited from $\mathbb{R}$. The left component $(-2, -1)$ is open in X . Its complement inside X is the right component, also open, so the left component is closed in X as well. The same subset is not closed as a subset of the ambient real line. Topology is always relative to the space under discussion.

Clopen sets will be central for finite observations. In a product of discrete Boolean coordinates, fixing finitely many coordinate values defines a clopen cylinder. One can decide both membership and nonmembership using the same finite observation.

6.3 Bases and neighborhoods

Listing every open set is often impractical. A *basis* is a collection $\mathcal{B}$ of subsets such that every open set is a union of basis elements and, whenever $x \in B_1 \cap B_2$, some $B_3 \in \mathcal{B}$ satisfies $x \in B_3 \subseteq B_1 \cap B_2$.

Open intervals form a basis for $\mathbb{R}$. Open balls form a basis for a metric space. Finite-coordinate cylinders form a basis for the product spaces we will use.

A *neighborhood* of x is a set containing an open set that contains x . Neighborhood language lets us state local properties without choosing a particular basis.

Definition 6.7 (Closure). The closure $\overline{A}$ of $A \subseteq X$ is the intersection of all closed sets containing A . Equivalently, $x \in \overline{A}$ exactly when every open neighborhood of x meets A .

The second formulation is observational: x lies in the closure of A when no finite local observation around x can exclude all of A .

Example 6.8 (A new boundary point). In $\mathbb{R}$, let $A = \{1, 1/2, 1/3, \dots\}$. Every neighborhood of 0 contains $1/n$ for sufficiently large n , so $0 \in \overline{A}$. No other new point is forced. Thus $\overline{A} = A \cup \{0\}$.

The recognition orbit $\{K_n : n \in \mathbb{N}\}$ behaves similarly under bounded-input observation. As n escapes every fixed horizon, the states converge to a boundary behavior that rejects every finite word. The boundary is not one of the K_n , just as 0 is not one of the positive reciprocals.

6.4 Continuity

Definition 6.9 (Continuous map). A map $f : X \rightarrow Y$ is continuous if $f^{-1}(U)$ is open in X for every open $U \subseteq Y$.

The inverse-image definition is chosen because inverse images preserve arbitrary unions and finite intersections automatically. Conceptually, a continuous map cannot require infinitely more input precision to decide a finite output observation. Every observable distinction in the target pulls back to an observable distinction in the source.

Example 6.10 (Maps out of a discrete space). Every map from a discrete space is continuous because every inverse image is a subset and hence open. A map into a discrete space is continuous only when the inverse image of every singleton is open; it must make each exact output locally decidable.

Proposition 6.11. *The composition of continuous maps is continuous.*

Guided proof. Let $f : X \rightarrow Y$ and $g : Y \rightarrow Z$ be continuous, and let U be open in Z . Then

$$(g \circ f)^{-1}(U) = f^{-1}(g^{-1}(U)).$$

Continuity of g makes $g^{-1}(U)$ open in Y ; continuity of f makes the remaining inverse image open in X . $\square$

6.5 Convergence and separation

A sequence (x_n) converges to x if every neighborhood of x contains all but finitely many x_n . In a metric space this agrees with the familiar $d(x_n, x) \rightarrow 0$.

Limits need not be unique in an arbitrary topology. In the indiscrete space, every sequence converges to every point because the only neighborhood condition is eventual membership in X . A *Hausdorff* space separates distinct points by disjoint neighborhoods; in such spaces sequence limits are unique.

All finite discrete spaces are Hausdorff. Their products are Hausdorff. The compact recognition spaces constructed later therefore retain ordinary distinctness even though their topology is generated by finite observations.

Intuition

Topology lets us say that two exact behaviors are close when no short experiment separates them. Closedness says a family contains the behaviors forced as limits of its approximations. Continuity says a recognition action respects finite observational resolution. Compactness, introduced next, will say that infinitely many compatible finite demands cannot evade all global realizations.

Exercise 6.12. For each of the following, determine all clopen sets: a three-point discrete space; a three-point indiscrete space; and the subspace $[0, 1] \cup [2, 3] \subseteq \mathbb{R}$. See the [solution](#).

Exercise 6.13. Let X have the cofinite topology. Prove that every injective sequence converges to every point of X . Which part of this conclusion becomes impossible in a Hausdorff space? See the [solution](#).

Exercise 6.14. Let A be the set of eventually-zero Boolean sequences inside $\{0, 1\}^{\mathbb{N}}$. Anticipating the product topology of the next chapter, argue that every finite Boolean prefix occurs in some member of A . What should the closure of A be? See the [solution](#).

Products and compactness

Recognition behavior is naturally coordinatewise. A language assigns one Boolean value to every word. A bounded observation sees finitely many of those values. Product topology is precisely the topology in which fixing finitely many coordinates describes a basic observation.

Compactness then supplies the global principle we need: when finite pieces of information remain mutually compatible, they cannot drift away without leaving a global limit point.

7.1 The product topology

Let $(X_i)_{i \in I}$ be topological spaces. Their Cartesian product

$$X = \prod_{i \in I} X_i$$

consists of tuples $x = (x_i)_{i \in I}$. For each coordinate there is a projection $\pi_i(x) = x_i$.

Definition 7.1 (Product topology). The product topology is the smallest topology on X making every projection $\pi_i : X \rightarrow X_i$ continuous.

A basis element restricts finitely many coordinates and leaves all others free:

$$U = \prod_{i \in I} U_i, \quad U_i \subseteq X_i \text{ open}, \quad U_i = X_i \text{ for all but finitely many } i.$$

Such a set is often called a cylinder.

Why finitely many restrictions? The product topology is designed so that an open observation requests only finitely much coordinate information. If infinitely many exact coordinate conditions were basic open sets, convergence would become much stricter and finite observations would no longer generate the topology.

Example 7.2 (The Cantor space of Boolean sequences). Take $X_i = \{0, 1\}$ with the discrete topology for every $i \in \mathbb{N}$. A point of $2^{\mathbb{N}}$ is an infinite Boolean sequence. Fixing the first N bits gives the cylinder

$$[b_0 \cdots b_N] = \{x : x_0 = b_0, \dots, x_N = b_N\}.$$

It is clopen: the complement is the finite union of cylinders where at least one fixed bit is flipped.

The same idea works when coordinates are indexed by words rather than natural numbers. Since Σ^* is countable, choose any enumeration $w_0, w_1, \dots$. A language becomes a point

$$\chi_L \in 2^{\Sigma^*}.$$

A cylinder prescribes membership for finitely many words. The bounded-input topology uses the especially natural cofinal family of cylinders fixing all words through a given length.

7.2 Convergence in a product

A sequence $x^{(n)}$ in a product converges to x exactly when it converges in every coordinate, provided the index set is countable and we use sequences. For discrete coordinate spaces, coordinate convergence means eventual equality.

Thus languages L_n converge to L in 2^{Σ^*} when, for every fixed word w , membership $w \in L_n$ eventually agrees with $w \in L$. Equivalently, for every length bound N , the languages eventually agree on every word of length at most N .

Example 7.3 (Finite singletons escaping to the empty language). Over $\{b\}$, let $L_n = \{b^n\}$. Fix a word b^j . For all $n > j$, $b^j \notin L_n$, so the coordinate eventually agrees with the empty language. Hence $L_n \rightarrow \emptyset$. This is the boundary behavior of the continuation family K_n .

7.2.1 An ultrametric for bounded observation

The topology can be metrized directly. For distinct languages L, L' , let

$$n(L, L') = \min\{|w| : \mathbf{1}[w \in L] \neq \mathbf{1}[w \in L']\}$$

and define

$$d(L, L') = 2^{-n(L, L')}, \quad d(L, L) = 0.$$

Then $d(L, L') < 2^{-N}$ means the two languages agree through a corresponding finite horizon, with the exact indexing convention adjusted at equality.

This metric is an *ultrametric*:

$$d(L_1, L_3) \leq \max\{d(L_1, L_2), d(L_2, L_3)\}.$$

If L_1 and L_2 agree through one horizon and L_2 and L_3 agree through another, then L_1 and L_3 agree through the smaller of the two horizons. Ultrametric balls are clopen, reflecting the finite-decision nature of bounded observations.

7.3 Compactness from open covers

Definition 7.4 (Compact space). A topological space X is compact if every open cover of X has a finite subcover. That is, whenever $X = \bigcup_{i \in I} U_i$ with every U_i open, there exist $i_1, \dots, i_k$ such that $X = U_{i_1} \cup \dots \cup U_{i_k}$.

In Euclidean space, closed bounded sets are compact by the Heine–Borel theorem. But compactness is not synonymous with closed-and-bounded outside that setting. The open-cover definition is topological and travels to product spaces and inverse limits.

Compactness often turns an infinite family of local possibilities into a finite witness for a global statement. If each point has some neighborhood on which a property is controlled, a finite subcover can sometimes make the control uniform. If an infinite collection of closed constraints is finitely compatible, compactness can force a point satisfying all of them.

Theorem 7.5 (Finite intersection characterization). *A space X is compact if and only if every family of closed sets $\{C_i : i \in I\}$ with the finite intersection property has nonempty total intersection. The finite intersection property means*

$$C_{i_1} \cap \cdots \cap C_{i_k} \neq \emptyset$$

for every finite subfamily.

Proof. Assume X is compact and suppose the closed sets have empty total intersection. Their open complements cover X :

$$X = \bigcup_{i \in I} (X \setminus C_i).$$

A finite subcover would imply $C_{i_1} \cap \cdots \cap C_{i_k} = \emptyset$, contradicting the finite intersection property.

Conversely, assume the closed-set property. Let (U_i) be an open cover with no finite subcover. The closed complements $C_i = X \setminus U_i$ then have the finite intersection property: an empty finite intersection would give a finite subcover. Their total intersection is empty because the U_i cover X , contradicting the assumption. $\square$

7.4 Why compactness resembles finiteness

Every finite space is compact: from an open cover, choose one set containing each point. Compact spaces may be infinite, but open-cover arguments allow many finite-space proofs to survive after replacing literal enumeration by a finite subcover or finite-intersection argument.

Compactness is useful for recognition dynamics for three recurring reasons.

1. An infinite orbit may acquire boundary points without becoming uncontrollably large; its closure inside a compact behavior space remains compact.
2. A compatible family of finite observations can be realized by a global behavior when the corresponding closed constraints have the finite intersection property.
3. Continuous actions on a dense orbit extend uniquely in situations where their finite observational behavior is coherent.

Compactness does *not* imply finite-state recognition. The Cantor space $2^{\mathbb{N}}$ is compact and uncountable. The closure of an infinite residual orbit can be compact while the language remains nonregular. Compactness is a form of global control, not a disguised state bound.

Boundary or warning

Do not infer computational tractability from compactness alone. Compactness can prove existence without providing an effective representation, algorithm, or finite-dimensional coordinate system. FPRD separately tracks finite presentation, finite observation, action, and uniform readout.

7.5 Compactness of a countable product of finite spaces

The full Tychonoff theorem states that arbitrary products of compact spaces are compact. Its general proof requires a choice principle. Our principal behavior spaces are countable products of finite discrete spaces, where a direct diagonal argument is available.

Theorem 7.6. *Let X_n be a nonempty finite discrete space for every $n \in \mathbb{N}$. Then $\prod_{n \in \mathbb{N}} X_n$ is compact.*

Proof walkthrough. Take any sequence of product points. Because X_0 is finite, infinitely many points share one value in coordinate zero; pass to that subsequence. Within it, infinitely many share one value in coordinate one; pass to a further subsequence. Continue.

Choose the diagonal subsequence whose k th member comes from the subsequence stabilized through coordinate k . For each fixed coordinate j , all sufficiently late diagonal members have the selected value in coordinate j . The diagonal subsequence therefore converges coordinatewise to a point of the product.

We have proved sequential compactness. The countable product is metrizable by a first-difference metric like the ultrametric above. In metric spaces, sequential compactness is equivalent to compactness. That final equivalence is a standard theorem; its proof extracts a sequence from a cover with no finite subcover and conversely uses finite-radius nets. $\triangle$

Since Σ^* is countable and each coordinate $\{0, 1\}$ is finite, 2^{Σ^*} is compact. It is also Hausdorff, totally disconnected, and metrizable by bounded-input agreement.

7.6 Closed subspaces and orbit closures

A closed subset of a compact space is compact. To see this, cover the closed set C by open sets in X and add the open complement $X \setminus C$; a finite subcover of X leaves a finite subcover of C after discarding the complement.

For a language L , its residual orbit

$$O_L = \{u^{-1}L : u \in \Sigma^*\}$$

sits inside the compact behavior space 2^{Σ^*} . Its closure

$$X_L = \overline{O_L}$$

is therefore compact. When O_L is finite it is already closed. When it is infinite, the closure may add limit behaviors.

Provenance

The completed residual space X_L is the minimal topological automaton in Steinberg’s compact-automata construction [20]: a language’s residual orbit is embedded in the Boolean behavior space and then closed. Our use of this object inside the FPRD observation tower is a reconstruction and extension of viewpoint, not a claim that the orbit-closure construction is new.

The orbit is dense in X_L by the definition of closure. Density means every finite observation of a completion point is matched by some literal residual. It does not mean every completion point is a residual.

Exercise 7.7. In $2^{\mathbb{N}}$, prove that a cylinder fixing coordinates 2, 5, 9 is clopen. Write its complement explicitly as a finite union of cylinders. See the [solution](#).

Exercise 7.8. Prove that the eventually-zero Boolean sequences are dense in $2^{\mathbb{N}}$ but not closed. Identify their closure. See the [solution](#).

Exercise 7.9. Give three different reasons why $2^{\mathbb{N}}$ is not finite, then explain which finite-space proof pattern compactness recovers and which it does not. See the [solution](#).

Inverse limits: assembling compatible shadows

An inverse limit is a space of globally compatible finite views. The phrase can sound forbidding because it belongs to category theory and topology, but the basic construction is concrete: take one observation at every resolution and require that forgetting detail gives the observation chosen at the coarser resolution.

8.1 Inverse systems

An inverse sequence consists of spaces

$$X_0 \xleftarrow{\rho_{1,0}} X_1 \xleftarrow{\rho_{2,1}} X_2 \leftarrow \cdots .$$

For $m > n$, write

$$\rho_{m,n} = \rho_{n+1,n} \circ \cdots \circ \rho_{m,m-1}.$$

The maps are called bonding maps because they state how a fine view restricts to a coarse one.

Definition 8.1 (Inverse limit). The inverse limit is

$$\varprojlim_n X_n = \left\{ (x_0, x_1, \dots) \in \prod_{n \geq 0} X_n : \rho_{n+1,n}(x_{n+1}) = x_n \text{ for all } n \right\}.$$

A point of the limit is a compatible thread.

The construction is characterized by a mapping property. This matters because it tells us that the coordinate formula is canonical rather than an arbitrary encoding.

Proposition 8.2 (Universal property of the inverse limit). *Let Y be a set and suppose maps $f_n : Y \rightarrow X_n$ form a compatible cone:*

$$\rho_{n+1,n} \circ f_{n+1} = f_n \quad \text{for every } n.$$

There is a unique map

$$f : Y \longrightarrow \varprojlim_n X_n$$

such that $\pi_n \circ f = f_n$ for every coordinate projection π_n . If the spaces are topological and every f_n is continuous, then f is continuous.

Guided proof. The equations force the only possible definition:

$$f(y) = (f_0(y), f_1(y), \dots).$$

Compatibility of the f_n says precisely that this tuple belongs to the inverse limit. Any other map with the same coordinate projections has the same value in every coordinate and therefore is equal to f . For the topological claim, a map into a product is continuous exactly when all its coordinate maps are continuous; its image already lies in the inverse-limit subspace. $\square$

Example 8.3 (Infinite sequences from finite prefixes). Let $X_n = \{0, 1\}^{n+1}$ and let $\rho_{n+1,n}$ delete the final bit. A compatible thread chooses a one-bit word, a two-bit word extending it, a three-bit word extending that, and so on. Such a thread is exactly one infinite Boolean sequence. Thus

$$\varprojlim_n \{0, 1\}^{n+1} \cong \{0, 1\}^{\mathbb{N}}.$$

The inverse limit is not a mysterious point “at infinity.” It is the set of all coherent choices across every finite level.

8.2 Why compatibility matters

Choosing arbitrary $x_n \in X_n$ produces an element of the product but not necessarily an element of the inverse limit. The thread might claim that the first two bits are 01 at one level and that the first three bits are 111 at the next. Each finite object is legal; the family is jointly impossible.

This distinction becomes essential in local-to-global reasoning. A list of individually realizable finite observations need not have a common global realization. Bonding equations are the first consistency conditions. Later, literal cut compatibility will play the same role across two axes.

Example 8.4 (Decimal truncations). Let $X_n = \{0, 1, \dots, 10^{n+1} - 1\}$ and let the bond discard the final decimal digit. Compatible threads look like infinite digit sequences. If one tries to interpret them as real numbers in $[0, 1]$, the familiar dual expansions $0.4999\dots = 0.5000\dots$ show that the coordinate presentation and the represented object need not be identical. An inverse limit records the chosen compatible coordinates; any further quotient must be stated.

8.3 Topology of the limit

Give $\prod_n X_n$ the product topology and the inverse limit the subspace topology. If every X_n is Hausdorff and every bonding map is continuous, the inverse limit is closed in the product.

Proposition 8.5 (Closedness of compatibility). *For continuous maps between Hausdorff spaces, the set of compatible threads is closed in $\prod_n X_n$.*

Guided proof. For each n , define

$$C_n = \{(x_j)_j : \rho_{n+1,n}(x_{n+1}) = x_n\}.$$

The map sending a product point to $(\rho_{n+1,n}(x_{n+1}), x_n)$ is continuous. The diagonal $\{(y, y) : y \in X_n\}$ is closed in $X_n \times X_n$ because X_n is Hausdorff. Hence C_n is closed as the inverse image of the diagonal. The inverse limit is $\bigcap_n C_n$, an arbitrary intersection of closed sets. $\square$

For the finite discrete systems used throughout this book, the product is compact by the countable finite-discrete product theorem proved in Chapter 7. The closed inverse limit is therefore compact. A corresponding theorem for arbitrary compact Hausdorff factors is an instance of the more general Tychonoff theorem; we do not need or prove that stronger result here.

8.4 Existence of compatible threads

Compactness becomes useful when we ask whether the inverse limit is nonempty. For a sequence of nonempty finite levels, compactness guarantees a thread even without surjective bonds: every finite initial chain can be built downward from its highest level. Surjectivity supplies the stronger fact that every chosen coarse point extends to a thread.

Theorem 8.6 (Finite discrete inverse-limit existence). *Let every X_n be a nonempty finite discrete space, with arbitrary bonding maps $\rho_{n+1,n} : X_{n+1} \rightarrow X_n$. Then $\varprojlim_n X_n$ is nonempty.*

Proof. Work in the compact product $P = \prod_n X_n$. For each k , let

$$D_k = \{(x_n)_n : \rho_{n+1,n}(x_{n+1}) = x_n \text{ for } 0 \leq n < k\}.$$

Each D_k is closed by the previous argument. It is nonempty: choose any $x_k \in X_k$, define lower coordinates by restriction, and choose arbitrary higher coordinates. More generally, every finite intersection of the D_k is just the strongest one among them and is nonempty. Compactness and the finite intersection property give

$$\bigcap_{k \geq 0} D_k \neq \emptyset.$$

That intersection is the inverse limit. □

Surjectivity was not used: we built each finite compatible prefix downward from its highest constrained level. If all bonds are surjective, however, then each projection from the inverse limit onto X_n is surjective. Given a specified x_n , choose successive preimages above it and use the bonds to define the coordinates below it. In our quotient towers surjectivity has an even more concrete meaning: every coarse observation is the restriction of the same literal prefix viewed more finely.

8.5 Finite language observations as an inverse system

For a language L , take $X_n = Q_n(L)$ and let restriction forget the longest future coordinates. Every bond is surjective. A literal residual $u^{-1}L$ determines the compatible thread

$$(B_0(u), B_1(u), B_2(u), \dots).$$

Two exact residuals give different threads because some finite future distinguishes them.

The inverse limit may contain threads not generated by any single prefix. For the family $K_n = \{b^n\}$, take the tail class at every horizon. These choices are compatible. No finite K_j belongs to the tail once the horizon reaches j , so the thread is not a literal orbit state. It represents the boundary language $\emptyset$.

Intuition

The inverse limit performs two jobs at once. It remembers every finite observation without choosing a privileged resolution, and it completes the literal orbit by admitting behaviors approximated at all finite depths. The first job is reconstruction; the second is completion. They coincide only when every compatible thread is already generated by a literal state.

8.6 Natural actions on compatible threads

Suppose each level has maps

$$f_n : X_{n+e} \rightarrow X_n$$

for a fixed exposure e , and these commute with restriction:

$$\rho_{n,m} \circ f_n = f_m \circ \rho_{n+e,m+e}.$$

Then a compatible thread $x = (x_n)$ determines another thread by

$$f(x)_n = f_n(x_{n+e}).$$

Naturality is exactly what proves the result is compatible. Conversely, a continuous exact action whose output through depth n depends only on input through depth $n + e$ yields such a shifted family.

Letter derivatives have exposure one: to know the future of $a^{-1}L$ through length n , one needs the future of L through length $n + 1$. Same-level maps have exposure zero. Exposure measures observational lookahead, not time complexity.

8.7 Orbit, completion, and closedness

Embed the residual orbit O_L into the inverse limit by its bounded rows. The image is dense in the compatible-thread space associated with the orbit's finite quotients. Under the natural identification with $\overline{O_L} \subseteq 2^{\Sigma^*}$, the inverse-limit topology is exactly bounded-input agreement.

If the residual orbit is finite, it is closed in the Hausdorff behavior space, so completion adds no states. Conversely, an infinite orbit can sometimes be closed as an abstract subset, but for residual orbits over a finite alphabet with the compact observational structure studied later, closedness and the finite-state plateau mechanism are tightly related. We will prove the exact regularity statement only after the local-to-global sewing construction has made the actions available at one stable level.

Boundary or warning

An inverse limit is a canonical completion relative to the selected finite observations. It is not automatically a finite-coordinate realization, an efficient data structure, or the exact dynamic generated by a grammar presentation. Those are separate obligations.

Exercise 8.7. Write the first four levels and bonding maps of the finite-prefix system for Boolean sequences. Construct the thread corresponding to 101000.... See the [solution](#).

Exercise 8.8. Give a sequence of individually valid finite binary prefixes that does not form a compatible thread. Identify the first failed bonding equation. See the [solution](#).

Exercise 8.9. For the K_n observation tower, prove that the all-tail thread corresponds to the empty language under coordinatewise membership. Show directly that $K_n \rightarrow \emptyset$ in the bounded-input topology. See the [solution](#).

Part III

Finite presentations and generated dynamics

Four layers of recognition dynamics

We now return to the program’s central object. The preceding monographs let us state its architecture without using “state” to mean four different things. The layers are semantic behavior, observation, finite presentation, and coordinate realization. Each answers a different question, and each has a different notion of equivalence.

9.1 Layer one: exact semantic behavior

A deterministic parser can be modeled by what it does to every input word. On success it returns an unconsumed suffix; on failure it returns no suffix. Let

$$\text{Suf}(w) = \{v : \exists u, w = uv\}$$

be the suffixes of w .

Definition 9.1 (Suffix-decreasing behavior). A deterministic recognition behavior is a partial function

$$f : \Sigma^* \rightarrow \Sigma^*$$

such that whenever $f(w)$ is defined, $f(w)$ is a suffix of w .

Failure is undefinedness. Epsilon succeeds without consuming: $\text{id}(w) = w$. A terminal a removes a leading a when present. Sequential composition is ordinary partial-function composition in parser order. Ordered choice f/g tries f and, only when f fails, applies g to the original input.

Example 9.2 (Consumption versus complete match). The terminal- a behavior maps ab to b and fails on bb . Its complete-match language is only $\{a\}$, because input ab leaves a nonempty suffix. The behavior contains strictly more information than its language.

For a nondeterministic or context-free doctrine, replace a partial suffix by a set of suffixes:

$$R : \Sigma^* \rightarrow \mathcal{P}(\Sigma^*), \quad R(w) \subseteq \text{Suf}(w).$$

Relational sequence composes possible intermediate suffixes. Commutative union collects alternatives. This differs from prioritized choice even when complete-match languages happen to agree.

The exact semantic layer decides what behaviors and operations mean before any finite approximation is chosen.

9.2 Layer two: observation and completion

An observation asks only part of the exact behavior. Complete match reads whether ε is among the successful suffixes. Bounded observation restricts inputs to length at most N . A finite table therefore identifies exact behaviors that agree on the current bounded domain.

Restriction from depth $N + 1$ to depth N creates an inverse system of finite quotients. Its inverse limit is the space of compatible finite observations. When every legal compatible table is realized by some exact behavior, the limit reconstructs the ambient semantic space. When we restrict to an orbit generated by selected actions, the limit reconstructs its observational closure.

Example 9.3 (Same bounded view, different exact behavior). The singleton languages $\{a^{N+1}\}$ and $\{a^{N+2}\}$ both reject every word of length at most N . They are equal at depth N and separate at depth $N + 1$. A bounded quotient is correct about what it sees and silent about the rest.

Topology enters because agreement on larger finite domains defines nearness. Compactness can turn compatible finite constraints into a global limit. It does not say that the exact semantics has a finite presentation or tractable coordinates.

9.3 Layer three: finite presentations and generated actions

A presentation names finitely many constructors, equations, nonterminals, or call sites. Solving it yields exact semantic behaviors, but the presentation can retain structure not recoverable from the final language.

For a grammar call occurring inside a sequence, the material to its right is a continuation frame. A labeled call site therefore generates an action on continuation behavior. Two extensionally equivalent grammars may have different labeled calls, different frame composition, or different action orbits.

Example 9.4 (An inert presentation distinction). A grammar may contain a labeled call in a branch that can never succeed from the start symbol. Its complete-match language is unaffected. Whether the call belongs to the generated dynamic depends on whether the presentation layer uses raw syntax, reachable semantics, or another explicit policy. The choice cannot be recovered from the language alone.

Presentation equivalence is therefore not automatically language equivalence or dynamic isomorphism. A normalization may preserve exact suffix semantics while splitting one source call into several target calls or merging several source occurrences. The correct comparison may be a finite origin/refinement relation rather than a bijection of action labels.

9.4 Layer four: coordinate realization and readout

An injective encoding of a state set is cheap: enumerate states and assign integers or real numbers. A useful realization must make the relevant actions and recognition observation uniformly accessible.

Definition 9.5 (Readout-aware realization). Informally, a coordinate realization of a recognition dynamic consists of an injective state map $\phi : X \rightarrow C$, coordinate actions $\hat{a} : C \rightarrow C$ for the named generators, and a uniform readout procedure such that

$$\phi(a \cdot x) = \hat{a}(\phi(x))$$

and the original recognition observation can be recovered from $\phi(x)$ and the input under consideration.

The adjective “tame” must be made specific. It may mean finite state, counter-like, rational, continuous, Lipschitz, polynomial-time, bounded exposure, or something else. FPRD does not declare one coordinate class universal in advance.

Example 9.6 (A counter realization). The running state family $K_n = \{b^n\}$ uses $C = \mathbb{N}$, successor action $n \mapsto n + 1$, and readout $\text{accept}(n, w) = \mathbf{1}[w = b^n]$. It is infinite, finitely generated, and exactly readable. Mapping K_n instead to an arbitrary noncomputable real sequence would preserve distinctness while destroying the point of the representation.

9.5 Three equivalence questions

Given two systems, ask separately:

1. Do they have the same observed language or suffix behavior?
2. Are their presentations related by a structure-preserving transformation?
3. Are their generated dynamics conjugate, quotiented, refined, or merely observation-equivalent?

A theorem answering one question does not silently answer the others. This discipline prevents an extensional normal form from erasing presentation provenance and prevents an elegant coordinate embedding from being called a recognizer before readout is supplied.

Provenance

Partial-function algebra, relational semantics, inverse limits, residual automata, and guarded recursion all have substantial prior theory. The FPRD program’s claim is not to originate these ingredients. Its prospective contribution lies in their explicit recognition-specific integration with finite presentation, generated actions, finite observations, and uniform readout.

Research pause

Suppose two grammars have the same complete-match language and isomorphic minimal residual DFAs, but one grammar exposes a useful counter action and the other exposes an opaque family of contexts. Should a representation theory identify them? Which question are you trying to answer when you decide?

Exercise 9.7. Classify each of the following as semantic, observational, presentational, or coordinate data: a returned suffix; acceptance of the empty suffix; a labeled call occurrence; a shortest distinguishing word; an integer counter; and a restriction map between finite tables. Some items legitimately touch two layers; explain why. See the [solution](#).

Recursion, guarding, and contraction

A finite recursive presentation can denote a unique infinite behavior, several behaviors, or no operationally meaningful behavior at all. Guarding is the mechanism that turns recursion from circular syntax into controlled approximation.

10.1 Recursive equations

Consider the language equation

$$X = AX \cup B,$$

where $A, B \subseteq \Sigma^*$. If $\varepsilon \notin A$, every recursive use of X is preceded by at least one terminal symbol. The familiar solution is

$$X = A^*B.$$

The equation can be unfolded from $X_0 = \emptyset$:

$$X_{n+1} = AX_n \cup B.$$

Inductively,

$$X_n = B \cup AB \cup \dots \cup A^{n-1}B.$$

For words through length N , sufficiently many unfoldings are exact because each recursive layer consumes positive length.

Provenance

The solution of $X = AX \cup B$ under $\varepsilon \notin A$ is commonly called Arden's rule [3]. The contraction theorem below is Banach's fixed-point principle [5]. Their use for bounded-input guarded recognition is an adaptation of classical language equations and metric semantics, not a claim that these fixed-point tools are new.

Example 10.1 (The equation $X = aX \cup \{b\}$). The solution is $\{a^n b : n \geq 0\}$. Unfolding n contains the first n members. Through input length N , no more than $N + 1$ unfoldings can matter.

If $\varepsilon \in A$, the recursion can call itself without observational delay. The equation may still have least or greatest fixed points under an order, but unique metric semantics no longer follows from the same argument.

10.2 The bounded-input ultrametric

Let $\mathcal{B}$ be a space of recognition behaviors. Define the agreement depth of distinct behaviors f, g as the least input length on which they differ, and set

$$d(f, g) = 2^{-n(f, g)}.$$

As with languages, this is an ultrametric. A Cauchy sequence eventually stabilizes on every finite input domain. Its limit is obtained by taking the eventual value on each input, so the full ambient behavior space is complete when the behavior laws are closed under this pointwise stabilization.

An operator $F : \mathcal{B} \rightarrow \mathcal{B}$ is a contraction with factor $c < 1$ if

$$d(Ff, Fg) \leq c d(f, g).$$

Prefixing every recursive contribution by one terminal shifts the first possible disagreement one length deeper, giving factor $1/2$.

Theorem 10.2 (Contraction fixed point). *If $\mathcal{B}$ is complete and F is a contraction, then F has a unique fixed point. Starting from any x_0 , the iterates $x_{n+1} = F(x_n)$ converge to it.*

Proof walkthrough. The contraction inequality gives

$$d(x_{n+1}, x_n) \leq c^n d(x_1, x_0).$$

The ordinary triangle inequality then bounds the distance between x_m and x_n by a geometric tail, so the sequence is Cauchy. Completeness supplies a limit x^* . Continuity of a contraction yields $F(x^*) = x^*$. If y^* were another fixed point, then

$$d(x^*, y^*) = d(Fx^*, Fy^*) \leq c d(x^*, y^*),$$

forcing distance zero.

In an ultrametric recognition space the proof is sharper: each iterate gains a fixed amount of exact input depth. The convergence rate becomes an explicit finite reconstruction schedule. $\triangle$

10.3 Strong guarding and operational adequacy

Metric uniqueness alone does not prove that a parser's operational evaluator computes the fixed point. A complete account needs two results:

1. the semantic equation operator is contractive and has a unique fixed point; and
2. finite recursive evaluation agrees with sufficiently deep unfoldings on each bounded input.

The second is an adequacy theorem. It checks the operational details of failure, choice, and sequencing rather than assuming that denotational uniqueness automatically determines an evaluator.

In a strongly guarded grammar, every recursive call is reached only after a dedicated terminal prefix has consumed input. Predicates require care: a lookahead may inspect input without consuming it and can therefore destroy a naive guarding argument. A call hidden behind an expression that merely *sometimes* consumes is also not uniformly guarded.

Boundary or warning

“Every successful cycle consumes” is weaker than “the recursive operator delays every possible disagreement.” Guarding is a statement about the operator on arbitrary candidate behaviors, not only about the final successful runs.

10.4 Mutual recursion and delay graphs

For a vector of equations $(X_1, \dots, X_k)$, individual calls may have different delays. Record the minimum guaranteed consumption along each dependency edge in a matrix or weighted graph. A zero-delay edge is not fatal when every directed cycle has positive total delay.

If the zero-delay subgraph is acyclic, some finite power F^r of the system operator moves every dependency through positive delay. The power is a contraction even when F itself is only nonexpansive.

Proposition 10.3. *If F^r is a contraction on a complete metric space, then F has a unique fixed point.*

Guided proof. The contraction F^r has a unique fixed point x . Since

$$F^r(Fx) = F(F^r x) = Fx,$$

the point Fx is also fixed by F^r . Uniqueness gives $Fx = x$, so x is fixed by F . Any fixed point of F is fixed by F^r , hence equals x . $\square$

The proof separates a local syntactic certificate—positive delay on every cycle—from a global semantic consequence—unique exact behavior and bounded reconstruction.

10.5 Least fixed points are a different doctrine

Positive relational grammars use monotone set-valued equations. Even nullable or left-recursive cycles have a least solution obtained as the union of finite derivation stages. This is not the same as PEG-style unique guarded evaluation. The equation $X = X \cup B$ has many fixed points but a clear least one, namely B .

FPRD therefore records the fixed-point policy as part of the semantic doctrine. One cannot classify a relational nullable cycle as a failed deterministic PEG merely because it is unguarded under the metric policy.

Exercise 10.4. Prove by induction that the n th unfolding of $X = AX \cup B$ from $\emptyset$ is $\bigcup_{j < n} A^j B$. Under the assumption $\varepsilon \notin A$, give an input-length bound after which the unfolding is exact through length N . See the [solution](#).

Exercise 10.5. Construct a two-variable operator whose first power is not contractive because one dependency has zero delay, but whose square is contractive because every cycle consumes a terminal. See the [solution](#).

From finite presentations to generated dynamics

A finite grammar does more than denote a language. Its labeled recursive occurrences specify how continuation contexts are assembled. This chapter develops that middle object: the dynamic generated by a presentation after its exact semantics has been fixed.

11.1 Call sites and right-success frames

Consider a call to nonterminal A inside a sequence

$$e_1 A e_2 e_3.$$

If control reaches the call and it succeeds, the expressions to its right remain to be performed. Their exact behavior forms a continuation frame. A labeled call occurrence ℓ therefore carries an action

$$\Phi_\ell(K) = K \circ R_\ell$$

or the parser-order equivalent, where R_ℓ is the solved behavior of the right-success context.

Calls in different syntactic locations may invoke the same nonterminal but have different right contexts. Unique occurrence labels prevent those actions from being merged accidentally.

Example 11.1 (Two calls, one callee). In alternatives Ab and Ac , the two occurrences of A share its semantic behavior but generate different continuation actions: one appends the obligation b , the other c . Language-level normalization may combine or split the alternatives; the labeled origin relation explains what happened to the actions.

Only right-success material belongs to this particular frame extraction. Prefixes before the call determine whether the call is reached but do not run after it returns. Predicates and prioritized failure control may require richer frames and must not be smuggled into the positive construction.

11.2 Graph-directed dynamics

With several nonterminals or control locations, states naturally carry a location and a continuation behavior. Labeled call actions form directed edges among locations. Starting from designated states and closing under the finite action alphabet produces a reachable graph-directed orbit.

At every finite observation depth the reachable orbit is finite because the behavior tables and location set are finite. Restriction maps commute with the actions. The inverse limit of these finite reachable shadows reconstructs the compact closure of the exact graph-directed orbit.

Uniform readouts must include whatever the presentation claims to observe: location, remaining suffix behavior, complete match, or labeled call effect. Without readout, the inverse-limit state space is only a topological encoding.

Example 11.2 (A free context family). Suppose a presentation has r call sites whose frames prepend distinct markers to a continuation. Composing frames along a word in the call labels produces a free orbit when no two label words have the same exact behavior. The orbit grows like $1 + r + \dots + r^N$ through action depth N , while its compact closure can add a common boundary reached by contexts whose pending markers escape every bounded input observation.

11.3 Transformations and provenance

A presentation transformation may preserve one layer and change another. Alpha-renaming should transport labels functorially. Reassociating a sequence should preserve the ordered right-success frame. Distributing positive terms into grammar productions can split a source call among several normalized occurrences. Eliminating an impossible branch can remove a call entirely.

The appropriate invariant is often a finite origin relation

$$\text{Origin} \subseteq \{\text{target calls}\} \times \{\text{source calls}\}.$$

A target call may have one source, several source occurrences may merge, or one source occurrence may be cloned. Exact language or suffix equality can coexist with this non-bijective action refinement.

Boundary or warning

A normal-form theorem for languages is not automatically a conjugacy theorem for presentation dynamics. It must state how named actions transform.

11.4 Context-free grammars as positive equations

A conventional context-free grammar translates into finite positive set-valued equations. A production

$$A \rightarrow X_1 \cdots X_k$$

becomes relational sequence of the terminal and nonterminal behaviors. Union over productions becomes commutative union. The least fixed point collects exactly the suffixes supported by finite derivation trees.

Complete-match readout recovers the ordinary context-free language. The suffix semantics is richer: on input uv , a derivation of terminal prefix u returns suffix v . Occurrence labels retain continuation actions through the translation.

Conversely, finite positive relational terms distribute into finite sets of terminal/nonterminal words, so under complete-match observation they have ordinary context-free expressivity. The extensional equivalence does not erase the action-refinement issue described above.

11.5 What a complete presentation theorem must provide

For the purposes of this program, a finite-presentation-to-dynamics theorem is complete only when it supplies:

1. exact semantics under an explicit fixed-point policy;
2. a finite schema of named generating actions;
3. a defined reachable exact orbit or graph-directed state space;
4. finite observational shadows and compatible action maps;
5. inverse-limit or other exact reconstruction;
6. uniform recognition readout; and
7. transformation/provenance rules for the presentation operations being claimed.

This list is demanding by design. Omitting any item can turn a representation claim into a state encoding, a language theorem, or a syntactic graph without the promised recognition dynamic.

Research pause

Presentation dependence can be treated as nuisance, signal, or both. Which presentation features should survive into a canonical dynamic? Labeled calls may reflect meaningful control structure, but arbitrary refactorings may also create meaningless distinctions. What transformation class should define the right notion of equivalence?

Exercise 11.3. For the expression $(AbBc)/d$, list the right-success context of each call occurrence in the first alternative. Explain why the prefix before a call is not part of its return frame. See the [solution](#).

Exercise 11.4. Give a positive grammar rewrite that duplicates one occurrence of a nonterminal call while preserving the recognized language. Describe a one-to-many origin relation for the call labels. See the [solution](#).

Part IV

Context-free geometry and controlled stacks

Context-free languages and Dyck geometry

Context-free languages are the first broad recognition family whose natural state is unbounded but highly structured. Dyck languages isolate the recursive geometry: properly nested opening and closing symbols. Finite control can then select and relabel the balanced traces needed for a particular grammar.

12.1 A brief return to context-free grammars

A context-free grammar is a tuple (V, Σ, P, S) of nonterminals, terminals, productions, and start nonterminal. Each production replaces one nonterminal by a word over $V \cup \Sigma$. The language consists of terminal words derivable from S .

The grammar

$$S \rightarrow aSb \mid \varepsilon$$

generates $\{a^n b^n : n \geq 0\}$. Its finite syntax produces an unbounded matching obligation: after reading n copies of a , the recognizer must remember that n copies of b remain.

Ambiguity concerns derivations, not language membership. Two parse trees with the same terminal yield contribute one accepted word under complete-match observation. A suffix-relation semantics also forgets multiplicity unless it is explicitly weighted or tree-valued.

Example 12.1 (Same language, different derivation structure). The grammar $S \rightarrow SS \mid a$ generates the nonempty unary language and is highly ambiguous. The grammar $S \rightarrow aS \mid a$ generates the same language unambiguously. Their complete-match observations agree; their presentation structures and call actions do not.

12.2 Dyck words

Fix $k \geq 1$ bracket types with opening symbols $o_1, \dots, o_k$ and matching closers $c_1, \dots, c_k$. Optional neutral symbols may be admitted without changing the stack.

Definition 12.2 (Generalized Dyck language). The Dyck language D_k is the least language containing ε and closed under concatenation and typed enclosure:

$$x, y \in D_k \Rightarrow xy \in D_k, \quad x \in D_k \Rightarrow o_i x c_i \in D_k.$$

Equivalently, scan left to right with a stack: push type i on o_i , require and pop type i on c_i , reject underflow or mismatch, and accept exactly when the final stack is empty.

For $k = 1$, type information disappears and the stack is a nonnegative height. For $k \geq 2$, the order of pending types matters. The prefixes o_1o_2 and o_2o_1 have the same height but require different completing suffixes.

Example 12.3 (Prefix classification). In D_2 , the prefix $o_1o_2c_2$ leaves stack 1 and is completable by c_1 . The prefix $o_1o_2c_1$ has a type mismatch and can never be repaired by a future. The prefix c_1 underflows immediately and is likewise irreparable.

12.3 Why Dyck structure is universal for context-free syntax

The Chomsky–Schützenberger representation theorem says, in one standard form, that every context-free language is a homomorphic image of the intersection of a Dyck language with a regular language [9]. Schematically,

$$L = h(D_k \cap R).$$

The Dyck component carries recursive matching. The regular language R carries finite grammar-specific control. The homomorphism h reads the desired terminal output. Variants differ in erasing behavior, bracket alphabet, and exact normal form; those hypotheses matter in constructive use.

Proof walkthrough. Put a context-free grammar into a form where each production choice can be recorded locally. Encode a derivation by matched brackets marking entry to and return from nonterminal expansions. Proper nesting of recursive calls is exactly a Dyck condition. A finite automaton checks that neighboring bracket labels correspond to legal production choices and terminal positions. A homomorphism forgets the derivation annotations and emits the terminal yield.

The theorem is not saying that context-free languages are regular after adding parentheses. It separates an unbounded universal nesting mechanism from finite language-specific control. $\triangle$

This separation is attractive for recognition dynamics. Instead of building a new infinite stack space for every grammar, study one canonical Dyck residual dynamic and vary a finite controller.

Provenance

The Chomsky–Schützenberger theorem and visibly pushdown formulations are classical. Visibly pushdown languages make stack action visible in the input alphabet [1]. FPRD uses these ideas as calibration and as ingredients in a readout-aware suffix-recognition construction.

12.4 A positive recognition equation

The one-type Dyck language satisfies the least equation

$$D = \{\varepsilon\} \cup oDcD.$$

The first recursive occurrence handles the content of one matched pair; the second handles concatenation with the remaining balanced word. Every nonconstant recursive branch begins with the terminal o , so the equation is positively guarded under bounded-input observation.

For k types,

$$D = \{\varepsilon\} \cup \bigcup_{i=1}^k o_i D c_i D.$$

The least fixed point is the ordinary Dyck language. Guarded unfolding recovers all words through length N after a bounded number of stages.

Suffix semantics enriches this equation. On input wv , every balanced prefix $w \in D_k$ yields suffix v . Thus the exact relation records all balanced prefixes, while complete-match observation selects $v = \varepsilon$.

12.5 Depth and typed stack as geometric coordinates

During a valid prefix scan, the stack word

$$s = i_1 i_2 \cdots i_h \in \{1, \dots, k\}^*$$

records pending closing types from bottom to top. Stack height h is a coarse coordinate; the full word is exact. The set of live stack coordinates forms a rooted k -ary tree. Adding an irreparable failure point yields the complete residual state set derived in the next chapter.

At bounded future horizon N , stacks deeper than N cannot be emptied and therefore become observationally indistinguishable from failure under complete Dyck membership. Stacks of height at most N remain visible through their canonical matching closers. The finite shadow contains

$$1 + \sum_{h=0}^N k^h$$

classes: one failure/tail class plus every typed stack through depth N .

For $k = 1$, this is $N + 2$. For $k \geq 2$, it grows exponentially like k^N . The same nesting discipline supports very different observational growth depending on bracket rank.

Research pause

The Dyck frame moves grammar-specific complexity into finite control, but the required bracket rank and coding width may depend on the grammar. Should a universal representation allow the recursive alphabet to vary, or should it compile every grammar into one fixed envelope and measure the recoding cost? Both choices preserve expressivity; they create different quantitative theories.

Exercise 12.4. Classify every prefix of length at most three over one bracket type as live with a stack height or irreparable failure. For each live prefix give its shortest completing suffix. See the [solution](#).

Exercise 12.5. For two distinct typed stacks of height at most N , construct a future of length at most N that distinguishes their residuals. Handle the case where one stack is a proper prefix of the other. See the [solution](#).

The exact residual dynamic of Dyck recognition

The stack algorithm suggests a recognizer. Residual theory tells us whether its states are canonical and minimal. We will show that every live typed stack and one failure state are exactly the left residuals of the generalized Dyck language.

13.1 Residuals indexed by stacks

For a stack $s = i_1 \cdots i_h$, let R_s be the future language that completes that pending stack while otherwise remaining well nested. Let $\perp$ denote the empty future language reached after mismatch or underflow.

Scanning a valid prefix u from the empty stack yields a stack $s(u)$. If the scan fails, set $s(u) = \perp$. The stack algorithm gives

$$u^{-1}D_k = \begin{cases} R_{s(u)}, & s(u) \neq \perp, \\ \emptyset, & s(u) = \perp. \end{cases}$$

Theorem 13.1 (Dyck residual classification). *The residuals of D_k are precisely*

$$\{R_s : s \in \{1, \dots, k\}^*\} \cup \{\emptyset\}.$$

Distinct stacks give distinct residuals.

Proof. The stack scan proves that every prefix reaches one of the displayed states. Conversely, the opening word $o_{i_1} \cdots o_{i_h}$ reaches stack s , so every R_s is reachable. A mismatching closer reaches $\emptyset$.

To separate stacks $s \neq t$, let $\bar{s}$ be the unique sequence of matching closers that empties s from top to bottom. The word $\bar{s}$ belongs to R_s . If t differs in a topmost required type, it mismatches. If t is deeper than s but agrees on the popped suffix, it remains nonempty after $\bar{s}$ and therefore does not accept. If t is shallower, it underflows before the future ends. Thus $\bar{s} \notin R_t$. The failure residual is separated from R_s by $\bar{s}$. $\square$

By Myhill–Nerode, this stack-or-failure dynamic is the unique minimal deterministic recognizer, although it is infinite.

13.2 Letter actions

Openers push:

$$R_s \xrightarrow{o_i} R_{si}.$$

A matching closer pops:

$$R_{si} \xrightarrow{c_i} R_s.$$

A mismatching closer or closer at the empty stack enters $\emptyset$, which is absorbing. Neutral symbols, when present, act as the identity.

The readout accepts exactly R_ϵ , because only the empty stack contains the empty future. The action is therefore exact and uniformly readable.

For $k = 1$, push and pop generate the bicyclic pattern: pop is a left inverse of push on live states, but push is not a left inverse of pop at the boundary. For $k \geq 2$, the typed actions realize the corresponding polycyclic structure; mismatches expose the zero element. The polycyclic monoids were introduced as generalizations of the bicyclic monoid by Nivat and Perrot [15]; the stack action here is their concrete recognition geometry.

Example 13.2 (Noncommuting stack actions). For two types, pushing 1 then 2 reaches stack 12, while pushing 2 then 1 reaches 21. The completing future c_2c_1 accepts from the first and mismatches from the second. Equal height does not imply equal state.

13.3 Exact bounded shadows

At horizon N , every stack s with $|s| \leq N$ remains distinct because its canonical closer word has length at most N . Every deeper stack rejects all futures of length at most N , as does $\emptyset$. Therefore

$$Q_N(D_k) = \{[R_s]_N : |s| \leq N\} \cup \{\tau_N\},$$

where τ_N is the common tail/failure row. Here $Q_N(D_k)$ is the horizon- N future-row quotient introduced in Chapter 5, and $[R_s]_N$ denotes the row of R_s restricted to futures through length N .

The quotient count is

$$q_{D_k}(N) = 1 + \sum_{h=0}^N k^h.$$

For $k = 1$, $q(N) = N + 2$. For $k > 1$,

$$q(N) = 1 + \frac{k^{N+1} - 1}{k - 1}.$$

Restriction from $N + 1$ to N fixes stacks of height at most N and sends every height- $(N + 1)$ stack into the tail. This map is surjective.

Push has exposure zero on the exact stack coordinate but exposure one on bounded complete-match observations: a stack newly pushed to height $N + 1$ falls beyond the depth- N visible region. Pop makes a previously hidden top symbol observable and likewise uses a shifted level. The exact action and its finite shadows must be typed carefully rather than forced into one same-level map.

13.4 Inverse reconstruction

Every finite stack determines a compatible thread that is eventually represented by the same stack at all sufficiently deep levels. There are also infinite typed sequences: at level N , remember their prefix through height N in the orientation appropriate to the quotient. These threads form boundary points of the compact completion.

The boundary is much larger for $k \geq 2$ than for rank one. At rank one there is essentially one way for height to escape to infinity. At higher rank, an infinite typed stack records an element of $\{1, \dots, k\}^{\mathbb{N}}$, producing a Cantor-like boundary.

This difference is invisible if one records only height. The finite-shadow growth already detects it: linear for one type, exponential for several.

Intuition

The Dyck residual space is a rooted tree of finite stacks, one absorbing failure point, and a boundary of infinite stack directions. Push moves away from the root, matching pop moves toward it, and mismatch falls into failure. Finite observation truncates the tree and merges everything beyond the cut with failure because no short future can return from there.

13.5 Access growth

Every stack of height h is reached by h opening symbols. The tail/failure class is reached at length one by a mismatching closer. Hence

$$a_{D_k}(N) = \max(1, N).$$

The access coordinate is linear for every bracket rank even though the number of visible rows is linear at $k = 1$ and exponential for $k \geq 2$.

This is the first clean separation between how many observations exist and how far one must search to see them all.

Exercise 13.3. Derive the depth- N quotient count for $k = 1, 2, 3$ and compute the values through $N = 3$. Identify the extra class not represented by a visible live stack. See the [solution](#).

Exercise 13.4. Construct two different infinite typed-stack threads for $k = 2$ and show that a finite quotient eventually distinguishes them. Why is there only one escaping height direction for $k = 1$? See the [solution](#).

Finite control over a universal recursive frame

Bare Dyck recognition remembers only stack validity. A context-free language also needs finite control specifying which bracketed traces correspond to legal grammar choices and which output symbols they produce. Combining an infinite canonical stack dynamic with a finite controller yields a large family of finitely presented recognition dynamics.

14.1 The controller product

Let $C = (P, p_0, \delta_C, F_C)$ be a finite automaton over the bracket-and-neutral alphabet. A controlled Dyck language has the form

$$L_C = D_k \cap L(C)$$

or its image under a fixed output map. A direct recognizer carries a pair

$$(s, p) \in \{\text{typed stacks}\} \times P$$

plus failure. Each input symbol updates both the stack and finite control.

The full Cartesian product is only a presentation. Some pairs are unreachable; others may have identical controlled futures. The canonical object is the reachable residual quotient of the product.

Example 14.1 (Recovering $a^n b^n$). Use one bracket type. Map each opening symbol to output a and each closing symbol to output b . A finite controller forbids concatenation of several balanced components and permits only one mountain of opens followed by closes. The projected outputs are exactly $a^n b^n$.

14.2 Balanced-word saturation

A controller can make arbitrarily long detours while returning the stack to the same height. Computing reachable semantic pairs at bounded stack height therefore cannot simply enumerate input words up to a matching length bound. One must saturate the finite relation induced by balanced words.

For controller states p, q , record whether some balanced word takes p to q without violating the stack discipline. Closure under concatenation and matched enclosure computes a finite balanced relation. This relation captures unbounded flat detours inside a finite control summary.

Once the balanced relation is known, recursive future refinement at each bounded stack height computes the exact controlled observation quotient. The finite quotients form a compatible tower, and compact nested-fiber arguments reconstruct the reachable exact dynamic.

Boundary or warning

Bounding stack height does not bound word length. A controller may traverse a long balanced loop without increasing height. Any algorithm that replaces balanced saturation by a fixed word-length cutoff is only an experiment unless it proves a sufficient cutoff.

14.3 A depth-threshold family

Let $D_k^{\geq r}$ contain the balanced words whose stack height reaches at least r . A finite controller needs only remember whether the threshold has already been crossed and, before crossing, the current height below r . The language is nonregular because after crossing the threshold it still must match an unbounded typed stack.

At future horizon N , a stack of height $h < r$ can reveal the threshold status only if enough future length remains to climb from h to r and return. This creates delayed observation classes in addition to the ordinary visible Dyck stacks. The exact candidate quotient count is

$$q_{k,r}(N) = 1 + \sum_{h=0}^N k^h + \sum_{h=\max(0, 2r-N)}^{r-1} k^h.$$

The first sum is the bare Dyck stack contribution; the second records below-threshold stacks whose future can now distinguish whether the threshold has already been reached.

The access radius is

$$a_{k,r}(N) = \max(2r, N).$$

At horizon zero the two output classes—accepted and rejected—already exist, but a shortest accepted threshold witness has length $2r$. This family again separates observation count from access delay.

Provenance

The formulas in this section are current FPRD candidate results supported by direct proofs and independent finite enumeration. They are presented to teach the mechanism, not as classical facts or externally reviewed theorems. The unbounded proof obligations remain visible in Chapter 19.

14.4 The shallow complement

The complementary balanced language that never reaches height r is regular: the live stack belongs to the finite set of typed stacks of height at most $r - 1$, plus failure. Its eventual residual count is

$$1 + \sum_{h=0}^{r-1} k^h,$$

its accessibility degree is $\max(1, r - 1)$, and its distinguishability degree is $r - 1$.

Threshold and shallow languages partition the Dyck language but have very different dynamics. The shallow side turns the stack tree into a finite truncation; the threshold side retains the entire unbounded stack after a finite control event.

14.5 Fixed-envelope recoding

The bracket alphabet in a Chomsky–Schützenberger representation may depend on the grammar. To obtain one fixed recursive frame, encode each source bracket name as a fixed-width digit block over a chosen base. Each output position becomes an envelope bracket carrying its digit and any required output label.

A finite trie controller checks block alignment and rejects malformed partial blocks. The block map is injective, preserves and reflects matching, and commutes with output projection. If the block width is m , source and envelope observation depths scale:

$$\text{output depth } N \quad \text{consults source depth roughly } \lfloor N/m \rfloor.$$

Shortest distinguishing words and action macros scale by the same block geometry.

This construction buys a universal alphabet at a measurable cost: more envelope symbols, finite controller states, macro-action length, and observation lag. Expressive equivalence alone would hide these tradeoffs.

14.6 Toward an affine access bound

For a fixed finite controller, let B bound the length of shortest balanced witnesses needed for its saturated balanced relation. A candidate general bound has the form

$$a_{LC}(N) \leq \max\{1, (N + 1)B + N\}.$$

The idea is to realize a representative bounded future by alternating at most $N + 1$ finite balanced controller transfers with at most N unmatched stack steps. Finiteness of the controller makes B finite; the stack contributes the linear horizon term.

The bound is intentionally coarse. Special families such as bare Dyck and depth thresholds have much sharper exact access laws. Its value would be uniformization: fixed finite control cannot postpone a horizon- N observation beyond an affine function of N once balanced transfer witnesses are bounded.

Research pause

Which quantitative cost should classify a controlled recognition family: controller size, balanced witness bound, bracket rank, coding width, observational growth, access radius, or some joint profile? The examples show that none of these coordinates determines all the others.

Exercise 14.2. For one bracket type and threshold $r = 2$, list the shortest balanced words that do and do not cross the threshold. Explain why horizon-zero access is four. See the [solution](#).

Exercise 14.3. Construct the minimal DFA for balanced one-type words of height less than $r = 3$. Compute its accessibility and distinguishability degrees. See the [solution](#).

Part V

Algebra, information, and resource lower bounds

Hankel tables, linear algebra, and semirings

The past–future table $H_L(u, v) = \mathbf{1}[uv \in L]$ can be studied as a set of Boolean rows or as a matrix over an algebraic domain. These viewpoints answer different questions. This chapter reactivates the linear algebra needed to use rank correctly and explains why Boolean arithmetic changes the invariant.

15.1 Vector spaces and linear maps

A vector space over a field $\mathbb{F}$ supports addition, scalar multiplication, additive inverses, and division by nonzero scalars. A list $v_1, \dots, v_r$ is linearly independent when

$$\alpha_1 v_1 + \dots + \alpha_r v_r = 0$$

forces every $\alpha_i = 0$. A basis is an independent spanning list, and the dimension is its size.

A linear map preserves addition and scalar multiplication. Once bases are chosen, a finite-dimensional linear map becomes a matrix. The matrix is a coordinate presentation; the map is the underlying object.

The rank of a matrix is the dimension of its row space, equivalently its column space. It measures the number of independent linear directions, not the number of distinct rows.

Example 15.1 (Distinctness versus rank). Over $\mathbb{R}$, the rows

$$(1, 0), \quad (0, 1), \quad (1, 1), \quad (2, 1)$$

are four distinct vectors but span a two-dimensional space. This illustrates the distinction: a deterministic recognizer must keep distinct Boolean future rows separate, while a weighted linear representation may synthesize later rows as combinations of a basis.

15.2 Identity minors as certificates

If a matrix contains an $r \times r$ identity submatrix, its rank over every field is at least r . The selected rows are independent because restricting a linear relation to the selected columns reads off each coefficient.

For language tables, identity minors often arise from canonical separators. Choose prefixes $u_1, \dots, u_r$ and futures $v_1, \dots, v_r$ such that

$$u_i v_j \in L \iff i = j.$$

Then the corresponding Hankel submatrix is the identity.

Example 15.2 (Dyck identity minor). Select distinct typed stacks $s_1, \dots, s_r$ and let u_i be their opening words. Let v_j be the canonical closer word for s_j . Then $u_i v_j$ is balanced exactly when the typed stacks agree. Every finite collection of visible stacks therefore yields an identity minor.

An identity minor proves a lower bound without computing all determinants. It also gives an operational interpretation: each row owns a private future that accepts only from that state.

15.3 Factorization and state coordinates

Suppose a finite matrix M factors as

$$M = AB$$

with A having r columns and B having r rows. Then $\text{rank}(M) \leq r$. Conversely, a rank- r matrix over a field admits such a factorization by choosing a row or column basis.

In recognition terms, A can encode each prefix into an r -dimensional state and B can evaluate futures linearly. This is a linear readout-aware realization of the finite table. It need not be deterministic, Boolean, or closed under the exact letter actions unless those properties are proved separately.

Boundary or warning

Low field rank does not imply few deterministic states. Distinct Boolean rows can lie in a low-dimensional linear space. Conversely, a zero Boolean row is a deterministic failure state but contributes no new linear dimension.

15.4 Semirings and the Boolean domain

A semiring has addition and multiplication with identities but need not have additive inverses. The Boolean semiring is

$$\mathbb{B} = (\{0, 1\}, \vee, \wedge, 0, 1).$$

Here $1 + 1 = 1$. Matrix multiplication uses OR of ANDs.

A Boolean factorization $M = AB$ says that each 1-entry is covered by at least one shared Boolean feature. The minimum factor width is Boolean rank. Without subtraction, familiar field arguments can fail; row rank, column rank, and factor rank require careful definitions.

Example 15.3 (A Boolean cover). The matrix

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

has three distinct rows. Over the Boolean semiring the third row is the OR of the first two. A deterministic state count remains three because the futures differ; a Boolean feature factorization may use two generators.

Identity matrices still have Boolean rank equal to their size. Each diagonal 1 must be covered without creating an off-diagonal 1, so one Boolean feature cannot cover two different diagonal entries. The canonical Dyck identity minors therefore lower-bound both field rank and Boolean rank.

15.5 Finite Hankel windows

Choose bounded prefix and future sets, for example

$$U_I = \{u : |u| \leq I\}, \quad V_N = \{v : |v| \leq N\}.$$

The finite Hankel window is

$$H_L[I, N] = (\mathbf{1}[uv \in L])_{u \in U_I, v \in V_N}.$$

Its number of distinct rows is $G_L(I, N)$. Its field rank and Boolean rank may be smaller.

For bare Dyck recognition and prefix radius $I \geq \max(1, N)$, saved stacks factor the window: a prefix first maps to its visible stack/failure coordinate, and that coordinate determines every future entry. Canonical closer columns provide an identity minor on the live visible stacks. Hence, up to the zero failure row, the factorization upper bound and identity-minor lower bound meet exactly.

If $S_k(N) = \sum_{h=0}^N k^h$ counts live typed stacks, then a sufficiently wide Dyck window has

$$\text{rank}_{\mathbb{F}} = S_k(N)$$

over every field, while the number of deterministic rows is $S_k(N) + 1$ when failure is reachable. Deeper live stacks and irreparable failure merge into one all-zero bounded row. The one-row discrepancy is mathematically meaningful: linear dimension ignores a zero vector; deterministic recognition cannot ignore a reachable rejecting future.

15.6 Infinite rank and regularity

The full Hankel matrix of a regular language has finite field rank because it has finitely many distinct residual rows. The converse over a field belongs to the classical finite-rank realization theory for weighted and stochastic automata [8]; for Boolean characteristic series, finite field rank is closely tied to regular structure but the exact domain and representation hypotheses should be stated.

FPRD primarily uses finite windows and explicit minors. This avoids claiming that one global rank notion captures every deterministic or semiring resource. The two-parameter profile records how algebraic complexity appears as both past and future observation increase.

Exercise 15.4. Construct five distinct Boolean rows in $\mathbb{R}^3$ whose field rank is three. Explain why a deterministic row recognizer still requires five row names on that finite table. See the [solution](#).

Exercise 15.5. Prove that the Boolean rank of the $r \times r$ identity matrix is r . Phrase the proof as a rectangle-cover argument. See the [solution](#).

Exercise 15.6. Write a Hankel window for one-type Dyck prefixes of height at most two and the three canonical closer futures. Identify an identity minor and the zero failure row. See the [solution](#).

From observation tables to information and memory

A row of a past–future table is not only a semantic residual. At a fixed cut it is the exact information that must pass from a prefix processor to a future processor. The same row count therefore controls layered recognizers, one-way communication, and streaming memory.

16.1 Strict layered recognition

Fix a total input length $m + n$. A strict layered recognizer reads exactly m symbols, crosses a cut, then reads exactly n more. At the cut, two prefixes $u, u' \in \Sigma^m$ may share a state only if every suffix $v \in \Sigma^n$ gives the same outcome:

$$\forall v \in \Sigma^n, \quad uv \in L \iff u'v \in L.$$

Thus cut states are exactly the distinct rows of the fixed matrix

$$C_{m,n}(u, v) = \mathbf{1}[uv \in L].$$

Theorem 16.1 (Fixed-cut state minimum). *The minimum number of reachable cut states in a deterministic strict layered recognizer is the number of distinct rows of $C_{m,n}$.*

Proof. For the lower bound, prefixes with different rows have some separating suffix and cannot share a deterministic cut state. For the upper bound, use the row itself as the cut state. On receiving suffix v , read the corresponding row coordinate. This recognizes the finite fixed-length problem exactly. $\square$

This is finite-table minimization. It says nothing by itself about one time-homogeneous automaton working at every length.

16.2 One-way deterministic communication

Imagine that Alice receives $u \in \Sigma^m$ and Bob receives $v \in \Sigma^n$. Alice sends one deterministic message; Bob must decide whether $uv \in L$.

If two prefixes have different rows, Bob can choose a suffix on which their answers differ. Alice must send different messages. Conversely, sending the row class is sufficient. Therefore the exact minimum number of messages is the same distinct-row count.

If there are r rows, a fixed-length binary encoding needs

$$\lceil \log_2 r \rceil$$

bits. The count r is the invariant; the bit ceiling is an encoding cost.

Provenance

The connection between communication matrices and deterministic one-way protocols is classical communication complexity; Yao's foundational work is one standard entry point [21]. The recognition-specific transfer identifies the same exact row quotient with layered machine width.

16.3 Streaming lower bounds

An exact one-pass streaming recognizer has some internal configuration after reading the first m symbols. If prefixes with different future rows reached the same configuration, feeding a separating suffix would force identical future computation from the same configuration, contradicting exactness.

Hence the number of possible cut configurations is at least the fixed-cut row count. A machine with s bits has at most 2^s configurations, so

$$s \geq \lceil \log_2 r \rceil.$$

This lower bound is model-sensitive. It assumes deterministic exact recognition and counts all information present at the cut. Randomization, error, multiple passes, advice, or nonuniformity changes the communication problem and must be analyzed separately.

Example 16.2 (Dyck memory). At a midpoint cut, different visible typed stacks have canonical closer suffixes. There are roughly k^h stacks of height h , so distinguishing them requires $\Theta(h \log k)$ bits. A direct stack recognizer stores h types using $\lceil \log_2 k \rceil$ bits each, matching the scale. For one bracket type, only height matters and $\Theta(\log h)$ bits suffice.

The gap between one type and many types is not merely an implementation detail. It is already present in the exact future rows.

16.4 Fixed cuts can be simultaneously exact and globally misleading

For the square-length language, every word in an (m, n) cut has total length $m + n$. The cut matrix is all ones when that total is square and all zeros otherwise. It has one row, needs one cut state, and requires zero message bits.

Yet the uniform residual dynamic is infinite. The fixed-cut protocol is allowed to know m and n as part of the problem instance; it never needs to use one state space across varying prefix lengths. A streaming algorithm for a known fixed total length can likewise bake the answer into nonuniform control.

Boundary or warning

Exact complexity at every separate length does not automatically assemble into a uniform recognizer. The missing information is compatibility across cuts: which literal row at one prefix length is the continuation of which row at the next. Chapters 17 and 18 restore that information.

16.5 Three invariants, three resource questions

For a finite Boolean table, distinguish:

1. the number of distinct rows, controlling deterministic states and deterministic one-way messages;
2. field rank, controlling linear factorization dimension over a chosen field; and
3. Boolean rank, controlling OR-of-rectangles feature factorization.

None generally determines the others exactly. A useful complexity bridge names the machine or communication model and proves the transfer, rather than calling all three quantities “Hankel complexity.”

16.6 Upper bounds require readout

A lower bound follows from separators. A matching upper bound requires a recognizer that stores the proposed coordinate, updates it, and reads the answer. For Dyck languages, a typed stack provides all three. For a controlled language, one needs the stack plus finite controller and must minimize or bound their reachable joint futures.

This is the same readout discipline encountered in Chapter 9. An information count becomes a complete representation result only when a uniform operational realization meets it.

Exercise 16.3. For equality on two-bit strings, where Alice and Bob must decide whether $u = v$, write the communication matrix, count its rows, and derive the exact deterministic one-way message count. See the [solution](#).

Exercise 16.4. Prove that every fixed cut of the square-length language has one row. Then use the residual separator from Chapter 2 to explain why no uniform finite DFA follows. See the [solution](#).

Exercise 16.5. Turn an arbitrary pair of distinct cut rows into a streaming-state lower-bound argument. State exactly where determinism and exactness are used. See the [solution](#).

Part VI

From local observations to global dynamics

Sewing local observations into a global language

Fixed cuts give exact finite problems, but a family of unrelated matrices need not describe one language. The missing ingredient is literal compatibility: the same symbol moved across a cut must denote the same concatenated word. Once labels and compatibility are retained, local tables sew into a unique global language and their joint rectangles recover the residual orbit and its completion.

17.1 A two-parameter field of cuts

For each $m, n \geq 0$, let

$$C_{m,n} : \Sigma^m \times \Sigma^n \rightarrow \{0, 1\}$$

be a Boolean matrix with literal word labels. A language L produces

$$C_{m,n}^L(u, v) = \mathbf{1}[uv \in L].$$

The same word has several cut decompositions. If $a \in \Sigma$, then $u(av)$ and $(ua)v$ are literally the same concatenation. Therefore every language field satisfies the diagonal transport law

$$C_{m,n+1}(u, av) = C_{m+1,n}(ua, v). \quad (17.1)$$

Definition 17.1 (Compatible cut field). A family $(C_{m,n})_{m,n \geq 0}$ is compatible when it satisfies (17.1) for every legal m, n, u, v, a .

Anonymous row counts cannot express this law. Even matrices known only up to independent row permutation lose the identity of the prefix being transported.

17.2 The sewing theorem

Theorem 17.2 (Unique language from compatible cuts). *Every compatible Boolean cut field determines a unique language*

$$L_C = \{w : C_{|w|,0}(w, \varepsilon) = 1\}$$

such that

$$C_{m,n}(u, v) = \mathbf{1}[uv \in L_C]$$

for every cut. Conversely, every language produces a compatible field.

Proof. The converse follows from associativity of word concatenation:

$$u(av) = (ua)v.$$

Now start with a compatible field and fix $u \in \Sigma^m$ and $v = a_1 \cdots a_n \in \Sigma^n$. Apply the transport law repeatedly:

$$\begin{aligned} C_{m,n}(u, a_1 \cdots a_n) &= C_{m+1,n-1}(ua_1, a_2 \cdots a_n) \\ &= \cdots \\ &= C_{m+n,0}(uv, \varepsilon). \end{aligned}$$

By definition, the last value is 1 exactly when $uv \in L_C$. Thus every field entry agrees with the language reconstructed on the zero-future axis.

Uniqueness follows because any realizing language L must satisfy

$$\mathbf{1}[w \in L] = C_{|w|,0}(w, \varepsilon),$$

so it equals L_C word by word. □

The theorem is elementary once the right data are retained. Its force lies in the boundary it draws: local cut widths do not assemble, but literal cut tables with the transport law do.

Example 17.3 (An incompatible family). Over $\{a\}$, set $C_{0,1}(\varepsilon, a) = 1$ but $C_{1,0}(a, \varepsilon) = 0$. Both one-entry matrices are valid Boolean tables in isolation. No language realizes them because they assign two values to the same word a across adjacent cuts.

17.3 Joint at-most rectangles

A fixed cut sees futures of one exact length. To preserve how those futures belong to one prefix, join all blocks through horizon N :

$$B_N(u) = (C_{|u|,|v|}(u, v))_{|v| \leq N}.$$

For prefix radius I , define

$$X[I, N] = \{B_N(u) : |u| \leq I\}.$$

Future restriction gives surjections

$$X[I, N+1] \twoheadrightarrow X[I, N].$$

Increasing I includes more literal prefix rows. Letter extension has the shifted type

$$X[I, N+1] \longrightarrow X[I+1, N], \quad B_{N+1}(u) \longmapsto B_N(ua).$$

Well-definedness is exactly the one-step future argument used in Chapter 5.

The rectangle has two independent meanings:

- I measures how much of the reachable past orbit has been explored;
- N measures how finely each reached state is observed.

Collapsing either axis too early hides a different phenomenon.

17.4 Two assembly orders

Fix a prefix radius I and take the inverse limit over N . A compatible thread belonging to one of the finitely many literal prefixes through I recovers that prefix's complete residual. Increasing I then unions these finite orbit fragments. This assembly order gives the literal residual orbit

$$O_L = \{u^{-1}L : u \in \Sigma^*\}.$$

Alternatively, fix N and let I grow until every horizon- N row has appeared, obtaining the bounded future-row quotient $Q_N(L)$. Taking the inverse limit over N gives the compact observational completion

$$X_L \cong \varprojlim_N Q_N(L) \cong \overline{O_L} \subseteq 2^{\Sigma^*}.$$

The comparison

$$O_L \hookrightarrow X_L$$

is injective and dense. It is onto exactly when every compatible observation thread is a literal residual, equivalently when the orbit is closed in its observational completion.

Boundary or warning

The two assembly orders are not being declared equal. One produces the literal orbit; the other may add boundary behaviors. Their comparison is the mathematical object of interest.

17.5 Naturality and reconstructed action

The shifted letter maps commute with future restriction. Therefore they act on compatible inverse threads. Under the identification with languages, this action is ordinary residual differentiation:

$$R \mapsto a^{-1}R.$$

This is an example of exposure one. The depth- N output after reading a letter depends on the depth- $(N+1)$ input. More generally, reading a word of length r has exposure r .

The finite rectangles thus retain enough information to reconstruct not only the state set but the canonical letter actions and epsilon readout. That is why the assembly is a recognition dynamic rather than a bare inverse-limit space.

17.6 What sewing does not prove

Every total compatible Boolean field comes from a language, so total-field sewing has no nontrivial existence obstruction beyond the transport equations. Interesting obstruction theory would require partial observations, noisy or inconsistent local data, restricted realizability classes, or additional presentation constraints.

Nor does sewing prove regularity. Every language, regular or not, has a compatible field. Regularity appears when the future tower stops refining at a genuine global plateau.

Research pause

Where should a nontrivial local-to-global obstruction live? Possibilities include partially specified cut tables, fields required to arise from one finite presentation doctrine, approximate observations with error, or local controllers that cannot be jointly realized. A useful obstruction must rule out something that the literal transport law alone permits.

Chapter 19 returns to the last possibility in a restricted controlled-Dyck setting. There the language and every literal observation are already globally compatible. The open issue is whether the cheapest semantic entry observations admit one shared finite regular presentation. The resulting entry-atlas sewing defect separates ordinary language sewing from finite-presentation sewing; its vanishing is semidecidable, while no positive example is currently known.

Exercise 17.4. For $v = abc$, write every step transporting $C_{m,3}(u, abc)$ to $C_{m+3,0}(uabc, \varepsilon)$. Identify exactly where literal labels are used. See the [solution](#).

Exercise 17.5. Specify three individually consistent adjacent cut entries over $\{a, b\}$ that become inconsistent when their transport equations are combined. See the [solution](#).

Global plateaus and finite-state collapse

The quotient count $q_L(N)$ is nondecreasing. If it ever has a genuine adjacent plateau after ranging over every prefix, the graded observation tower collapses to one time-homogeneous finite automaton. The proof is short enough to see completely and strong enough to explain why finite sampled plateaus are not certificates.

18.1 From equal cardinality to equal equivalence

The restriction bond

$$\rho : Q_{N+1}(L) \twoheadrightarrow Q_N(L)$$

is always surjective. If

$$q_L(N) = q_L(N+1), \tag{18.1}$$

then a surjection between finite sets of equal size is bijective.

The horizon- $(N+1)$ relation refines the horizon- N relation. Bijectivity means that no horizon- N class was split. Therefore

$$u \equiv_N u' \implies u \equiv_{N+1} u'. \tag{18.2}$$

The reverse implication always holds, so the two equivalence relations are equal.

18.2 The plateau theorem

Theorem 18.1 (Global adjacent plateau). *If $q_L(N) = q_L(N+1)$ for some N , then L is regular and $Q_N(L)$ is the state set of its minimal residual DFA.*

Proof. Suppose $u \equiv_N u'$. By (18.2), the prefixes also agree through horizon $N+1$. For any letter a and any v with $|v| \leq N$, the future av has length at most $N+1$, so

$$uav \in L \iff u'av \in L.$$

Hence

$$ua \equiv_N u'a.$$

Thus $\equiv_N$ is a right congruence.

Iterating the right-congruence property gives $uw \equiv_N u'w$ for every word w . Horizon- N agreement includes the empty future, so

$$uw \in L \iff u'w \in L$$

for every w . Therefore u and u' have the same complete residual. The finite relation $\equiv_N$ is exactly residual equivalence.

Letter actions $[u]_N \mapsto [ua]_N$ are now well defined at the same level. Take start state $[\varepsilon]_N$ and accept $[u]_N$ when $u \in L$. The resulting finite DFA recognizes L . Since its classes are exact residuals, the Myhill–Nerode theorem identifies it as minimal. $\square$

This theorem is a local numerical signal with a global quantifier hidden inside $q_L(N)$: every prefix in Σ^* has already been considered.

18.3 Why finite tables can lie by omission

In practice one may compute only

$$G_L(I, N) = |\{B_N(u) : |u| \leq I\}|.$$

Equal sampled counts

$$G_L(I, N) = G_L(I, N + 1)$$

do not show that longer prefixes introduce no new rows. A language can mimic a finite automaton on an enormous finite rectangle and diverge later.

Even a table extending through a familiar bound such as twice the current row count is a certificate only when one already knows an upper bound on the true number of residuals or has an equivalence oracle. Otherwise the table supports a hypothesis that can be refuted by more data, not a proof of regularity.

Example 18.2 (A delayed singleton). At small prefix radius $I < T$, the language $L_T = \{a^T\}$ looks empty at horizon zero: every sampled prefix rejects. The accepting row appears only at radius T . Any finite search limit can be defeated by choosing a later singleton.

18.4 Fixed cuts do not see the plateau

For the square-length language

$$L_{\square} = \{w : |w| \text{ is a perfect square}\},$$

each fixed (m, n) cut has one row. If $m + n$ is square, the matrix is all ones; otherwise it is all zeros. Width is permanently flat.

The uniform residual orbit is nevertheless infinite. If $p < q$, choose a square $s = t^2 \geq p$ whose next gap $2t + 1$ exceeds $q - p$. Append a word of length $s - p$. The prefix of length p reaches total length s , while the prefix of length q reaches

$$s + (q - p),$$

strictly between s and $(t + 1)^2$. The two residuals differ.

Fixed-cut row counts erase how prefixes at different lengths correspond. The joint horizon quotient restores that information, and its count cannot plateau for the square language.

18.5 Closed orbit and finite orbit

If L is regular, its residual orbit is finite and therefore closed in the Hausdorff behavior space. Its completion adds nothing.

For the observation tower considered here, a global adjacent plateau makes the orbit finite by the theorem. Conversely, if the orbit has k residuals, some finite horizon distinguishes every pair: choose one shortest separating word for each pair and take the maximum length. At that horizon $q_L(N) = k$ and the count remains k forever. Thus regularity, eventual quotient stabilization, and finite residual orbit are equivalent.

The phrase “closed implies finite” is not true for arbitrary subsets of a compact space. The theorem depends on the residual action and finite alphabet, not topology alone.

Provenance

The plateau mechanism is a form of classical finite-state partition refinement and Myhill–Nerode stabilization [12]. The FPRD sewing framework makes explicit how the same-level action emerges from a graded future-observation tower.

Exercise 18.3. Fill in the exact quantifiers proving that $u \equiv_N u'$ implies $ua \equiv_N u'a$ under the adjacent plateau. Why is horizon $N + 1$, rather than only N , necessary? See the [solution](#).

Exercise 18.4. Let a minimal DFA have k states. Prove that there is a finite horizon distinguishing every state pair and hence an eventual constant value $q_L(N) = k$. See the [solution](#).

Observation growth and access

The two-parameter rectangle separates how many future distinctions exist from how far into the prefix orbit one must travel to witness them. These coordinates connect finite automaton degrees, symbolic factor appearance, controlled-stack geometry, and uniformization cost.

19.1 Three related functions

Recall

$$\begin{aligned} G_L(I, N) &= |\{B_N(u) : |u| \leq I\}|, \\ q_L(N) &= |\{B_N(u) : u \in \Sigma^*\}|, \\ a_L(N) &= \min\{I : G_L(I, N) = q_L(N)\}. \end{aligned}$$

where $B_N(u)$ is the membership row of prefix u on every future of length at most N .

The function G_L is the finite rectangle. The function q_L is eventual growth on the prefix axis. The function a_L records where eventuality is first achieved at a fixed future horizon.

Proposition 19.1 (Basic access laws). *For every language over a finite alphabet:*

1. $a_L(N)$ exists;
2. $a_L(N)$ is nondecreasing in N ; and
3. if L has a reachable k -state DFA, then $a_L(N) \leq k - 1$.

Proof. Existence follows because the finite set $Q_N(L)$ has finitely many classes, each with a finite prefix representative. Take the maximum length among shortest representatives.

If prefixes through radius I represent every horizon- $(N + 1)$ class, their restrictions represent every horizon- N class. Hence the least sufficient radius cannot decrease as the horizon grows.

For a reachable k -state DFA, every state has a shortest access word of length at most $k - 1$. Every bounded row is the projection of a residual state, so these access words cover $Q_N(L)$ at every horizon. $\square$

19.2 Prefix-volume lower bounds

If $d = |\Sigma|$, the number of prefixes through radius I is

$$V_d(I) = \sum_{j=0}^I d^j.$$

At most one bounded row can be contributed by each prefix, so

$$q_L(N) \leq V_d(a_L(N)).$$

For $d = 1$,

$$a_L(N) \geq q_L(N) - 1.$$

For $d \geq 2$, inversion gives

$$a_L(N) \geq \lceil \log_d((d-1)q_L(N) + 1) \rceil - 1.$$

The alphabet size is the number of actual input symbols, not the bracket rank alone when neutral symbols or separate open/close symbols are present.

19.3 A gap in observation growth

Monotonicity and the global plateau theorem leave no room for sublinear unbounded quotient growth.

Theorem 19.2 (Observation-growth gap). *If L is nonregular, then $q_L(N)$ is strictly increasing and*

$$q_L(N) \geq N + 2$$

for every $N \geq 0$. Equivalently, if q_L ever fails to increase at an adjacent pair of horizons, then it remains bounded.

Proof. The restriction bonds make q_L nondecreasing. If $q_L(N) = q_L(N+1)$ at any horizon, Theorem 18.1 makes L regular, so its quotient growth is bounded. Contrapositively, a nonregular language must increase strictly at every step.

Moreover $q_L(0) = 2$ for a nonregular language. If it were one, every prefix would have the same acceptance value, making L either $\emptyset$ or Σ^* , both regular. A strictly increasing integer sequence beginning at two satisfies $q_L(N) \geq N + 2$. $\square$

The rank-one Dyck language is sharp: $q_{D_1}(N) = N + 2$. For unary languages, the bound also follows from the classical Morse–Hedlund complexity gap after the unary factor bridge [13]. Here the plateau theorem extends it to residual observations over every finite alphabet. Combining the gap with prefix volume gives universal access lower bounds for nonregular languages: $a_L(N) \geq N + 1$ in the unary case and at least logarithmic growth in N when $|\Sigma| \geq 2$.

19.4 A regularity criterion

Theorem 19.3 (Bounded access characterizes regularity). *For a language L , the following are equivalent:*

$$L \text{ is regular} \iff \sup_N a_L(N) < \infty \iff a_L(N) \text{ is eventually constant.}$$

Proof. If L is regular, the finite-DFA access bound gives a uniform bound on $a_L(N)$. Since $a_L(N)$ is a nondecreasing sequence of integers, boundedness is equivalent to eventual constancy.

Conversely, suppose $a_L(N) \leq A$ for all N . Prefix volume gives

$$q_L(N) \leq V_d(A)$$

uniformly. The nondecreasing integer sequence $q_L(N)$ must therefore have an adjacent plateau. The global plateau theorem makes L regular. $\square$

For a regular language, once the horizon distinguishes all residual states, $a_L(N)$ equals the accessibility degree of the minimal DFA. The first horizon where $q_L(N)$ reaches the residual count is its distinguishability degree.

Boundary or warning

An access plateau at one pair of horizons does not imply regularity. The proof uses a uniform bound over all horizons. Bare Dyck recognition can have $a(0) = a(1) = 1$ and still grow unbounded later.

19.5 From a finite controller to exact access

We now obtain the first general restriction imposed by one of our finite presentation doctrines. Fix a generalized-Dyck alphabet and a total finite controller

$$M = (Q, \Omega, \delta, q_0, F), \quad L_M = D \cap L(M).$$

Chapter 14 computed which stack/controller coordinates are globally reachable and which coordinates have the same bounded future. It did not ask how long the earliest prefix reaching each bounded row must be.

19.5.1 Balanced distance is a min-plus object

For controller states p, q , define

$$d_M(p, q) = \min\{|v| : v \in D, \delta(p, v) = q\},$$

with value ∞ if no balanced word realizes the pair. Forgetting the length recovers the balanced controller relation from Chapter 14. Retaining the length gives more.

The distance table is closed under four operations:

$$\begin{aligned}
d_M(p, p) &= 0, \\
d_M(p, \delta(p, z)) &\leq 1 && \text{for a neutral letter } z, \\
d_M(p, r) &\leq d_M(p, q) + d_M(q, r), \\
d_M(p, \delta(t, c_i)) &\leq d_M(\delta(p, o_i), t) + 2.
\end{aligned}$$

The third line concatenates balanced words. The fourth wraps an already balanced word between a matched opener and closer. Repeatedly relaxing these inequalities is a finite min-plus calculation: alternative witnesses take a minimum, while concatenated lengths add.

There is a small order-theoretic trap here. In the ordinary numerical order, d_M is the *greatest* nonnegative matrix satisfying these upper-bound inequalities: the all-zero matrix satisfies them too. Operationally we start at infinity, insert literal witnesses, close under the two constructors, and keep the smallest *generated witness cost*. This computes shortest words without confusing “smallest witnessed cost” with “least numerical solution of the inequalities.”

Remark 19.4. Min-plus arithmetic is ordinary shortest-path arithmetic written algebraically. Addition along a route becomes $+$; choosing among routes becomes $\min$. The terminology is useful because the same calculation can later retain several boundary costs rather than one coarse maximum.

Let R_Q be the states reachable from q_0 in the underlying controller DFA, temporarily ignoring stack legality, and let

$$B_M = \max\{d_M(p, q) : p \in R_Q, d_M(p, q) < \infty\}.$$

This is a certificate attached to the chosen presentation. Adding redundant reachable controller structure can change B_M without changing L_M . Unreachable controller gadgets are deliberately excluded: no balanced segment of a legal prefix can begin there.

19.5.2 Shortest arrival of a saved stack

For a reachable coordinate (s, q) , let $\rho_M(s, q)$ be the length of its shortest prefix witness. Empty stack gives

$$\rho_M(\varepsilon, q) = d_M(q_0, q).$$

If o is an opener, the last unmatched occurrence of o splits every prefix ending with stack so into:

$$\text{a prefix reaching } (s, p), \quad o, \quad \text{a balanced suffix.}$$

Consequently

$$\boxed{\rho_M(so, q) = \min_{p \in Q} (\rho_M(s, p) + 1 + d_M(\delta(p, o), q)).}$$

This is an equality, not just a construction. Every prefix has the stated factorization, giving the lower bound; concatenating shortest witnesses for the three pieces gives the upper bound.

At horizon N , let S_N be the bounded semantic shadow from Chapter 14. A class $K \in S_N$ may contain several raw coordinates. Its earliest arrival is

$$\alpha_{M,N}(K) = \min\{\rho_M(s, q) : (s, q) \text{ represents } K\},$$

also allowing the tail representative with cost one. That cost is exact: one closer underflows from the empty initial stack.

This order embodies an important rule:

minimize before maximizing.

First remove presentation redundancy by taking the earliest coordinate inside each semantic class. Only then measure the last class to appear. Maximizing raw arrival first would charge late controller syntax that the language does not observe.

Theorem 19.5 (Exact access calculus for controlled Dyck presentations). *For every finite controlled-Dyck presentation and every horizon N ,*

$$a_{L_M}(N) = \max_{K \in S_N} \alpha_{M,N}(K).$$

The quantity is effectively computable by finite balanced-distance relaxation, the stack recurrence above, and the finite controlled-Dyck shadow quotient. Moreover,

$$a_{L_M}(N) \leq \max\{1, (N+1)B_M + N\}.$$

Proof. The bounded future row of a prefix is exactly the controlled-Dyck shadow class of its raw endpoint. The stack recurrence gives the shortest prefix reaching each visible endpoint, while underflow reaches the tail in one step. Therefore $\alpha_{M,N}(K)$ is precisely the first prefix length at which row K appears. The first radius containing every row is their maximum.

For the bound, a prefix leaving stack $s = o_1 \cdots o_h$ factors at its unmatched openers as

$$v_0 o_1 v_1 \cdots o_h v_h,$$

with every v_i balanced. Replace each balanced segment by a shortest word having the same ordered pair of controller endpoints. Its source lies in R_Q , so there are $h+1$ such segments, each of length at most B_M , and h unmatched openers. Preserving both endpoints preserves all controller states at opener boundaries and the final state; balanced replacement preserves the saved opener stack. Thus every reachable coordinate of height $h \leq N$ has a witness of length at most $(h+1)B_M + h$. The tail costs one, and semantic quotienting can only reduce arrival. Taking the maximum proves the display. $\square$

Theorem 19.6 (The linear access certificate is pointwise sharp). *For every horizon $N \geq 0$, some finite controlled-Dyck presentation attains the displayed upper bound. For every $N \geq 1$, one can arrange $B_M = 2$ and*

$$a_{L_M}(N) = 3N + 2 = (N+1)B_M + N.$$

The presentation is allowed to depend on N .

Proof. At $N = 0$, bare rank-one Dyck has $B_M = 0$ and access one, attaining the outer $\max\{1, \cdot\}$ guard.

Fix $N \geq 1$. Use two bracket pairs $(o_1, c_1), (o_2, c_2)$ and neutrals $z_0, \dots, z_{N-1}$. At each level $0 \leq j \leq N$, put controller states T_j, M_j, W_j with the forced balanced detour

$$T_j \xrightarrow{o_1} M_j \xrightarrow{c_1} W_j.$$

Only W_j survives the next saved opener: $W_j \xrightarrow{o_2} T_{j+1}$. Return states $R_{N-1}, \dots, R_0$ form a c_2 -chain, $W_j \xrightarrow{c_2} R_{j-1}$, and only R_0 accepts. Finally, every neutral z_i sends every tier state directly to R_i ; all unspecified transitions die.

The neutral shortcuts ensure that every realizable balanced controller pair has a witness of length at most two. The pair $T_j \rightarrow W_j$ needs exactly the two-letter word o_1c_1 , so $B_M = 2$.

Now inspect the coordinate (o_2^N, W_N) . Factoring any prefix at its N unmatched o_2 symbols creates $N + 1$ balanced segments. Each segment must take T_j to W_j ; otherwise the next o_2 dies. Thus every witness spends two letters in each segment and one on every saved opener, for total at least $2(N + 1) + N = 3N + 2$. The word

$$(o_1c_1o_2)^N o_1c_1$$

attains that length.

Semantic quotienting cannot find an earlier representative: the future c_2^N is accepted only from stack o_2^N and controller state W_N . Hence this bounded row has earliest arrival $3N + 2$. The preceding upper bound supplies equality. $\square$

The construction shows exactly what “sharp” means here. It rules out a smaller uniform coefficient based only on B_M and N . It does not produce one fixed controller that is extremal at every horizon; its alphabet and state set grow with the selected N .

Example 19.7 (A balanced detour visible at horizon zero). Suppose a controller first reaches an accepting empty-stack state after the balanced word $(())$. The accepting horizon-zero row first appears at prefix length four even though its stack is empty. A height-only calculation would predict zero and fail. Balanced distance records the missing cost: $d_M(q_0, q_{\text{accept}}) = 4$.

Combining this theorem with bounded-access regularity gives a dichotomy. Every fixed controlled-Dyck presentation has $a_{LM}(N) = O(N)$. If its language is nonregular, its access is unbounded but still at most linear. The conclusion is language-level; the displayed coefficient remains a presentation-level certificate.

19.6 When is a large presentation cost real?

The last sentence raises a more delicate question than ordinary automaton minimization. Suppose a controller has a very large value of B_M . Does the language force that cost, or did we merely choose a poor regular language away from the well-nested inputs that are actually observed?

There are two different minimizations to perform:

1. minimize the DFA for one fixed regular controller language R ; and
2. vary R itself while preserving only $D \cap R$.

The second operation is easy to miss. A minimal DFA for R may spend most of its states remembering distinctions between malformed bracket words. Those distinctions are real for R but invisible after intersection with D .

19.6.1 The sources the proof actually uses

Let P_D be the set of bracket words that never underflow or mismatch. They need not end with empty stack. Define

$$P_M = \{\delta(q_0, u) : u \in P_D\}.$$

These are controller states reached by legal Dyck prefixes. Now put

$$E_M = \{q_0\} \cup \{\delta(p, o) : p \in P_M, o \text{ an opener}\}.$$

Why not simply use P_M ? Return to the factorization

$$v_0 o_1 v_1 \cdots o_h v_h.$$

The source of v_0 is q_0 . For $i \geq 1$, the source of v_i is the state *immediately after* the opener o_i . Thus E_M names exactly the places where the balanced obligations begin.

This set is finite and computable even though legal prefixes may have unbounded stack height. Start with all endpoints of balanced words from q_0 . Repeatedly take one opener transition and then all possible balanced endpoints. No more than $|Q|$ state additions can occur.

Define the entry-sourced diameter

$$B_M^{\text{ent}} = \max\{d_M(p, q) : p \in E_M, d_M(p, q) < \infty\}.$$

The earlier replacement proof now gives the sharper law

$$a_{L_M}(N) \leq \max\{1, (N + 1)B_M^{\text{ent}} + N\}.$$

Proof walkthrough. Take a prefix ending at stack height h . Its $h + 1$ balanced pieces begin only at states in E_M . Replace each by a shortest balanced word with the same controller endpoints. The h unmatched openers remain literal. The new length is at most $(h + 1)B_M^{\text{ent}} + h$. Horizon N sees only $h \leq N$; the tail still costs one; semantic quotienting still minimizes arrival. Nothing else in the earlier proof changes. $\triangle$

This is a useful example of proof analysis improving a definition. We did not guess a smaller constant and hope. We inspected exactly where the old constant entered the proof and retained only the sources consumed there.

19.6.2 Why DFA quotients can only help

Suppose

$$\pi : M \twoheadrightarrow M'$$

is a surjective morphism of reachable DFAs: it preserves the start state, final output, and every letter action. Then

$$\pi(P_M) = P_{M'}, \quad \pi(E_M) = E_{M'}.$$

Both identities follow by running the same legal prefix in the two machines.

Theorem 19.8 (Entry diameter is monotone under DFA quotient). *For every such morphism,*

$$B_{M'}^{\text{ent}} \leq B_M^{\text{ent}}.$$

Consequently the reachable minimal DFA of one fixed regular language has no larger entry diameter than any DFA recognizing that regular language.

Guided proof. Choose a finite balanced pair (p', q') in the target with $p' \in E_{M'}$. Lift p' to some $p \in E_M$. A balanced target witness w ends, from p , at some q with $\pi(q) = q'$. Now replace w by a shortest source word v from p to that particular q . The source bound gives $|v| \leq B_M^{\text{ent}}$, and applying π shows that the same v sends p' to q' . Hence the shortest target distance is no larger.

The direction of the argument matters. Merely observing that a target word lifts to the source gives the wrong comparison. We must shorten the lifted source path first and then project that shorter path back down. $\square$

The Myhill–Nerode quotient is one such morphism, so ordinary DFA minimization is safe. But it solves only the first minimization problem.

19.6.3 Forgetting behavior away from the Dyck trace

Call two regular languages *Dyck-trace equivalent* when

$$R \equiv_D R' \iff D \cap R = D \cap R'.$$

For a controlled-Dyck language $L \subseteq D$, define

$$\beta_D(L) = \min_{\substack{R \text{ regular} \\ D \cap R = L}} B_{\min(R)}^{\text{ent}}.$$

The definition hides no compactness argument. At least one regular controller exists by hypothesis. Each candidate value is a nonnegative integer, so the nonempty set of candidate values has an attained minimum. What the definition does *not* provide is an algorithm for finding the best R .

It does, however, provide an exact comparison test. Given two controllers, form their product and mark the states where their Boolean outputs disagree. The two Dyck traces are equal exactly when no balanced word takes the product start to a marked state. Balanced-relation saturation decides this, and balanced distance returns the shortest separating word when equality fails.

We may therefore enumerate all finite DFAs, discard those with the wrong Dyck trace, minimize each survivor, and retain the smallest entry diameter seen. The resulting upper bounds eventually reach $\beta_D(L)$ because an optimum exists and is eventually enumerated. The sequence need not announce when it has reached the optimum. Thus $\beta_D(L)$ is computable in the limit from a controller, even though no terminating global minimization algorithm has been proved.

The earlier access theorem can now be applied to a minimizing controller.

Theorem 19.9 (Dyck-trace minimax certificate). *For every controlled-Dyck language inside the fixed envelope D ,*

$$a_L(N) \leq \max\{1, (N+1)\beta_D(L) + N\}.$$

The number $\beta_D(L)$ depends only on L and the selected Dyck envelope, not on one chosen regular controller.

The qualification “selected envelope” is essential. Re-encoding a context-free language with a different bracket alphabet changes the class of regular controls over which the minimum is taken. No invariance under that larger change of coordinates has been proved.

19.6.4 A semantic lower bound: balanced residual access

The minimax definition removes the chosen controller, but it does not by itself tell us what any controller must pay. For that we return to the language and deliberately use only empty-stack boundaries. For $u, u' \in D$, write

$$u \equiv_L^{\text{bal}} u' \iff \forall v \in D, (uv \in L \iff u'v \in L).$$

Two balanced histories are equivalent when no balanced future distinguishes them. This is a Myhill–Nerode relation internal to the monoid D , whose multiplication is concatenation. Closure of D under concatenation is what makes the definition compositional.

The quotient has finite index for a controlled-Dyck language. Indeed, if two balanced histories leave a finite controller in the same state, every future controller run—and hence every balanced test—agrees. The semantic quotient may merge controller states, but it cannot split one controller state.

For each balanced residual class C , measure its first arrival

$$\alpha(C) = \min\{|u| : u \in C\},$$

and define

$$\gamma_D(L) = \max_{C \in D / \equiv_L^{\text{bal}}} \alpha(C).$$

This is an access radius, but not the earlier horizon- N access radius. It asks how late the last full balanced residual first appears. No presentation occurs in its definition.

Theorem 19.10 (Balanced residual access lower bound). *Every regular controller presenting L satisfies*

$$B_M^{\text{ent}} \geq \gamma_D(L),$$

and therefore

$$\beta_D(L) \geq \gamma_D(L).$$

Guided proof. Choose a semantic class C and a shortest balanced word $u \in C$. Let q be the controller state after u . The balanced distance from the start to q is at most $|u|$. Suppose a shorter balanced word u' reached the same q . Starting any continuation $v \in D$ from the same controller state gives the same final output, so

$$uv \in L \iff u'v \in L.$$

Then $u' \in C$, contradicting the choice of u . Thus the distance to q is exactly $\alpha(C)$. The start is a legal segment-entry source, so $B_M^{\text{ent}} \geq \alpha(C)$. Maximize over the semantic classes and then minimize over presentations. $\square$

The proof illustrates the handoff from semantics to presentation. The semantic class tells us that some behavior cannot appear early; a controller state reached by its shortest representative transfers that delay into every finite presentation.

19.6.5 Moving the experiment inside a legal entry context

Top-level balanced boundaries are only one kind of semantic observation. The entry diameter also starts balanced factors immediately after legal openers. Those sources have a language-level meaning.

Let

$$\mathcal{E}_D = \{\varepsilon\} \cup \{xo : x \text{ is a legal Dyck prefix and } o \text{ is an opener}\}.$$

An $x \in \mathcal{E}_D$ remembers a fixed unfinished typed stack. Its valid completions are

$$\mathcal{C}_D(x) = \{z : xz \in D\}.$$

If $v \in D$ is inserted after x , it begins and ends with the same unfinished stack. The domain of possible completions therefore does not depend on which balanced factor v was inserted.

For balanced factors v, v' , define

$$v \equiv_{L,x}^{\text{ent}} v' \iff \forall z \in \mathcal{C}_D(x), (xvz \in L \iff xv'z \in L).$$

This experiment asks whether the surrounding legal nested context can tell the two balanced factors apart. Let

$$\eta_D(L) = \sup_{x \in \mathcal{E}_D} \max_{C \in D / \equiv_{L,x}^{\text{ent}}} \min\{|v| : v \in C\}.$$

At first sight the supremum ranges over infinitely many entry histories. It is nevertheless finite for a controlled-Dyck language. Any one finite controller bounds all of the class arrivals at once.

Theorem 19.11 (Entry-context residual lower bound). *Every regular controller presenting L satisfies*

$$B_M^{\text{ent}} \geq \eta_D(L).$$

Consequently

$$\beta_D(L) \geq \eta_D(L) \geq \gamma_D(L).$$

Guided proof. Fix an entry history x , one contextual factor class C , and a shortest $v \in C$. If p is the controller state after x and q is the state after xv , then p is one of the exact legal segment-entry sources. Suppose a shorter balanced factor v' also took p to q . Every completion $z \in \mathcal{C}_D(x)$ would then start in the same controller state after xv and xv' . Because the two balanced factors leave the unfinished Dyck stack unchanged, they also have exactly the same valid completion domain. Thus v and v' would belong to the same contextual class, contradicting minimality.

So the shortest balanced controller distance from p to q is exactly the first arrival of C . The entry diameter bounds it. Taking all classes and all entry histories proves the first inequality. Choosing $x = \varepsilon$ recovers the top-level balanced residual relation, proving $\eta_D(L) \geq \gamma_D(L)$. $\square$

The definition ranges over infinitely many unfinished stacks, but their effect on a finite controller can be summarized finitely. Form the paired controller on Q^2 and mark a pair when its two coordinates have opposite final outputs. Let $W_{\oplus}$ be the balanced-word relation of this paired controller.

For an unfinished typed stack s , define $G_s \subseteq Q^2$ to be the state pairs from which some legal completion of s reaches opposite outputs. With relational preimage notation,

$$G_{\varepsilon} = W_{\oplus}^{-1}(F_{\oplus}).$$

If opener o is pushed on top of s , every completion first reads a balanced factor, then the forced matching closer c_o , then a completion of s . Therefore

$$G_{so} = W_{\oplus}^{-1}(\delta_{c_o}^{-1}(G_s)).$$

The whole unfinished stack has been replaced by one subset of the finite set Q^2 . There are at most $2^{|Q|^2}$ such summaries.

Pair a controller state with one completion summary. Starting at (q_0, G_ε) , close under balanced controller transitions and opener transitions. The unmatched-opener factorization proves that this finite saturation reaches exactly the state/summary pairs realized by legal prefixes. The start and the opener successors are the semantic entry contexts. At one such pair (p, G_s) , two balanced-reachable endpoints q, q' belong to the same contextual class exactly when $(q, q') \notin G_s$.

Proposition 19.12 (Finite completion-summary saturation). *For every controlled-Dyck language, $\eta_D(L)$ is exactly computable from any finite controller presentation by a terminating finite saturation.*

The saturation can be exponentially large, so this is not an efficiency claim. It is a termination claim with an important contrast: enumerating all regular extensions gives successively better upper bounds on β_D but no known stopping certificate, while completion summaries compute η_D exactly and announce completion.

That exact lower bound can sometimes stop the minimax search itself. Given one regular presentation M , first minimize its ordinary DFA and put

$$b_M = B_{\min(M)}^{\text{ent}}.$$

The semantic computation and this concrete presentation give the finite sandwich

$$\eta_D(L) \leq \beta_D(L) \leq b_M.$$

Proposition 19.13 (Minimax sandwich stopping rule). *If $b_M = \eta_D(L)$, then the supplied presentation attains the global Dyck-trace minimum and*

$$\beta_D(L) = \eta_D(L) = b_M.$$

Both endpoints are computed by terminating finite procedures.

The qualification matters. If the endpoints are different, we have an honest interval, not evidence that the current presentation is wasteful: another trace-equivalent regular language may or may not lower the upper endpoint. If we enumerate presentations, however, the first equality with the fixed semantic lower bound is a sound stopping signal. Thus the method computes β_D exactly on every trace for which $\beta_D = \eta_D$, without assuming that the equality holds universally.

This leaves a sharply named obstruction rather than a vague compiler gap. Define the *entry-atlas sewing defect*

$$\sigma_D(L) = \beta_D(L) - \eta_D(L) \geq 0.$$

For a supplied presentation, the computable gap $b_M - \eta_D(L)$ is an upper bound on $\sigma_D(L)$, not necessarily its value. As trace-equivalent controllers are enumerated, these upper bounds decrease and eventually stabilize at $\sigma_D(L)$. Consequently the defect is computable in the limit from above, and the statement $\sigma_D(L) = 0$ is semidecidable: endpoint equality is a finite witness. No analogous finite certificate for a positive defect is known.

The terminology is deliberately local-to-global. The compatible-cut theorem of Chapter 17 already says when literal observations describe one language. Here the language exists. A positive σ_D would mean something stricter: the separately cheap entry-context observations cannot be carried by one finite regular coordinate system without extra access cost. All families presently understood have $\sigma_D = 0$; whether every controlled-Dyck trace does is open.

This is the semantic interpretation of the apparently mechanistic phrase “segment-entry source.” We are not paying for a bucket that carries stack debt. We are paying for the latest balanced behavior that some legal nested context can force us to distinguish.

19.6.6 The first balanced flip

How can we prove that the minimax cost is positive without inspecting every regular presentation? Look for an obligation visible in the language itself. If L is neither empty nor all of D , define

$$\tau_D(L) = \min\{|w| : w \in D, \mathbf{1}_L(w) \neq \mathbf{1}_L(\varepsilon)\}.$$

This is the first well-nested word whose acceptance differs from epsilon.

Theorem 19.14 (First-flip lower bound). *Every regular controller presenting L satisfies*

$$B_M^{\text{ent}} \geq \tau_D(L),$$

and therefore

$$\beta_D(L) \geq \tau_D(L).$$

Moreover, $\beta_D(L) = 0$ exactly for $L = \emptyset$ and $L = D$.

Proof. Let w be a shortest flip and let $q = \delta(q_0, w)$. The output at q is opposite to the output at q_0 . If some shorter balanced word reached the same q , that shorter word would have the same opposite output, contradicting the choice of w . Hence

$$d_M(q_0, q) = |w| = \tau_D(L).$$

Since $q_0 \in E_M$, the entry diameter is at least this large. The statement survives minimization over presentations.

One-state accepting and rejecting controllers give value zero for D and $\emptyset$. Conversely, if the entry diameter is zero, every balanced word from q_0 must return to q_0 : a different endpoint would require a positive-length witness. All of D therefore has the epsilon output. $\square$

The first flip is the class-access argument with only the empty continuation observed. A shortest flip cannot share its balanced residual class with a shorter word, because membership itself would then agree. Hence

$$\boxed{\beta_D(L) \geq \eta_D(L) \geq \gamma_D(L) \geq \tau_D(L).}$$

The second inequality may be strict: two residual classes can have the same current acceptance bit and still disagree on a later balanced experiment.

The access profile gives another lower bound. For $N \geq 1$,

$$\beta_D(L) \geq \left\lceil \frac{a_L(N) - N}{N + 1} \right\rceil.$$

At $N = 0$, the outer guard means that $a_L(0) > 1$ implies $\beta_D(L) \geq a_L(0)$, while $a_L(0) = 1$ alone implies nothing positive.

19.6.7 A family where the semantics forces the full cost

Consider again the depth-threshold languages. Epsilon has not reached depth r . Any balanced word that changes this truth value needs at least r openers and the same number of closers. The word $o^r c^r$ attains the bound, so

$$\tau_D(D^{\geq r}) = \tau_D(D^{< r}) = 2r.$$

The ordinary threshold controller has pre-threshold states $0, 1, \dots, r-1$ and one absorbing hit state. From pre-state i , a balanced word either returns to i or first reaches hit. The shortest hitting word is $o^{r-i} c^{r-i}$, of length $2(r-i)$. Its entry diameter is therefore $2r$. The lower and upper bounds meet:

$$\boxed{\beta_D(D^{\geq r}) = \beta_D(D^{< r}) = 2r.}$$

Notice the asymmetry between the two branches. For $D^{\geq r}$, horizon-zero access is already $2r$, so access forces the minimax value. For $D^{< r}$, access is small; the first acceptance flip supplies information that access does not.

19.6.8 A punctured threshold exposes hidden future obligation

The first flip is not the complete semantic lower bound. For $r \geq 2$, let

$$L_r^\circ = \{\varepsilon\} \cup D^{\geq r}.$$

This language accepts epsilon and every balanced word that reaches height r , but rejects nonempty balanced words that remain below the threshold.

The word oc already flips the epsilon output, so

$$\tau_D(L_r^\circ) = 2.$$

If the first flip measured the whole presentation cost, every member of this family would have the same lower bound. Balanced continuations reveal what the acceptance bit hides.

There are exactly three balanced residual classes:

history kind	first representative	arrival
epsilon	ε	0
nonempty, below threshold	oc	2
threshold already reached	$o^r c^r$	$2r$

To see why the middle row is one class, take any nonempty balanced history that stayed below r . Concatenation starts the next balanced component at height zero, so the combined word is accepted exactly when the continuation reaches r . Once a history has reached r , every balanced continuation is accepted. Epsilon differs from the middle class on the empty continuation and from the hit class on a short below-threshold continuation.

Therefore

$$\gamma_D(L_r^\circ) = 2r.$$

The ordinary threshold controller, with one extra bit distinguishing the fresh start from later below-threshold returns, has entry diameter $2r$. The semantic lower bound and this presentation meet.

Theorem 19.15 (Punctured-threshold separation). *For every $r \geq 2$,*

$$\tau_D(L_r^\circ) = 2 < \gamma_D(L_r^\circ) = \beta_D(L_r^\circ) = 2r.$$

This rules out the attractive guess $\beta_D = \tau_D$. The next example tests whether top-level balanced residual access captures everything.

19.6.9 A nested context sees what top-level residuals miss

For $r \geq 3$, let H_r contain the nonempty Dyck words whose first primitive top-level component is either the two-letter word oc or reaches height at least r . A primitive component is a nonempty balanced word that returns to height zero only at its final symbol. Once the first primitive component has closed, later components cannot change the answer.

At a top-level balanced boundary there are only three residual behaviors:

1. epsilon still waits to inspect the first component;
2. a word whose first component succeeded accepts every balanced future;
3. a word whose first component failed rejects every balanced future.

The successful class first appears at oc , in length two. For every $r \geq 3$, the word $oocc$ is a failing first component and supplies the third class in length four. Hence

$$\tau_D(H_r) = 2, \quad \gamma_D(H_r) = 4.$$

Neither number grows with r .

Now place the observation immediately after the first opener, so $x = o$. A balanced factor inserted there has three contextual behaviors:

factor kind	first representative	arrival
empty	ε	0
nonempty, below relative height $r - 1$	oc	2
relative height $r - 1$ reached	$o^{r-1}c^{r-1}$	$2(r - 1)$

An immediate closer distinguishes the empty factor from the nonempty-low factor: the former makes the special component oc , while the latter makes a longer below-threshold component. A hit factor is permanently accepting and is distinct from both. Therefore this one entry context already forces

$$\eta_D(H_r) \geq 2(r - 1).$$

A finite controller meets the bound. Before the first component closes, it tracks exact height below r , distinguishes the state just after the first opener from a later return to height one, and enters an absorbing accept or reject state once the first component is decided. Crucially, the immediate oc route and the threshold-hit route use the same accepting sink; minimizing a version with two accepting sinks performs exactly this merger. From the first entry state, the shortest balanced factor that reaches the threshold is $o^{r-1}c^{r-1}$. Every other legal entry needs no longer balanced transition. Thus its entry diameter is $2(r - 1)$.

Theorem 19.16 (First-component contextual separation). *For every $r \geq 3$,*

$$\boxed{\tau_D(H_r) = 2, \quad \gamma_D(H_r) = 4, \quad \eta_D(H_r) = \beta_D(H_r) = 2(r - 1).}$$

In particular, $\beta_D(H_r) > \gamma_D(H_r)$ for $r \geq 4$.

So $\beta_D = \gamma_D$ is false in general. The presentation cost is not an artifact: it is forced by a semantic experiment inside a legal nested context. The remaining equality question is the sharper one

$$\beta_D(L) \stackrel{?}{=} \eta_D(L).$$

No general answer is claimed.

19.6.10 A smaller controller need not be the cheapest controller

The previous examples compare a semantic lower bound with one presentation. They do not yet ask how two different finite resources compete. State count measures how many ordinary DFA coordinates a controller uses; entry diameter measures how late a balanced transition first appears from a legal segment entry. There is no reason in advance for one controller to optimize both.

For a trace $L \subseteq D$ and a nonnegative budget b , define the *state budget at access b* by

$$\nu_D(L; b) = \min\{|Q_M| : D \cap L(M) = L, B_M^{\text{ent}} \leq b\},$$

with value ∞ when no finite controller meets the budget. Because unreachable states can be discarded, the minimum may be taken over reachable controllers. The pair

$$(|Q_M|, B_M^{\text{ent}})$$

is partly ordered coordinatewise: a presentation is interesting only when no other one uses both fewer states and no more access.

Here is a small trace for which the two objectives disagree. Work with one bracket pair o, c . For a nonempty Dyck word w , let $t(w)$ be the length of its final consecutive run of closers, and set

$$K_3 = \{\varepsilon\} \cup \{w \in D \setminus \{\varepsilon\} : t(w) \equiv 2 \pmod{3}\}.$$

This definition looks global, but a reset makes it regular on the full bracket alphabet. Take states r_0, r_1, r_2 , make r_0 initial and accepting, send every opener transition to r_1 , and cycle closers by

$$r_1 \longmapsto r_2 \longmapsto r_0 \longmapsto r_1.$$

In a nonempty balanced word, the final opener erases everything before it. The remaining $t(w)$ closers end at r_0 exactly when $t(w) \equiv 2 \pmod{3}$. Thus this three-state controller has trace K_3 . The states are ordinary-DFA distinguishable: r_0 is final, and one closer separates r_1 from r_2 . Hence three is not merely the size of this construction; the exact trace decision below will show that it is the minimum over all controllers.

The controller is small but its entry diameter is six. Its only legal entry sources are r_0 and r_1 . From r_0 , the balanced words

$$\varepsilon, \quad oc, \quad o^3c^3$$

first reach r_0, r_2, r_1 at lengths 0, 2, 6. From r_1 , every balanced endpoint appears by length four. Therefore

$$B_{A_3}^{\text{ent}} = 6.$$

This six is not intrinsic to the trace. Consider the following controller, whose only final state is q_0 :

	o	c	final?
q_0	q_2	q_4	yes
q_1	q_4	q_0	no
q_2	q_4	q_3	no
q_3	q_2	q_5	no
q_4	q_3	q_1	no
q_5	q_3	q_2	no

The table has a natural derivation. On a legal prefix, write h for stack height and t for the current terminal closer-run length. Track

$$(h \bmod 3, [t + h \equiv 2 \pmod{3}]) \in \mathbb{Z}_3 \times \{0, 1\}.$$

A closer adds one to t and subtracts one from h , so the Boolean coordinate is invariant. An opener resets t to zero and increases h , so it resets the Boolean coordinate to $[h + 1 \equiv 2]$. At height zero, the Boolean is exactly the desired terminal-descent test. Start at $(0, 1)$ so epsilon is accepted, and make it the only final coordinate. In formulas,

$$\begin{aligned} \delta((m, b), o) &= (m + 1, [m + 1 \equiv 2]), \\ \delta((m, b), c) &= (m - 1, b). \end{aligned}$$

The displayed q -table is this controller under the relabeling

q_i	q_0	q_1	q_2	q_3	q_4	q_5
(m, b)	$(0, 1)$	$(1, 1)$	$(1, 0)$	$(0, 0)$	$(2, 1)$	$(2, 0)$

Thus the extra coordinates are not solver debris: height residue makes the obligation bit compositional and shortens every balanced access obligation.

To verify the trace without testing words up to an arbitrary cutoff, form the balanced product with the three-state controller. Starting at (q_0, r_0) , the exact balanced-reachable pairs are

$$(q_0, r_0), \quad (q_3, r_1), \quad (q_3, r_2).$$

The first pair has two accepting coordinates; the other two have two rejecting coordinates. No balanced word can separate the controllers, so the table has trace exactly K_3 .

Its legal entry sources and finite balanced distances are

q_0	$q_0 : 0, q_3 : 2$
q_2	$q_2 : 0, q_1 : 2$
q_3	$q_3 : 0, q_0 : 4$
q_4	$q_4 : 0, q_5 : 2.$

Consequently $B_{A_6}^{\text{ent}} = 4$. The semantic value is also at least four. Place a factor experiment after the legal history $x = o^3$ and complete the unfinished stack by c^3 . The factors ε and oc yield terminal

descents of lengths three and four and are rejected, while *oocc* yields terminal descent five and is accepted. No balanced factor of length one, two, or three supplies that accepting contextual class. Therefore

$$4 \leq \eta_D(K_3) \leq \beta_D(K_3) \leq 4.$$

Theorem 19.17 (Terminal-descent sewing collapse). *For the trace K_3 ,*

$$\boxed{\eta_D(K_3) = \beta_D(K_3) = 4.}$$

The three-state presentation is state-minimal but not access-optimal.

The equality closes the candidate sewing gap. This example does *not* have positive entry-atlas defect. What it gives instead is an exact presentation tradeoff.

Why the lower-left corner is exact

There are infinitely many balanced words but only finitely many state pairs. For a proposed n -state controller, introduce an unknown relation $W \subseteq Q \times Q$. Require W to contain every identity pair, to be closed under relational composition, and to satisfy matched wrapping:

$$(\delta_o(p), q) \in W \implies (p, \delta_c(q)) \in W.$$

Every such W contains the endpoint pair of every balanced word. Conversely, the exact balanced-word relation itself satisfies these closure laws. This two-sided observation is what makes a finite constraint encoding exact.

Use a second closed set to contain every legal-prefix state. At the start and after every opener from that set, require each W -endpoint to equal an endpoint reached by one of

$$\varepsilon, \quad oc, \quad oocc, \quad ococ.$$

These are all one-bracket Dyck words of length at most four, so the condition is exactly $B_M^{\text{ent}} \leq 4$. Apply the same construction to the product with the three-state reference controller and forbid a balanced-reachable pair with opposite outputs. That condition is exact trace equality, not finite sampling.

The resulting finite system is unsatisfiable with five available states and satisfiable with six. Unused states are allowed, so a smaller controller could be padded to a five-state model; unsatisfiability therefore rules out every size through five. Independently, exhaustive enumeration of the one- and two-state total binary DFAs, followed by exact balanced-product comparison, finds no presentation of K_3 .

A second exact decision reaches the same lower bound without guessing a symbolic balanced relation. It asks the solver for a concrete five-state controller, computes that controller's exact balanced closure, and adds either the concrete trace equation or legal-entry endpoint equation that the model violates. Every refinement is necessary for any cost-four presentation, and each excludes its generating model. This counterexample-guided route reaches unsatisfiability after 26 iterations. An independent one-hot encoding reaches the same conclusion after 25. The three routes share the Z3 kernel but not the mathematical encoding of balanced reachability.

Theorem 19.18 (Exact state/access frontier of K_3). *The minimum size of a controller presenting K_3 is three, and*

$$\nu_D(K_3; 4) = 6.$$

The nondominated state/access pairs are exactly

$$\boxed{(3, 6) \quad \text{and} \quad (6, 4)}.$$

This last lower bound is computer-assisted: it trusts the finite encoding and the Z3 solver kernel. The satisfying model is exported and rechecked by the independent trace and distance algorithms, but the unsatisfiability proof is not yet a proof-assistant certificate.

The conceptual lesson is asymmetric. In the expensive full-trace family below, extra off-Dyck distinctions create useless cost and minimization removes them. For K_3 , extra ordinary-DFA coordinates reduce the cost of carrying the observed balanced transitions. Ordinary state minimization and fixed-envelope access minimization are therefore genuinely different optimization problems.

19.6.11 The terminal-descent modulus atlas

The modulus-three example is not isolated. For $m \geq 2$, define

$$K_m = \{\varepsilon\} \cup \{w \in D \setminus \{\varepsilon\} : t(w) \equiv m - 1 \pmod{m}\}.$$

The nearby moduli first appear to disagree: the natural two-state controller for K_2 already has cost four, the natural three-state controller for K_3 has cost six but can be recoded to cost four, and the natural four-state controller for K_4 has cost eight but can be recoded to cost six. The semantic calculation explains both the pattern and the exceptional first term.

Fix a legal entry immediately after an opener, at stack height $h \geq 1$. A completion of that stack factors uniquely as

$$b_h c b_{h-1} c \cdots b_1 c b_0, \quad b_i \in D.$$

There are two qualitatively different cases. If every b_i is empty, the completion is the bare descent c^h and remembers the factor's terminal run. If some b_i is nonempty, take the last one. Its last opener resets the terminal run, so the final answer depends only on the completion and not on the preceding factor. Thus every nested entry has exactly two contextual factor classes: one distinguished residue and its complement.

More explicitly, the natural controller enters state r_1 after every opener. The factor $o^k c^k$ reaches $r_{1+k \bmod m}$, first at length $2k$ for $0 \leq k < m$. At height h , the bare completion accepts exactly the state $r_{-h \bmod m}$. As h ranges through the positive integers, its shortest arrival ranges up to $2(m-1)$.

The empty-stack context creates one exception. For $m \geq 3$, oc is already rejected and the second root class arrives at length two. For $m = 2$, both ε and oc are accepted; $oocc$ is the first rejected balanced word, so the root radius is four. We have proved the semantic half of the following theorem.

Theorem 19.19 (Terminal-descent family law). *For every $m \geq 2$,*

$$\eta_D(K_2) = \beta_D(K_2) = 4, \quad \eta_D(K_m) = \beta_D(K_m) = 2(m-1) \quad (m \geq 3).$$

The matching presentation makes the proof unusually transparent. Use states

$$(a, b) \in \mathbb{Z}_m \times \{0, 1\}, \quad b = [t + h \equiv m - 1 \pmod{m}].$$

A closer adds one to t and subtracts one from h , preserving the bit. An opener resets t and increments h . Hence

$$\begin{aligned} \delta((a, b), o) &= (a + 1, [a + 1 = m - 1]), \\ \delta((a, b), c) &= (a - 1, b). \end{aligned}$$

Take $(0, 1)$ as start and sole final state. At height zero the bit is exactly the terminal-descent test, while the accepting start handles epsilon. A balanced factor preserves a and its two possible bit values are precisely the two semantic classes just calculated. Their shortest state arrivals are therefore their shortest semantic arrivals. This $2m$ -state presentation has entry diameter $\eta_D(K_m)$, proving the theorem's upper bound.

This is stronger than a list of examples: the proposed positive sewing defect vanishes on an infinite family, for a structural reason. The semantic completion classes themselves can be made into finite controller coordinates. But state count and access cost still pull in different directions.

Even moduli need fewer coordinates

When m is even, the $2m$ coordinates contain redundancy. Put $\sigma = t + h$. Acceptance requires the odd residue $m - 1$. If σ is even, the prefix is hopeless until the next opener, and only the parity of h must be retained. If σ is odd, an m -cycle can compute the residue positionally.

Use cycle states $P_0, \dots, P_{m-1}$ and hopeless states H_e, H_o . Start and accept at P_{m-1} , and set

$$\begin{aligned} P_p &\xrightarrow{c} P_{p+1}, & P_p &\xrightarrow{o} \begin{cases} P_0, & p \text{ odd}, \\ H_e, & p \text{ even}, \end{cases} \\ H_e &\xrightarrow{c} H_o, quad H_o &\xrightarrow{c} H_e, & H_e &\xrightarrow{o} P_0, quad H_o &\xrightarrow{o} H_e, \end{aligned}$$

with cycle indices modulo m .

On a nonempty legal prefix, P_p means that σ is odd and $p = t \bmod m$; H_e, H_o mean that σ is even and record the parity of h . The transition table preserves this invariant because an even cycle length preserves the parity of p . At height zero, P_p has $p = \sigma = t$, while a hopeless state cannot accept. The trace is K_m .

The entry sources are only P_{m-1}, P_0, H_e . Their balanced endpoints are reached by factors $o^p c^p$, with the two hopeless endpoints appearing by length at most four. The greatest necessary length is $2(m-1)$. We obtain:

Theorem 19.20 (Even positional construction). *For every even m , an $(m+2)$ -state controller presents K_m at the optimal entry cost $\beta_D(K_m)$. At $m = 4$ it is isomorphic to the six-state controller displayed above.*

The theorem is a state upper bound, not a general minimality result. Exact finite decisions give the first three atlas entries

$$K_2 : (2, 4), \quad K_3 : (3, 6), (6, 4), \quad K_4 : (4, 8), (6, 6).$$

It is tempting to conjecture that optimal-cost presentations require $m + 2$ states for even $m \geq 4$ and $2m$ states for odd $m \geq 3$. The constructions give those upper bounds, and moduli three and four give matching lower bounds. Modulus five is the first unresolved test. That is exactly where the theorem ends and a new research problem begins.

19.6.12 A family where minimization removes the cost

Now we deliberately build the opposite phenomenon. Use one bracket pair o, c , and for $m \geq 1$ let

$$F_m = \{(oc)^j c : 0 \leq j \leq m\}, \quad R_m = \{o, c\}^* \setminus F_m.$$

Every word in F_m underflows on its last closer. Therefore

$$D \cap R_m = D.$$

Yet the minimal DFA of R_m must distinguish the balanced prefixes

$$\varepsilon, oc, (oc)^2, \dots, (oc)^m.$$

After $(oc)^j$, the illegal continuation c is rejected, and the remaining longer forbidden continuations depend on j . These are genuine distinct right languages of R_m even though none of their final rejecting words lies in D .

Example 19.21 (An expensive minimal controller for the full trace). The minimal DFA has states $q_0, \dots, q_m$ for the displayed balanced prefixes, intermediate states $h_0, \dots, h_{m-1}$ after their next opener, one rejecting endpoint, and one accepting sink: $2m + 3$ states in all.

The balanced word $(oc)^j$ first takes q_0 to q_j in length $2j$. The shortest balanced route from q_0 to the accepting sink is $oocc$, of length four. Every nonempty balanced word from an intermediate h_j goes to that sink in two steps. Hence

$$B_{\min(R_m)}^{\text{ent}} = \max\{4, 2m\}.$$

These values diverge with m , even though every controller has exactly the same Dyck trace D . The one-state universal controller shows $\beta_D(D) = 0$.

This example gives a precise warning:

minimal for the regular control language is not minimal for its well-nested observation.

The cost is not fake inside the DFA; Myhill–Nerode minimization cannot remove it. It becomes irrelevant only after we decide that behavior away from D is outside the observation interface.

Research pause

The minimax construction is a successful layer change, but it is not yet a complete classification. We know that a best coefficient exists and that semantic data can force it. We do not yet know how to compute it from an arbitrary controller, whether changing the Dyck envelope admits a canonical normalization, or which additional finite boundary data would make the minimization effective. These are mathematical questions about presentation equivalence, not requests for a more elaborate stack simulator.

19.7 Boolean transformations

Let $\Phi : \{0, 1\}^r \rightarrow \{0, 1\}$ be a pointwise Boolean operation and let all languages use the same alphabet. The output future row is obtained by applying Φ coordinatewise to the tuple of component rows. Therefore

$$G_{\Phi(L_1, \dots, L_r)}(I, N) \leq \prod_{j=1}^r G_{L_j}(I, N),$$

$$q_{\Phi(L_1, \dots, L_r)}(N) \leq \prod_{j=1}^r q_{L_j}(N).$$

Complement is exact: negating every row coordinate preserves row equality, so

$$G_{\bar{L}} = G_L, \quad q_{\bar{L}} = q_L, \quad a_{\bar{L}} = a_L.$$

Access under intersection cannot be bounded from marginal access alone. For even $T \geq 2$, let

$$F_T = \{a, a^T\}, \quad R_{\text{even}} = \{a^{2j} : j \geq 0\}.$$

At horizon zero, each marginal has access one, but

$$F_T \cap R_{\text{even}} = \{a^T\}$$

has access T . A fixed two-state filter can create arbitrary joint access delay through correlation.

Example 19.22 (A two-state controller creates late joint access). Take $T = 6$. At horizon zero, both $F_6 = \{a, a^6\}$ and the even-length language expose accepting and rejecting rows by prefix length one. Their intersection is the singleton $\{a^6\}$, whose accepting row is not reached until length six. Nothing in the two marginal access radii records the parity correlation selecting the late member.

19.8 Unary words and factor appearance

For a unary language $L \subseteq \{a\}^*$, form its characteristic sequence

$$x_i = \mathbf{1}[a^i \in L].$$

The bounded row at prefix a^i is the length- $(N+1)$ factor

$$B_N(a^i) = (x_i, x_{i+1}, \dots, x_{i+N}).$$

Thus $q_L(N)$ is the number of distinct length- $(N+1)$ factors of the one-sided word x .

Let $A_x(n)$ be the least prefix length containing every length- n factor, and let $S_x(n)$ be the largest starting position of a first occurrence plus one under the corresponding indexing convention. These are standard appearance functions in combinatorics on words [7]. With our zero-based prefix-radius convention,

$$a_L(N) = A_x(N+1) - (N+1) = S_x(N+1) - 1.$$

Observation count becomes subword complexity; access becomes factor-appearance delay.

19.9 Worked family gallery

19.9.1 Singletons

For $L_T = \{a^T\}$,

$$q_{L_T}(N) = \min(T, N) + 2,$$

and

$$a_{L_T}(N) = \begin{cases} T, & N < T, \\ T+1, & N \geq T. \end{cases}$$

The extra final step reaches the dead residual after the accepting prefix.

19.9.2 Square lengths

The factor complexity of the characteristic word of perfect squares grows quadratically, so

$$q_{L_\square}(N) = \Theta(N^2).$$

This is an imported fact about the square-indicator word [11], transported through the unary identity $q_L(N) = p_x(N+1)$.

The access radius admits a sharper elementary formula.

Theorem 19.23 (Exact access radius for square lengths). *For every $m \geq 0$,*

$$a_{L_\square}(2m) = (m+1)^2 + 1, \quad a_{L_\square}(2m+1) = (m+1)^2 + 2.$$

Proof. Write $x_j = \mathbf{1}[j \text{ is a square}]$ and put $n = N+1$, so horizon- N rows are length- n factors of x . Consecutive squares t^2 and $(t+1)^2$ are separated by $2t$ zero positions.

First let $N = 2m$, so $n = 2m+1$. A block of n zeros first fits immediately after $(m+1)^2$, because the preceding zero runs have length at most $2m$ and the run after $(m+1)^2$ has length $2m+2$. Its starting position is

$$I = (m+1)^2 + 1.$$

Every factor containing exactly one 1 occurs no later than I : place that 1 at any prescribed offset $r \in \{0, \dots, 2m\}$ by taking the factor that starts at $(m+1)^2 - r$. The neighboring squares lie outside this window. Every factor containing at least two 1s must occur before $(m+1)^2$, since all later consecutive-square gaps exceed the window width. Thus every factor has appeared by I , while the all-zero factor appears there for the first time. Hence $a_{L_\square}(2m) = I$.

Now let $N = 2m+1$, so $n = 2m+2$. The factor $0^{n-1}1$ first ends at $(m+2)^2$: earlier squares have fewer than $n-1$ zeros immediately before them. Its first starting position is

$$(m+2)^2 - (n-1) = (m+1)^2 + 2 =: J.$$

The all-zero factor already starts at $(m+1)^2 + 1 < J$. A factor with exactly one 1 at any offset other than the final offset occurs around $(m+1)^2$ and starts before J . A factor with at least two 1s must use a pair no later than $m^2, (m+1)^2$, because the next square gap is $2m+3 > n-1$; it too starts before J . Therefore $0^{n-1}1$ is the last factor to appear and $a_{L_\square}(2m+1) = J$. $\square$

The quadratic count is classical ownership; the parity formula is an elementary FPRD derivation, checked independently on finite horizons but carried here by the unbounded consecutive-square proof rather than by enumeration.

19.9.3 Bare Dyck

For bracket rank k ,

$$q_{D_k}(N) = 1 + \sum_{h=0}^N k^h, \quad a_{D_k}(N) = \max(1, N).$$

Observation count can be exponential while access remains linear.

19.9.4 Depth threshold

Theorem 19.24 (Exact depth-threshold profiles). *Let $D_k^{\geq r}$ be the balanced words over k bracket types whose stack height reaches at least $r \geq 1$. Then*

$$q_{k,r}^{\geq}(N) = 1 + \sum_{h=0}^N k^h + \sum_{h=\max(0, 2r-N)}^{r-1} k^h,$$

$$a_{k,r}^{\geq}(N) = \max(2r, N).$$

For the complementary balanced language $D_k^{< r}$ that stays below r ,

$$q_{k,r}^{<}(N) = 1 + \sum_{h=0}^{\min(N, r-1)} k^h, \quad a_{k,r}^{<}(N) = \max(1, \min(N, r-1)).$$

The formulas remain unchanged when neutral letters are added.

Proof. After the threshold has been seen, every typed stack s of height at most N gives one visible row, distinguished by its canonical closing word. These contribute $\sum_{h=0}^N k^h$. Before the threshold, a stack of height $h < r$ needs at least $r - h$ further opens and then r closes, so its shortest accepting future has length $2r - h$. It becomes visible exactly when $h \geq 2r - N$, producing the second sum. Failure and every not-yet-visible coordinate share one all-false row.

A post-threshold empty stack first appears by rising to r and returning, at prefix length $2r$. A post-threshold stack of height N first appears at length N once $N \geq r$. Every other visible row appears no later than the larger of these witnesses, proving $\max(2r, N)$.

For the below-threshold language, the only live rows are typed stacks through height $\min(N, r-1)$, and one all-false row contains failure and every threshold-crossing prefix. A live height- h row first appears after h openers; failure appears after one closer. This proves the remaining formulas. Neutral letters change neither stack height nor the shortest witnesses. $\square$

At horizon zero, $q_{k,r}^{\geq}(0) = 2$ but $a_{k,r}^{\geq}(0) = 2r$. Thus even within a finite-controller family, fixed observation count does not bound access across presentations. This does not contradict the affine theorem: increasing r changes the finite controller and its certificate.

19.10 What should a growth theory explain?

Closed formulas are only a beginning. A useful classification should address:

- realizability: which two-parameter profiles can come from languages or from finitely presented dynamics;
- inequalities among row count, rank, access, exposure, and controller cost;
- transformations under coding, Boolean operations, quotient, and presentation refinement;
- asymptotic regimes and sharp examples separating them; and
- uniformization: when nonuniform fixed-horizon recognizers assemble into one controlled dynamic.

Exercise 19.25. Reprove the bounded-access regularity criterion, explicitly naming the point where the global plateau theorem is used. See the [solution](#).

Exercise 19.26. Verify the fixed-filter intersection example for $T = 6$ at horizon zero. Then explain why the product bound on row counts does not imply a comparable bound on access. See the [solution](#).

Exercise 19.27. Compare $q(N)$ and $a(N)$ for bare Dyck recognition at ranks one and two. Which coordinate detects the bracket rank most strongly? See the [solution](#).

Exercise 19.28. A rank-one controller has states p, q , starts at p , and leaves its state unchanged on brackets. One neutral letter x toggles p and q ; only q accepts. Compute the balanced-distance matrix, the shortest arrivals of empty-stack states, and $a_{L_M}(0)$. Explain why maximizing raw arrivals would give the wrong answer if neither state accepted. See the [solution](#).

Exercise 19.29. For $m = 2$, list the forbidden set F_m in the expensive full-trace example. Write the rejected-continuation sets at q_0, q_1, q_2 and explain why the states are pairwise distinct even though $D \cap R_m = D$. Finally compute the entry diameter and compare it with $\beta_D(D)$. See the [solution](#).

Exercise 19.30. Take $r = 3$ in the punctured-threshold language. Give one balanced test that distinguishes epsilon from the nonempty-below-threshold class, and one that distinguishes the nonempty-below-threshold class from the hit class. Explain why the three class arrivals are 0, 2, 6, even though the first acceptance flip occurs at length two. See the [solution](#).

Exercise 19.31. Take $r = 4$ in H_r . Verify that the top-level balanced residual classes first arrive at lengths 0, 2, 4. Then work immediately after the first opener and exhibit contextual factor classes arriving at 0, 2, 6. Which completion distinguishes the empty factor from the nonempty-low factor? See the [solution](#).

Exercise 19.32. For the three-state terminal-descent controller, compute the endpoint of ε , oc , $oocc$, and o^3c^3 from each legal entry source. Explain why the entry diameter is six. Then explain why allowing unused states makes unsatisfiability at a five-state budget rule out every smaller cost-four presentation. See the [solution](#).

Research consultations at the frontier

The purpose of this final chapter is not to announce a roadmap already chosen. It is to expose the decisions for which the preceding mathematics prepares us. Each direction below has a different notion of leverage and a different risk of becoming formalism without explanatory gain.

20.1 Direction one: classify observation and access growth

The functions $G_L(I, N)$, $q_L(N)$, and $a_L(N)$ already separate regular, square-length, singleton, Dyck, and controlled-stack families. The next questions are structural.

Which monotone two-parameter functions are realizable? Does finite presentation impose recursive, algebraic, submultiplicative, or affine constraints? Can profile equivalence classify meaningful dynamic families, or is it too coarse? Which transformations preserve asymptotic regimes?

This direction offers near-term theorem leverage because it has explicit families, candidate inequalities, and executable finite oracles. Its risk is cataloguing formulas without discovering constraints that explain them.

Research pause

Would you prioritize a realizability theorem, a transformation calculus, or a taxonomy of sharply separated examples? A realizability theorem is deeper but may be difficult; a transformation calculus may immediately support later work; examples can reveal the right conjectures but can also become a census without a theory.

20.2 Direction two: uniformization cost

Fixed cuts admit exact minimal layered recognizers and one-way protocols. The sewing theorem says literal compatible data determine a language. The plateau theorem says one global adjacent plateau yields a uniform DFA. Between these facts lies a quantitative question:

How much additional state, access depth, exposure, or presentation structure is needed to turn a family of nonuniform finite observations into one uniform recognition dynamic?

The square-length example proves that fixed-cut width alone gives no bound. Controller-specific affine

access bounds suggest that finite presentation may restore control in restricted families. A successful theory would connect nonuniform automata, branching programs, streaming, learning, and residual completion.

Its risk is that “uniformization cost” may split into several inequivalent notions. That is not fatal if the book states which operational model each cost belongs to.

20.3 Direction three: presentation transformations

Fixed-envelope coding, CFG normalization, and controller products all change a presentation while preserving an observation. They incur different costs: alphabet size, block width, controller states, action macros, observation lag, and residual refinement.

A transformation theory could ask for functorial laws and optimal tradeoffs. When does a recoding preserve growth up to a sharp modulus? When can presentation refinement increase access without increasing residual count? Which quantities are invariants of language, dynamic, or presentation class?

This direction stays close to constructive algorithms and may yield practical compiler guidance. Its risk is optimizing representations before the right equivalence notion has been chosen.

20.4 Direction four: partial-data gluing obstructions

Total compatible cut fields sew trivially once the diagonal law is imposed. A nontrivial local-to-global theory needs less complete data or stronger global requirements.

Promising variants include:

- partially specified tables whose overlaps may conflict;
- local fields required to arise from a fixed presentation doctrine;
- approximate or noisy observations with error bounds;
- local controllers whose actions cannot be made globally natural; and
- cohomological or sheaf-like obstructions that detect more than the total Boolean transport identity.

This direction most closely resembles a genuine sewing obstruction. Its risk is importing sophisticated language before a small example exhibits a real failure that the machinery explains.

Research pause

Before adopting sheaf or cohomological terminology, construct the smallest partial recognition dataset whose every local piece is realizable but whose whole is not. What information lives on overlaps, and what exactly fails to glue? If no such example survives scrutiny, the abstraction has not earned its place.

20.5 Direction five: difficult automata and complexity problems

The program should eventually justify itself outside its internal architecture. Candidate applications include lower bounds where finite residual shadows, access delay, or presentation-generated actions expose a resource missed by ordinary state complexity.

Promising neighbors include nonuniform versus uniform automata, streaming space, communication complexity, profinite methods, symbolic dynamics, subword complexity, and visibly pushdown systems. An analogy counts as progress only when it transfers a theorem, inequality, construction, or counterexample.

The standard for entering this phase should be high: the FPRD invariant must solve, sharpen, or reorganize an external problem more effectively than its native tools.

20.6 Direction six: broader finitely presented dynamics

The architecture has five pieces: finite presentation, a generated dynamic, finite observations, completion, and readout. This form is not specific to languages and may apply to other finitely presented systems. But broadening too early would erase the discipline supplied by words, residuals, and exact membership.

A broader application should be attempted only when it carries a precise functor, invariant, representation theorem, or counterexample out of language recognition. “This resembles dynamics” is not enough.

20.7 A current judgment

The narrowest complete next mathematical path is:

1. finish the general laws of access and their classical ownership;
2. establish the singleton, square, and bare-Dyck families completely;
3. develop transformation laws, beginning with complement, Boolean row bounds, and coding moduli;
4. then treat depth-threshold and arbitrary finite-controller access as a controlled-family extension.

This path has direct proofs, independent computations, and clear counterexamples. It earns a classification vocabulary before attempting the more speculative partial-gluing or broader-dynamics programs.

The judgment is provisional. A decisive comparator theorem, counterexample, or human research-direction intervention may change it. That possibility is not a weakness in the plan; it is why the book was written.

Example 20.1 (What a theorem-sized frontier choice looks like). “Study growth” is too broad to guide a proof. One theorem-sized version is: every language controlled by a fixed finite balanced controller has access radius bounded affinely in the future horizon, with constants computable from the controller. It names a domain, invariant, quantitative conclusion, and source of constants. A counterexample with superlinear access inside that domain would decisively refute it; a proof would transfer finite presentation into a uniformization constraint.

Exercise 20.2. Choose two frontier directions and write the strongest case for each. Name one theorem-sized success criterion and one failure signal that should cause the program to stop or redirect. There is no unique solution; the backmatter gives a decision rubric. See the [solution](#).

Solutions and substantial hints

The solutions emphasize the mathematical move each exercise was designed to practice. Several research-facing exercises have rubrics rather than unique answers.

Solution to Exercise 1.3. At depth zero, $K_0 = \{\varepsilon\}$ has the accepting row and all K_n , $n > 0$, share the rejecting tail. At depth one the classes are K_0 , K_1 , and $\{K_n : n > 1\}$; at depth two they are K_0, K_1, K_2 , and the remaining tail. Restriction fixes K_0, K_1 and sends both the K_2 class and the depth-two tail to the depth-one tail. The compatible all-tail thread cannot equal K_j : at depth j , the state K_j has the visible accepting future b^j .

Solution to Exercise 1.4. Map K_n to the exact real 2^{-n} . Successor is multiplication by $1/2$, but complete-match readout requires recognizing whether an arbitrary exact real coordinate is a negative power of two and recovering its exponent before comparing it with $|w|$. In a realistic real-number model this can be a poor or undefined operation. Requiring a uniform computable readout with a stated cost rules out the encoding as a complete realization.

Solution to Exercise 2.8. The residuals are

$$L = \{\varepsilon, a, ab\}, \quad a^{-1}L = \{\varepsilon, b\}, \quad (ab)^{-1}L = \{\varepsilon\}, \quad \emptyset.$$

From L , letter a reaches $\{\varepsilon, b\}$ and b reaches failure. From $\{\varepsilon, b\}$, b reaches $\{\varepsilon\}$ and a reaches failure. Every letter from $\{\varepsilon\}$ reaches failure, which loops. The first three states accept because each contains ε .

Solution to Exercise 2.9. The length-two factors of the square characteristic word include 11, 10, 00, 01, so Q_1 has four rows. Prefixes a^2 and a^5 both see 00 through horizon one. Through horizon two they see 001 and 000, respectively, because 4 is square and 7 is not.

Solution to Exercise 2.10. If membership depends only on length, then every entry in an (m, n) cut has the same total length $m + n$ and therefore the same truth value. Every prefix row is identical. Across different prefix lengths the total changes, so the one-row facts do not identify which future behaviors belong to one uniform state; square lengths give an infinite-residual counterexample.

Solution to Exercise 3.6. Choose a shortest access word $a_1 \cdots a_r$ for state s . The states reached after prefixes of lengths $0, 1, \dots, r$ are distinct: if two repeated, delete the intervening loop and obtain a shorter access word. There are $r + 1$ distinct states, so $r + 1 \leq k$. The automaton need not be minimal; only the chosen word is shortest.

Solution to Exercise 3.7. Take both component DFAs to be the two-state parity machine with the same start and transition. Only diagonal pairs (E, E) and (O, O) are reachable in the four-state product. If both accept E , their intersection is again the even-length language.

Solution to Exercise 4.5. The residual after a^i depends on $i \bmod 3$. There are three residuals, cycled by a , and the class $i \equiv 1$ accepts the empty future. Prefixes in different congruence classes are separated by a suffix making exactly one total congruent to one, so Myhill–Nerode gives the three-state lower bound.

Solution to Exercise 4.6. For distinct $w, x \in \{a, b\}^*$, use future cw^{rev} . The word wcw^{rev} is accepted. The word xcw^{rev} can have the required form only if $x = w$. Thus every two distinct pre-marker prefixes have different residuals.

Solution to Exercise 5.5. Before saturation, distances 0 through N to the singleton give $N + 1$ live rows, while more distant live prefixes and prefixes past T have the same reject-all bounded row, so the total is $N + 2$. At $N \geq T$, all $T + 1$ live singleton residuals plus failure are visible. If $N < T$, the accepting row first appears at prefix T . Once the horizon sees every live residual, failure first appears at prefix $T + 1$.

Solution to Exercise 5.6. Let $L = \{a\}$, $u = \varepsilon$, and $u' = aa$. Both prefixes reject at horizon zero, so $u \equiv_0 u'$. Their a -children are a and aaa ; the first accepts and the second rejects. At horizon one the parents were already distinguishable by future a , exactly as the map $Q_1 \rightarrow Q_0$ requires.

Solution to Exercise 6.12. Every subset of a discrete three-point space is clopen. Only $\emptyset$ and the whole set are clopen in the indiscrete space. In $[0, 1] \cup [2, 3]$ with the subspace topology, each component and its complement are open, so $\emptyset$, the whole space, and the two components are clopen.

Solution to Exercise 6.13. A neighborhood of any point omits only finitely many elements. An injective sequence visits each omitted element at most once and hence is eventually in the neighborhood. It converges to every point. In a Hausdorff space, distinct limits have disjoint neighborhoods, which a sequence cannot eventually occupy simultaneously.

Solution to Exercise 6.14. Given a finite coordinate prescription, copy those bits and set every other coordinate to zero. The resulting sequence is eventually zero and belongs to the requested cylinder. Hence every cylinder meets the set, so its closure is all of $2^{\mathbb{N}}$.

Solution to Exercise 7.7. The cylinder is the intersection of the three coordinate conditions $x_2 = b_2$, $x_5 = b_5$, $x_9 = b_9$, each clopen because the coordinate is discrete. Its complement is the union of the three cylinders requiring the corresponding coordinate to be $1 - b_i$.

Solution to Exercise 7.8. Every finite-coordinate cylinder has an eventually-zero extension, so the set is dense. A sequence with infinitely many ones is a limit of its finite truncations, all eventually zero, but is not itself eventually zero. Thus the set is not closed and its closure is the full product.

Solution to Exercise 7.9. The product contains all constant-tail encodings of subsets of $\mathbb{N}$, has arbitrarily large finite sets of distinct prefixes, and by diagonalization is uncountable. Compactness recovers finite-subcover and finite-intersection arguments. It does not permit enumeration, bound cardinality, or decide membership algorithmically.

Solution to Exercise 8.7. The levels are $\{0, 1\}$, $\{0, 1\}^2$, $\{0, 1\}^3$, and $\{0, 1\}^4$, with deletion of the last bit. The requested thread begins 1, 10, 101, 1010, 10100, ...

Solution to Exercise 8.8. For example choose $x_0 = 0$, $x_1 = 11$, and any longer words. Deleting the last bit of 11 gives 1, not 0, so the first bond already fails.

Solution to Exercise 8.9. At each horizon the tail row rejects every visible future, so each fixed word coordinate has value zero once it becomes visible. The associated language is empty. For any fixed b^j , membership in K_n is zero for all $n > j$, proving coordinatewise convergence to $\emptyset$.

Solution to Exercise 9.7. A returned suffix is exact semantic data; empty-suffix acceptance is an observation; a labeled call is presentation data; a shortest separator belongs to the semantic residual but measures observation depth; an integer counter is a coordinate; and a restriction map belongs to the observational system. The overlaps are legitimate when the same object mediates layers, but its role must be named.

Solution to Exercise 10.4. The induction step gives $AX_n \cup B = A(\bigcup_{j < n} A^j B) \cup B = \bigcup_{j < n+1} A^j B$. If every word of A has length at least one, a contribution from $A^j B$ has length at least j plus the minimum length of B . Terms with $j > N$ cannot affect words through length N , so $N + 1$ unfoldings are a safe general bound.

Solution to Exercise 10.5. On pairs of language behaviors define $F(X, Y) = (Y, aX)$. The first coordinate copies Y with zero delay, so F need not contract the maximum metric. But $F^2(X, Y) = (aX, aY)$ prefixes both coordinates and contracts by $1/2$.

Solution to Exercise 11.3. In the first alternative, the frame after A is bBc ; the frame after B is c . The material before a call has already executed when the call returns, so including it would run it twice and misrepresent continuation order.

Solution to Exercise 11.4. Distribute $A(B/C)$ into AB/AC . The single source occurrence of A is duplicated into two target occurrences while the positive relational language is preserved. Both target labels map to the one source label in the origin relation.

Solution to Exercise 12.4. At length zero the stack height is zero. An opener raises height; a closer at height zero fails. For each word, scan left to right, rejecting the first underflow. A live prefix of height h is completed by h closers. This rule classifies all fifteen words through length three without a separate case list.

Solution to Exercise 12.5. Use the canonical closer word for the first stack. It accepts there. At the first top-down type disagreement the other stack mismatches; if one stack ends first it underflows or remains nonempty. The closer length is the first stack's height, at most N .

Solution to Exercise 13.3. For $k = 1$, the counts through $N = 3$ are 2, 3, 4, 5. For $k = 2$ they are 2, 4, 8, 16; for $k = 3$, 2, 5, 14, 41. In each case the extra class merges failure with stacks too deep to close within the horizon.

Solution to Exercise 13.4. Choose infinite stacks 1111... and 1211.... Once the quotient sees the second stack position, their finite stack prefixes differ and canonical closers separate them. At rank one every stack word is determined solely by its height, leaving one infinite direction.

Solution to Exercise 14.2. With $r = 2$, the shortest crossing balanced word is $(())$, of length four; $()$ is the shortest nonempty balanced word that stays shallow. Horizon zero sees only acceptance, but no accepted threshold word occurs before length four, so the accepting row first appears there.

Solution to Exercise 14.3. Use live heights 0, 1, 2 and failure. An opener from height two fails; a closer from zero fails; otherwise open increments and close decrements. Only height zero accepts. The

farthest live state is reached in two steps, so accessibility degree is two. Heights 0, 1, 2 are separated by closer words of lengths 0, 1, 2, giving distinguishability degree two.

Solution to Exercise 15.4. For example use $(0, 0, 0)$, the three standard basis rows, and $(1, 1, 1)$. They are distinct and the basis rows give rank three. A deterministic table state must retain all five different future functions, including the zero row.

Solution to Exercise 15.5. A Boolean rank-one factor is a rectangle of ones. A rectangle containing two diagonal positions (i, i) and (j, j) also contains off-diagonal positions (i, j) and (j, i) , impossible in the identity. Thus each factor covers at most one diagonal one, requiring r factors; the r singleton rectangles give the matching upper bound.

Solution to Exercise 15.6. Use prefixes reaching heights 0, 1, 2 and canonical futures ε, c, cc . Their live submatrix is identity when rows and columns are paired by required closing length. Add an underflow prefix such as c ; its row is all zero and increases deterministic row count without increasing field rank.

Solution to Exercise 16.3. Index rows and columns by 00, 01, 10, 11. The matrix is the 4×4 identity. Every row is distinct, so Alice needs four possible messages, or two bits, and sending her input achieves the bound.

Solution to Exercise 16.4. Every concatenation in the cut has length $m + n$, so all entries share the truth value of “ $m + n$ is square.” Yet prefixes of lengths $p < q$ are separated by extending p to a square whose next gap exceeds $q - p$. The fixed cut is nonuniform in the lengths and does not provide one finite state space.

Solution to Exercise 16.5. Let prefixes u, u' have different rows and choose a suffix v on which they differ. If one deterministic exact streaming configuration followed both prefixes, continuing with the same v would produce the same output, a contradiction. Randomization could map both prefixes to overlapping distributions, and error would weaken the required separation.

Solution to Exercise 17.4. The chain is

$$C_{m,3}(u, abc) = C_{m+1,2}(ua, bc) = C_{m+2,1}(uab, c) = C_{m+3,0}(uabc, \varepsilon).$$

Literal labels identify ua , then uab , rather than merely asserting that some row of the next matrix has the same values.

Solution to Exercise 17.5. Require $C_{0,2}(\varepsilon, aa) = 1$ and transport it to $C_{1,1}(a, a) = 1$. Separately require transport from $C_{1,1}(a, a) = 0$ to $C_{2,0}(aa, \varepsilon) = 0$. Each isolated cut assignment is a legal Boolean table, but their shared overlap entry receives both values, so the partial family cannot glue.

Solution to Exercise 18.3. Assume $u \equiv_N u'$. Plateau bijectivity upgrades this to $u \equiv_{N+1} u'$. For every $|v| \leq N$, the word av has length at most $N + 1$, hence $uav \in L \iff u'av \in L$. Horizon N alone would control only futures of parent length at most N and might not include av .

Solution to Exercise 18.4. For each unordered pair of distinct minimal states choose a shortest distinguishing word. There are finitely many pairs, so the maximum separator length N is finite. At horizon N all k residuals are distinct, giving $q_L(N) = k$; no later quotient can exceed the exact residual count.

Solution to Exercise 19.25. Regularity gives a uniform access bound from shortest paths in the minimal DFA. Conversely, $a_L(N) \leq A$ implies $q_L(N) \leq \sum_{j=0}^A d^j$. A bounded nondecreasing integer sequence has an adjacent plateau; the global plateau theorem, not the finite prefix count alone, converts that equality into a right congruence and regularity.

Solution to Exercise 19.26. For $T = 6$, F_T accepts at lengths one and six, so both horizon-zero rows appear by radius one; parity also exposes accepted and rejected rows by radius one. Their intersection accepts only length six, so its accepting row first appears at radius six. Product bounds count possible tuples of rows but do not bound where a particular joint tuple first occurs.

Solution to Exercise 19.27. At rank one, $q(N) = N + 2$ and $a(N) = \max(1, N)$. At rank two, $q(N) = 2^{N+1}$ while access has the same linear law. Observation count detects typed branching exponentially; access records only the depth needed to reach each stack.

Solution to Exercise 19.28. Epsilon gives both diagonal distances zero. The one-letter neutral toggles the states, so $d(p, q) = d(q, p) = 1$ and

$$d = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

The empty-stack arrivals from the start are $\rho(\varepsilon, p) = 0$ and $\rho(\varepsilon, q) = 1$. At horizon zero the two semantic rows are rejection and acceptance, first represented at lengths zero and one, so $a_{L_M}(0) = 1$. If neither state accepted, both raw coordinates would have the same all-false semantic row. Its earliest representative is epsilon at length zero; maximizing raw arrivals first would incorrectly report one.

Solution to Exercise 19.29. Here

$$F_2 = \{c, occ, ococc\}.$$

From q_j , the rejected continuations are

$$\begin{aligned} q_0 &: \{c, occ, ococc\}, \\ q_1 &: \{c, occ\}, \\ q_2 &: \{c\}. \end{aligned}$$

Their different longest members distinguish the three states. All three forbidden words underflow at their last closer, so those distinctions disappear after intersection with D . The balanced words oc and $ococ$ first reach q_1 and q_2 in lengths two and four; $ococc$ reaches the accepting sink in length four. Thus the entry diameter is four. The one-state universal controller presents the same Dyck trace with diameter zero, so $\beta_D(D) = 0$.

Solution to Exercise 19.30. The empty continuation distinguishes epsilon from a nonempty word such as oc : epsilon is accepted, while oc is nonempty and never reaches height three. The same empty continuation distinguishes that below-threshold class from a hit word such as $oooccc$: the hit word is accepted. A nonempty balanced continuation such as $oooccc$ also illustrates why the middle class remembers a future obligation: appended after any below-threshold balanced history, it reaches the threshold.

Epsilon arrives at length zero. The shortest nonempty balanced word is oc , so the middle class arrives at length two. Reaching height three requires at least three openers and three matching closers; $oooccc$

realizes length six. Thus $\gamma_D = 6$, while oc already differs in current acceptance from epsilon and gives $\tau_D = 2$.

Solution to Exercise 19.31. At top level, epsilon is the undecided class. The word oc makes the first primitive component succeed and arrives at length two. The word $oocc$ is longer than oc but reaches only height two, so it fails the height-four test and supplies the permanently rejecting class at length four.

Immediately after the first opener, the empty factor arrives at zero. The factor oc is nonempty but reaches only relative height one, so it supplies the low class at length two. Reaching absolute height four from the current height one requires relative height three; $oooccc$ is the shortest balanced factor that does so and has length six. Appending the immediate closer c distinguishes the empty factor from oc : oc is the special accepted first component, whereas $oocc$ is a rejected below-threshold component.

Solution to Exercise 19.32. The legal entry sources are r_0 and r_1 . From r_0 , the four words end at r_0, r_2, r_0, r_1 , respectively. The first arrivals of the three possible endpoints are therefore 0, 2, 6. From r_1 , they end at r_1, r_2, r_0, r_1 , so all endpoints appear by length four. Maximizing the finite shortest distances over both entry sources gives six.

If a controller with fewer than five states had cost at most four, add unreachable self-loop states until the declared state set has size five. Neither its Dyck trace nor its legal-entry distances change. It would then satisfy the five-state constraint system, contradicting unsatisfiability. This padding argument depends on the encoding permitting unused states.

Solution to Exercise 20.2. A strong answer names the domain, proposed invariant or construction, a theorem-sized outcome, nearest comparator, required evidence, and a failure signal. Prefer directions whose negative result would also teach something. Reject choices justified only by fashionable vocabulary or by the existence of available machinery.

Notation and dependency guide

Symbol	Meaning and first substantial use
Σ	Finite input alphabet; Chapter 2.
Σ^*	Finite words, including ε .
$u^{-1}L$	Left residual $\{v : uv \in L\}$; Chapter 2.
$H_L(u, v)$	Infinite Boolean Hankel entry $\mathbf{1}[uv \in L]$.
$B_N(u)$	Future row of prefix u on all futures through length N .
$Q_N(L)$	Set of all horizon- N future rows.
$q_L(N)$	Cardinality of $Q_N(L)$.
$X[I, N]$	Rows represented by prefixes through length I at horizon N .
$G_L(I, N)$	Cardinality of $X[I, N]$.
$a_L(N)$	Least prefix radius representing every member of $Q_N(L)$.
B_M^{ent}	Largest shortest balanced-controller distance sourced at a legal segment entry.
$\beta_D(L)$	Least entry certificate over regular controllers with fixed Dyck trace $L \subseteq D$.
$\gamma_D(L)$	Access radius of semantic right residuals restricted to balanced histories and tests.
$\eta_D(L)$	Supremal balanced-factor residual access over all legal segment-entry histories.
$\nu_D(L; b)$	Minimum controller state count among fixed-Dyck-trace presentations with entry diameter at most b .
$\tau_D(L)$	Length of the first balanced word whose acceptance differs from epsilon.
O_L	Literal residual orbit $\{u^{-1}L : u \in \Sigma^*\}$.
X_L	Observational completion $\overline{O_L} \subseteq 2^{\Sigma^*}$.
K_n	Singleton continuation state $\{b^n\}$ in the running family.
D_k	Generalized Dyck language with k bracket types.
R_s	Dyck residual belonging to finite typed stack s .
$\perp$	Irreparable failure state or empty residual, by context.
$C_{m,n}$	Literal cut table on prefixes of length m and futures of length n .
$\rho_{n+1,n}$	Restriction or bonding map from fine to coarse observation.
$\varprojlim$	Inverse limit: compatible threads across all finite levels.

The core dependency route is

words and residuals $\longrightarrow$ finite automata $\longrightarrow$ finite observations
 $\longrightarrow$ topology and inverse limits $\longrightarrow$ presented dynamics
 $\longrightarrow$ Dyck control and Hankel methods
 $\longrightarrow$ sewing, plateaus, and growth.

How to read mathematical provenance

The book uses several kinds of mathematical statement. Their differences are part of the content.

Classical theorem. A result established in prior literature, such as Myhill–Nerode, compactness of countable finite products, or the Chomsky–Schützenberger representation. The exposition may give a new proof or emphasis without claiming the result.

Imported framework. Existing mathematics adapted to the recognition setting, such as partial-function algebra, relational least fixed points, weighted Hankel methods, or communication matrices.

FPRD synthesis. A recognition-specific assembly of ingredients with explicit presentation, observation, action, and readout interfaces.

Written FPRD result. A claim with a complete or informal written proof in the lab. It remains subject to ordinary mathematical review and is not called externally peer reviewed.

Candidate result. A formula or theorem under active proof audit. It may have independent finite evidence without yet carrying an unbounded proof at publication standard.

Finite experiment. A statement proved by exhaustive or sampled computation on an explicitly bounded domain. It can refute an unbounded conjecture but cannot establish one by extrapolation.

Research analogy. A structural comparison used to generate questions, such as the Robson globalization analogy. It transfers no theorem until a precise map and hypotheses are supplied.

Open question. A decision or mathematical problem whose answer is not represented as known.

The public claim ledger and executable evidence remain available in the FPRD repository. They are not reproduced as the narrative spine of this textbook because evidence status and mathematical explanation answer different reader needs.

Example 20.3 (One formula, several possible statuses). Checking an access formula through horizon two hundred establishes a finite experiment. A proof from consecutive-square gaps may establish the formula for every horizon. Finding the same formula in earlier literature transfers ownership. None of these facts follows from the other two.

Bibliography

- [1] Rajeev Alur and P. Madhusudan. Visibly pushdown languages. *Proceedings of STOC 2004*, pages 202–211, 2004.
- [2] Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.
- [3] Dean N. Arden. Delayed-logic and finite-state machines. In *Proceedings of the Second Annual Symposium on Switching Circuit Theory and Logical Design*, pages 133–151. IEEE, 1961.
- [4] Lisanne Bainbridge. Ironies of automation. *Automatica*, 19(6):775–779, 1983.
- [5] Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 3:133–181, 1922.
- [6] Janusz A. Brzozowski. Derivatives of regular expressions. *Journal of the ACM*, 11(4):481–494, 1964.
- [7] Rob Burns. The appearance function for paper-folding words, 2022.
- [8] Jack W. Carlyle and Azaria Paz. Realizations by stochastic finite automata. *Journal of Computer and System Sciences*, 5(1):26–40, 1971.
- [9] Noam Chomsky and Marcel-Paul Schützenberger. The algebraic theory of context-free languages. In *Computer Programming and Formal Systems*, volume 35 of *Studies in Logic and the Foundations of Mathematics*, pages 118–161. 1963.
- [10] Madeleine Clare Elish. Moral crumple zones: Cautionary tales in human-robot interaction. *Engaging Science, Technology, and Society*, 5:40–60, 2019.
- [11] Alina Firicel. Subword complexity and laurent series. *Integers*, 11B:A7, 2011.
- [12] Edward F. Moore. Gedanken-experiments on sequential machines. In Claude E. Shannon and John McCarthy, editors, *Automata Studies*, volume 34 of *Annals of Mathematics Studies*, pages 129–153. Princeton University Press, 1956.
- [13] Marston Morse and Gustav A. Hedlund. Symbolic dynamics. *American Journal of Mathematics*, 60(4):815–866, 1938.
- [14] Anil Nerode. Linear automaton transformations. *Proceedings of the American Mathematical Society*, 9(4):541–544, 1958.
- [15] Maurice Nivat and Jean-François Perrot. Une généralisation du monoïde bicyclique. *Comptes rendus de l’Académie des sciences de Paris, Série A*, 271:824–827, 1970.

- [16] Raja Parasuraman and Dietrich H. Manzey. Complacency and bias in human use of automation: An attentional integration. *Human Factors*, 52(3):381–410, 2010.
- [17] Henry L. Roediger and Jeffrey D. Karpicke. Test-enhanced learning: Taking memory tests improves long-term retention. *Psychological Science*, 17(3):249–255, 2006.
- [18] Leonid Rozenblit and Frank Keil. The misunderstood limits of folk science: An illusion of explanatory depth. *Cognitive Science*, 26(5):521–562, 2002.
- [19] Filippo Santoni de Sio and Jeroen van den Hoven. Meaningful human control over autonomous systems: A philosophical account. *Frontiers in Robotics and AI*, 5:15, 2018.
- [20] Benjamin Steinberg. Topological dynamics and recognition of languages, 2013.
- [21] Andrew Chi-Chih Yao. Some complexity questions related to distributive computing. *Proceedings of STOC 1979*, pages 209–213, 1979.

Index

- access radius, 15, 68
- action, 34
- bonding map, 28
- boundary behavior, 47
- bounded future row, 5
- Cantor space, 23
- Chomsky–Schutzenberger theorem, 44
- claim boundary, 92
- closed set, 19
- communication complexity, 57
- compactness, 19
- compatible field, 61
- context-free language, 44
- continuation
 - infinite family, 2
- controller, 50
- cylinder set, 23
- depth threshold, 50
- deterministic finite automaton, 9
- Dyck language, 44
 - residuals, 47
- finite control, 50
- FPRD
 - four layers, 34
 - research frontier, 92
- generated action, 40
- guarding, 37
- Hankel table, 5
 - rank, 54
- horizon equivalence, 15
- information complexity, 57
- inverse limit, 28
- language, 5
- least fixed point, 37
- linear representation, 54
- local-to-global, 61
- Myhill–Nerode theorem, 12
- Nerode equivalence, 12
- normalization, 40
- observation, 34
- observation growth, 68
- open problem, 92
- open set, 19
- orbit closure, 23
- partial function, 37
- plateau
 - global adjacent, 15, 65
- presentation, 34
- presentation transformation, 40
- product topology, 23
- reachability, 9
- readout, 34
- recognition problem, 2
- recursion, 37
- regularity, 65
- residual automaton, 12
- residual language, 5
- right congruence, 65
- semiring, 54
- sewing theorem, 61
- stack
 - typed, 47
- state future, 9
- streaming complexity, 57
- topological space, 19
- uniformization cost, 68
- universal property, 28