

The Hierarchy of Languages Is Set by the Type of Memory, Not Its Amount

— One Unbounded Stack Beats 10^{80} States of Finite Memory — Each Step Up Moves the Cost from n to n^3 to Exponential to Impossible — [Paper 296]

Yuuki YAMAGISHI (山岸 勇輝)

Yamagishi Complexity Torus Lab, Japan

August 2026

— *From thinking to syncing.* —

Abstract

The Chomsky hierarchy divides languages into four levels. Asked what divides them, one usually answers “complexity.” This paper asks what that complexity actually is—the answer is the type of memory of the recognising machine. No new mathematical theorem and no new law is claimed. First, we line up the four levels by type of memory. None / one stack / a tape proportional to the input / an unbounded tape (Section 2). Second, this is the core of the paper. A finite automaton with 10^{80} states still cannot recognise $a^n b^n$, while one stack can (Section 3). Third, the equivalence classes are countably infinite. a^i and a^j are told apart by appending b^i , so finitely many states cannot suffice in principle (Section 3). Fourth, each step up makes the cost jump. At $n = 1000$ the gap between $O(n)$ and $O(n^3)$ is 10^6 , and 2^n is 10^{301} (Section 4). Fifth, at the top level the cost is not defined. Membership for phrase-structure languages is undecidable (Section 4). Sixth, the separator is whether brackets can be matched. Fix the depth and finiteness suffices; arbitrary depth demands the infinite (Section 5).

Scope of this paper (scope note): No new mathematical theorem and no new law is claimed—the Chomsky hierarchy, the Myhill–Nerode theorem, the pumping lemma, $O(n^3)$ parsing, and undecidability are all standard. We do not build formal language theory—all we use is one pigeonhole argument and three comparisons of cost. We do not prove the theorems—Myhill–Nerode and the pumping lemma are merely quoted, and we do not enter their proofs. We do not discuss natural language—where human language sits in the hierarchy is not treated at all. Only the hierarchy of formal languages is treated. We do not discuss computational complexity—we do not enter P versus NP nor the details of PSPACE. We only line up the orders of cost by level. We do not say $O(n^3)$ is best—context-free parsing is known to reduce to the exponent of matrix multiplication. n^3 is a classical upper bound. We do not say four levels is the only division—deterministic context-free, linear languages and many intermediate families exist. This paper treats only the coarsest four. We do not confuse this with the metaphorical “grammar”—Paper 87 called the action principle “the grammar of physical law” and Paper 92 called differential forms “a shared alphabet.” Both are metaphors. This paper treats formal grammar literally, so the word is the same and the object is not. Relation to earlier papers: Paper 179 showed that “cannot be constructed” has distinct roots, touching Turing’s halting problem—the undecidability of the top level here is one of those roots, placed here as the apex of a hierarchy. Paper 163 separated “a good code exists” from “here is a good code”—this paper likewise separates “can be recognised” from “can be recognised at what cost”. Paper 112 counted “six distinct roots sharing one rhyme”—the four levels here are four different things under the one word “complexity”. Paper 100 set down “one word, distinct roots” as a discipline of the corpus—this paper applies that discipline to formal languages. What is added is re-lining the four levels by type of memory, showing numerically that 10^{80} states still fail to reach $a^n b^n$, computing the cost ratio at $n = 1000$ as 10^6 , and placing the separator at bracket matching.

1. Notation and the Question

Setting: Formal languages divide into four levels by the machine that recognises them (the Chomsky hierarchy).

Question: What divides the levels? “Complexity” is an answer, but complexity of what?

Observation 1.1

This paper’s answer is the type of memory. Not speed, not the number of rules, not the number of states—the shape of what can be held.

Observation 1.2

A note before we start—the corpus has used “grammar” as a metaphor. Paper 87 called the action principle “the grammar of physical law”; Paper 92 called differential forms “a shared alphabet”—this paper differs from both and treats formal grammar literally. The same word, a different object.

2. Lining Up the Four Levels by Type of Memory

Check 2.1:

Family	Machine	Memory	Cost of membership
Regular	finite automaton	none	$O(n)$
Context-free	pushdown	one stack	$O(n^3)$
Context-sensitive	linear bounded automaton	tape $\propto$ input	PSPACE
Phrase structure	Turing machine	unbounded tape	undecidable

Observation 2.2

The second column (the machine’s name) is merely a definition. What acts is the third—it is not the machines that differ but the memory.

Observation 2.3

Reading the third column downwards: none $\rightarrow$ one $\rightarrow$ bounded $\rightarrow$ unbounded. What grows is type, not amount—the gap between “none” and “one stack” cannot be closed with quantity (Section 3).

Observation 2.4

In the rightmost column, only the last entry is not a number. After $O(n)$, $O(n^3)$ and PSPACE comes undecidable—the cost is not large; the quantity “cost” is not defined.

3. The Core: One Unbounded Stack Beats 10^{80} States

Setting: Consider the language $a^n b^n$ (n copies of a followed by n copies of b).

Check 3.1: The Myhill–Nerode distinction. For $i \neq j$, a^i and a^j are told apart by appending b^i .

String	In the language?
$a^i b^i$	yes
$a^j b^i$ ($j \neq i$)	no

Observation 3.2

So $a^0, a^1, a^2, \dots$ must all be distinguished from one another. The equivalence classes are countably infinite — finitely many states cannot suffice in principle.

Check 3.3: We try adding states.

Number of states k	a^n it can distinguish	Input on which it must fail
2	a^0 to a^1	$a^2 b^2$
10	a^0 to a^9	$a^{10} b^{10}$
10^6	a^0 to a^{10^6-1}	$a^{10^6} b^{10^6}$
10^{80}	a^0 to $a^{10^{80}-1}$	$a^{10^{80}} b^{10^{80}}$

Observation 3.4

This is the core of the paper. Give it as many states as there are atoms in the observable universe and it still breaks just beyond — while a pushdown automaton recognises $a^n b^n$ with one stack.

Observation 3.5

10^{80} states of finite memory < one unbounded stack — the shortest way to say that type acts, not amount.

Observation 3.6

What the stack does is count. Push on each a , pop on each b —a finite automaton does not lack the operation of counting; it lacks counting without a ceiling.

Check 3.7: Then is one stack enough? Consider $a^n b^n c^n$.

Language	Machine that recognises it
$a^n b^n$	pushdown (one stack)
$a^n b^n c^n$	not a pushdown (needs a linear bounded automaton)

Observation 3.8

The stack empties while matching a against b , leaving nothing to match c with. Each added step of memory strictly enlarges what can be recognised—no level can be skipped.

4. Each Step Up Makes the Cost Jump

Check 4.1: We line up the cost against the input length n .

n	n (regular)	n^3 (context-free)	2^n (exponential)
10	10	1000	1.024×10^3
100	100	10^6	1.268×10^{30}
1000	1000	10^9	10^{301}
10000	10^4	10^{12}	10^{3010}

Observation 4.2

At $n = 1000$, context-free costs 10^6 times what regular costs. The same length of text, read at a million times the effort—the price of becoming able to count brackets.

Observation 4.3

$2^{1000} = 10^{301}$ is 10^{221} times the number of atoms in the observable universe (10^{80}). Possible in principle, unperformable in fact——“can” and “can at a feasible cost” are different (the same figure as Paper 163’s distinction).

Observation 4.4

A point to write honestly—— $O(n^3)$ is a classical upper bound. Context-free parsing reduces to the exponent of matrix multiplication—— n^3 is a number showing that a level was crossed, not the best value.

Observation 4.5

At the top level no row can be written in this table. Membership for phrase-structure languages is undecidable, and no cost is defined for any n ——one of the roots of “cannot be constructed” treated in Paper 179 appears here as the apex of a hierarchy.

5. Separator: Whether Brackets Can Be Matched

Observation 5.1

Separator——what must be moved for regular and context-free to exchange? Whether the bracket depth has a ceiling.

Check 5.2:

Brackets handled	States required	Level
Depth up to 1	2	regular
Depth up to 5	6	regular
Depth up to 100	101	regular
Arbitrary depth	infinitely many	context-free

Observation 5.3

Every row but the last is regular.Brackets to depth 100 can be written as a regular expression—the boundary lies not at “deep” but at “no ceiling”.

Observation 5.4

This bites in practice.A regular expression cannot parse HTML because tags nest to arbitrary depth, and limiting the depth to 100 makes it writable—what forbids it is not complexity but the absence of a ceiling.

Observation 5.5

Paper 112 counted “six distinct roots sharing one rhyme.”Under the one word “a complex language” stand four different things—divided not by a degree of complexity but by the qualitative difference of memory type.

Observation 5.6

Paper 100 set down the corpus’s discipline as “one word, distinct roots.”This paper applied that discipline to formal languages—to grammar taken literally, not as metaphor.

6. Roll Call and Closing

1. Four levels by memory type (Check 2.1): **None / one / bounded / unbounded.**
2. **Only the last is not a number** (Observation 2.4): **Undecidable.**
3. Countably infinite classes (Check 3.1): $a^i b^i$ against $a^j b^i$.
4. 10^{80} states still break (Check 3.3): At $a^{10^{80}} b^{10^{80}}$.
5. 10^{80} finite < one unbounded (Observation 3.5): **Type acts.**
6. $a^n b^n c^n$ defeats one stack (Check 3.7): **No level can be skipped.**
7. 10^6 at $n = 1000$ (Check 4.1): **The price of counting brackets.**
8. $2^{1000} = 10^{301}$ (Observation 4.3): **10^{221} times the atoms.**
9. **Depth 100 is regular** (Check 5.2): **101 states.**

10. The boundary is the missing ceiling (Observation 5.4): Not the depth.

Closing: The hierarchy of languages is set by the type of memory, not its amount. The four levels line up as none / one stack / a tape proportional to the input / an unbounded tape, and it is the shape of the memory, not the name of the machine, that divides them. The shortest statement is this—a finite automaton with 10^{80} states, as many as there are atoms in the observable universe, still cannot recognise $a^n b^n$, while a pushdown automaton recognises it with **one stack. Because a^i and a^j are told apart by appending b^i , the equivalence classes are countably infinite. And each step up makes the cost jump—at $n = 1000$ the gap between $O(n)$ and $O(n^3)$ is 10^6 , 2^n is 10^{301} , and at the top level the quantity “cost” is not defined at all. One thing separates them—whether the bracket depth has a ceiling. Up to depth 100, a finite automaton of 101 states suffices. What forbids it is not depth but the absence of a ceiling.**

References

1. Chomsky, N. (1956). Three Models for the Description of Language. *IRE Transactions on Information Theory*, 2(3).
2. Myhill, J. (1957). *Finite Automata and the Representation of Events*. WADD Technical Report 57-624.
3. Nerode, A. (1958). Linear Automaton Transformations. *Proceedings of the American Mathematical Society*, 9(4).
4. Bar-Hillel, Y., Perles, M. and Shamir, E. (1961). On Formal Properties of Simple Phrase Structure Grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*, 14.
5. Hopcroft, J. E., Motwani, R. and Ullman, J. D. (2006). *Introduction to Automata Theory, Languages, and Computation*. Boston: Addison-Wesley.
6. Valiant, L. G. (1975). General Context-Free Recognition in Less Than Cubic Time. *Journal of Computer and System Sciences*, 10(2).
7. Yamagishi, Y. (2026). *The Action Principle as the Description Language of Physical Law* [Paper 87]. Zenodo. 10.5281/zenodo.21747287
8. Yamagishi, Y. (2026). *The Crossing of Royal Roads* [Paper 100]. Zenodo. 10.5281/zenodo.21758864
9. Yamagishi, Y. (2026). “*A Good Code Exists*” and “*Here Is a Good Code*” Are Different Sentences [Paper 163]. Zenodo. 10.5281/zenodo.22092677
10. Yamagishi, Y. (2026). “*Cannot Be Constructed*” Has Distinct Roots [Paper 179]. Zenodo. 10.5281/zenodo.22098419

— *From thinking to syncing.* —

Yuuki YAMAGISHI (山岸 勇輝)

© 2026 Yamagishi Complexity Torus Lab, Japan · Licensed under Creative Commons Attribution 4.0 International (CC BY 4.0)