EXACT VERSUS UNIQUE NONDETERMINISTIC AUTOMATIC COMPLEXITY: THE MISSING LENGTH-16 AUTOMATON, EXPLICIT WITNESSES AT LENGTHS 15 AND 16, AND COMPLETE CENSUSES AT LENGTHS 14 AND 15

KOREA SUPERINTELLIGENCE LABS

ABSTRACT. For a word x over a finite alphabet, $A_{Ne}(x)$ is the least number of states of a nondeterministic finite automaton that accepts x and no other word of length $|x|$, and $A_N(x)$ is the least number of states of one that in addition accepts x along a single computation; always $A_{Ne}(x) \leq A_N(x)$. Kjos-Hanssen and Rivera Petit have recently shown that the inequality can be strict, that 13 is the least length at which it can be, and that up to renaming of letters the separating words of length 13 are exactly the 365 letter-mergings of `0123444445126`. They also print a table of the pairs (A_{Ne}, A_N) at lengths 13 through 17, but leave the rows for lengths 15 and 16 without a witness of any kind: their length-15 automaton diagram carries no letters on any arrow, and there is no length-16 diagram at all. (At lengths 13, 14 and 17 a word is printed, or recoverable from a labelled diagram.)

We supply the missing witnesses and certify them. The word `0123444444445126` $=$ `0123` 4^8 `5126` — their length-13 witness with its run of fours lengthened from five to eight — satisfies $A_{Ne} = 8 < 9 = A_N$, with an explicit eight-state exactly accepting automaton and an explicit nine-state uniquely accepting one; deleting its last letter leaves a length-15 word with $A_{Ne} = 7 < 8 = A_N$. Two further separating words of length 16 and the reversal of the length-15 one are certified in the same way. We also carry the census two lengths further: at length 14 exactly 20 words separate the two measures, all letter-mergings of `01233333333401`, all with $(A_{Ne}, A_N) = (6, 7)$, and each of the twenty is certified individually; at length 15 exactly 74 do, forming 37 reversal pairs, all with $(A_{Ne}, A_N) = (7, 8)$, and none of them binary, although lengths 13 and 14 both carry binary witnesses. The Hyde–Kjos-Hanssen bound $A_N(x) \leq \lfloor n/2 \rfloor + 1$ is verified for all 190,899,322 canonical words of length 14, one length beyond the published verification. Every value stated below is machine-checked in Lean 4 except where it is explicitly labelled otherwise: the two censuses, the bound verification and a handful of auxiliary counts are computations outside the formal development.

1. INTRODUCTION

The two measures. Automatic complexity, introduced by Shallit and Wang [11], measures a finite word by the size of the smallest automaton that singles it out among all words of its length. Hyde and Kjos-Hanssen [4, 3] carried the definition to nondeterministic automata by asking that the automaton accept the given word along a *unique* computation path; the book [7] surveys the area.

For nondeterministic automata, uniqueness of the accepting path is not the only way to formalise "singles out". Chen, Kjos-Hanssen, Koswara, Richter and Stephan [2] studied the variant in which the automaton must accept the target word and reject every other word of the same length, while remaining free to accept the target along several paths. Writing A_{Ne} for the resulting measure and A_N for the Hyde–Kjos-Hanssen one, a uniquely accepting automaton is in particular exactly accepting, so

$$A_{Ne}(x) \ \leq \ A_N(x) \qquad \text{for every word } x.$$

Whether the inequality can be strict was their closing question. Their notation for the two measures is the one used here, and Question 48 of [2, p. 24:16] reads

Let $\Sigma = \{0, 1\}$. Does there exist $x \in \Sigma^*$ for which $A_{Ne}(x) < A_N(x)$?

(quoted from the published version; the arXiv version of the same paper opens the question with "Let $|\Sigma| = 2$" instead). The question is older than that paper: [6] records it as open and traces it to [5].

The source, and the three gaps in its table. Kjos-Hanssen and Rivera Petit [8] answer the question. Their Theorem 1.1 is that the binary word `1101000000100` has $A_{Ne} = 6 < 7 = A_N$, and their Theorem 1.2 is an exhaustive census: no word of length at most 12 over any alphabet separates the two measures, and up to renaming of letters the separating words of length 13 are precisely the 365 letter-mergings — out of the $B_7 = 877$ possible ones — of the seven-letter word

$$x^* \;=\; \texttt{0123444445126} \;=\; \texttt{0123}\,\texttt{4}^5\,\texttt{5126},$$

each with $A_{Ne} = 6$ and $A_N = 7$. Alongside these they print a five-row table (their Figure 2) of the pairs (A_{Ne}, A_N) together with the Hyde–Kjos-Hanssen upper bound:

n	A_{Ne}	A_N	$\lfloor n/2 \rfloor + 1$
13	6	7	7
14	6	7	8
15	7	8	8
16	8	9	9
17	8	9	9

The rows for $n = 15$, 16 and 17 carry no word. Their Figure 1 prints automaton diagrams for $n = 13$, 14, 15 and 17, and of these the $n = 15$ diagram carries *no letters on any arrow* and marks no state initial or final, so no word — indeed no language — can be read off it; there is *no* $n = 16$ diagram; and the text offers no derivation for the last three rows beyond the sentence, in the caption of the table, that "the witnessing NFA simply gets a head and/or tail segment appended". The $n = 14$ and $n = 17$ diagrams are fully labelled and a word can be decoded from each.

Results. Everything below concerns words over an arbitrary finite alphabet, in the canonical form in which letters are introduced in the order $0, 1, 2, \ldots$; both measures depend only on the partition of positions into equal letters [8, §6], so this loses nothing.

- *The missing length-16 witness* (Theorem 4.1). The word `0123444444445126` $=$ $\texttt{0123}\,\texttt{4}^8\,\texttt{5126}$, which is the source's x^* with its run of fours lengthened from five to eight, satisfies $A_{Ne} = 8$ and $A_N = 9$. Both bounds are certified: the upper ones by the explicit automata of Section 4, the lower ones by exhausted searches. By Proposition 3.5 the lower bounds hold over every alphabet, not only over the seven letters the word uses (Corollary 4.2). Two further length-16 witnesses, forming a reversal pair, are certified in Theorem 4.4.
- *A length-15 witness* (Theorem 4.3). Deleting the last letter of the word above leaves `012344444444512`, with $A_{Ne} = 7$ and $A_N = 8$. Its reversal, whose canonical form is `012333333334015`, has the same two values.
- *Every separating word of length 14* (Theorem 5.1). Twenty canonical words of length 14 are listed, and each is certified to have $A_{Ne} = 6$ and $A_N = 7$. That these twenty are *all* of them is Remark 5.2, a computation outside the formal development.

The two pairs of values agree with the source's printed table. What is supplied here is the missing data — the words, their automata — and a machine-checked proof of the values, which for a lower bound means an exhausted search together with a checked proof that exhausting it settles the statement about all automata of that size.

Three further findings are computations run outside the formal development, and are stated as such:

- *The census at length 14* (Remark 5.2): exactly 20 canonical words of length 14 over all alphabets separate the two measures. They are 20 of the $B_5 = 52$ letter-mergings of `01233333333401`, all have $(A_{Ne}, A_N) = (6, 7)$, and exactly two of them are binary. The source's census stops at length 13.

- *The census at length 15* (Remark 6.1): exactly 74, forming 37 reversal pairs, all with $(A_{Ne}, A_N) = (7, 8)$. They split into the 37 letter-mergings of `012344444444512` and the 37 of its reversal, and *none* of them is binary — where lengths 13 and 14 each carry binary witnesses.
- *The Hyde–Kjos-Hanssen bound at length 14* (Remark 5.3): every one of the B_{14} = 190,899,322 canonical words of length 14, over every alphabet, satisfies $A_N(x) \le \lfloor 14/2 \rfloor + 1 = 8$. The source verifies the bound rather than assuming it, but only for lengths $n \le 13$.

Section 7 reproduces and certifies the published data — the source's Theorem 1.1, its Proposition 3.1, the length-14 and length-17 words decodable from its diagrams, and its length-13 census — and Section 8 collects the negative controls. Section 9 lists what is left, with the cost of the two computations we did not run.

What is not claimed. We do not prove the completeness half of either census inside the proof assistant: that exactly twenty words of length 14 and exactly 74 of length 15 separate the two measures rests on the aggregate computations of Remarks 5.2 and 6.1, which were run in C. What is certified is that each listed word does separate. We do not extend the census to length 16; we do not re-verify the Hyde–Kjos-Hanssen bound at length 15, and the length-15 census therefore takes its state cap from that published bound instead of re-deriving it, as was done at every length up to 14. We prove nothing about any infinite family of separating words; Remark 8.3 shows that the obvious one-parameter family does not separate at every parameter. Finally, we claim no priority over the values in the table above, which are the source's; only the witnesses realising rows 15 and 16, the two censuses, and the machine-checked proofs are ours.

Provenance. The notions themselves have already been formalised in Lean 4 by the first author of [8]: the repository [1] defines both measures and proves both $A_{Ne} \le A_N$ and the Hyde–Kjos-Hanssen bound $A_N(x) \le \lfloor n/2 \rfloor + 1$. What that repository does not contain, as of its last commit before this work, is any separating word, any folding lemma, any exhaustive search, or any computed value of either measure. So the definitions were formalised before this work and the values were not.

Beyond [8] we could find no source printing a separating word of length 15 or 16, and none carrying a census beyond length 13. A full-text search of a large snapshot of arXiv sources for the literal A_{Ne} returns exactly three papers — [8], [2] and [6] — all by the same group; the bibliographic database OpenAlex, queried eleven days after [8] was submitted, had no record of it at all, so no citer list for it exists there; and the counts $0, \dots, 0, 365, 20, 74$ match no sequence we could find in the OEIS. Read these as "not found", not as "new". We note the specific risk: [8] is recent and its own Section 7 describes a fast, agent-assisted workflow, so a revision filling in the length-15 and length-16 words would turn Theorems 4.1 and 4.3 into re-derivations. Their certification would stand either way.

2. Preliminaries

We follow [8] throughout. Fix a finite alphabet Σ. A nondeterministic finite automaton (NFA) is $M = (Q, \Sigma, \delta, q_0, F)$ with Q finite, $\delta \colon Q \times \Sigma \to \mathcal{P}(Q)$, initial state $q_0 \in Q$ and final states $F \subseteq Q$; ε-transitions are not allowed. An *accepting path* (or accepting computation) for $x = x_1 \cdots x_n$ is a sequence of states $p_0 = q_0, p_1, \dots, p_n$ with $p_i \in \delta(p_{i-1}, x_i)$ for each i and $p_n \in F$.

We quote the two definitions verbatim from [8, §2]:

> Following [2] we say that M *exactly accepts* x if M accepts x and rejects every word $y \ne x$ with $|y| = |x|$, and that M *uniquely accepts* x if M exactly accepts x and there is only one accepting path for x.
>
> $A_{Ne}(x) = \min\{k : \text{some } k\text{-state NFA exactly accepts } x\}$, $\quad A_N(x) = \min\{k : \text{some } k\text{-state NFA uniquely accepts } x\}$.

What is counted is states, not transitions, and the alphabet is arbitrary. Since a uniquely accepting automaton is exactly accepting, $A_{Ne}(x) \le A_N(x)$.

Both measures are invariant under injective renaming of letters, and more: they depend only on the partition of the positions $1, \dots, n$ induced by equality of letters [8, §6]. It therefore suffices to consider words in *canonical* form, where the letters are $0, 1, 2, \dots$ introduced in that order — the restricted growth strings, of which there are B_n (the nth Bell number) at length n. All words displayed below are canonical. Both measures are also invariant under reversal [8, §6], so the set of separating words of a given length is closed under reversal followed by recanonicalisation. We write $\mathtt{4}^m$ for a run of m copies of the letter $\mathtt{4}$, so that $x^* = \mathtt{0123}\,\mathtt{4}^5\,\mathtt{5126}$.

Finally we use, in one place and with attribution, the bound of Hyde and Kjos-Hanssen [4]: $A_N(x) \leq \lfloor n/2 \rfloor + 1$ for every word of length n.

3. From all automata to state sequences

A lower bound $A_{Ne}(x) \geq k + 1$ is a statement about every k-state automaton, of which there are $2^{k^2|\Sigma|}$ once the initial and final states are fixed. The reduction that makes the problem finite and small is the source's, and we restate it before recording what the formal proofs add to it.

Lemma 3.1 (Folding; [8, Lemma 4.1])**.** *Let M be an NFA with state set Q that exactly accepts x, $|x| = n$, and let $p = (p_0, \dots, p_n)$ be an accepting path for x. Let M_p be the automaton with state set Q whose transitions are exactly the triples (p_{i-1}, x_i, p_i) for $1 \leq i \leq n$, with initial state p_0 and sole final state p_n. Then M_p exactly accepts x, and if M uniquely accepts x then M_p uniquely accepts x.*

Consequently the search for a lower bound may range over *state sequences* $p = (p_0, \dots, p_n)$ rather than over transition relations. Whether the induced automaton exactly (resp. uniquely) accepts x is a property of p and x alone; we write $\mathrm{Ex}(p, x)$ and $\mathrm{Un}(p, x)$ for it. The formal version of Lemma 3.1 is stated in exactly this form: from a k-state automaton exactly accepting x it produces a sequence p of length $n+1$ with all values below k and $\mathrm{Ex}(p, x)$. Like [8, Lemma 4.1] it quantifies over automata with an arbitrary set of final states, so the lower bounds it yields are statements about all automata of a given size, not only about those with a single final state.

Lemma 3.2 (Normalisation)**.** *Let $\rho(p)$ be the sequence obtained from p by replacing each value by its rank in the order of first appearance along p. Then $\rho(p)$ is a restricted growth sequence starting at 0; if every value of p is less than k then so is every value of $\rho(p)$; and $\mathrm{Ex}(\rho(p), x) \leftrightarrow \mathrm{Ex}(p, x)$, $\mathrm{Un}(\rho(p), x) \leftrightarrow \mathrm{Un}(p, x)$.*

This is one sentence of [8, §4] — "after discarding unused states and renaming states in order of first appearance along p, we may assume $p_0 = 0$ and that (p_i) is a restricted growth sequence" — and is what removes the $k!$ relabellings of each sequence. Its effect is large: see the node counts in Section 10.

The search itself, written $\mathrm{search}(k, \star, x)$ with $\star \in \{\mathrm{exact}, \mathrm{unique}\}$, is a depth-first enumeration of restricted growth sequences of length $n + 1$ with values below k, pruned as follows. Along a prefix P of the sequence, with the transitions that prefix has already produced, track three sets of states: those reachable while spelling the corresponding prefix of x; those reachable while spelling something else ("bad"); and those reachable along a computation that has already diverged from P ("diff"). If the next value of P lies in the bad set, no extension of P can satisfy Ex; if it lies in the diff set, none can satisfy Un. Both sets grow monotonically with the transition set, so the test is sound at every prefix. The prune is the exact-acceptance analogue of the two-walk prune of [8, §4].

Proposition 3.3 (Soundness of the search)**.** *If* $\mathrm{search}(k, \mathit{exact}, x)$ *returns* false *then no NFA with at most k states, over the alphabet of x, exactly accepts x; hence $A_{Ne}(x) > k$. Likewise if* $\mathrm{search}(k, \mathit{unique}, x)$ *returns* false *then $A_N(x) > k$.*

The proof combines Lemmas 3.1 and 3.2 with the monotonicity of the prune; it is a statement about a program, and neither it nor the two lemmas is new mathematics. It is what makes the lower bounds below proofs rather than reports of a computation.

For the upper bounds the difficulty is the opposite one: *exactly accepts* quantifies over all $|\Sigma|^n$ words of length n, which is $7^{16} \approx 3.3 \cdot 10^{13}$ for the length-16 witness. The quantifier is replaced by a subset construction of size $n \cdot |Q| \cdot |\Sigma|$.

Proposition 3.4 (Certificates)**.** *Let M be an NFA and x a word of length n. Let $R(x)$ be the set of states reachable from q_0 by reading some word of length n other than x, computed by the obvious induction on n alongside the set reachable by reading x itself. If $R(x) \cap F = \emptyset$ and M has an accepting path for x, then M exactly accepts x. Let moreover $D(x,p)$ be the set of states reachable, from q_0 by reading x, along a computation differing from a given accepting path p in at least one position. If also $D(x,p) \cap F = \emptyset$ then M uniquely accepts x.*

Only the completeness direction of the two constructions is needed — an offending word puts its endpoint into $R(x)$, an offending computation puts its endpoint into $D(x,p)$ — and that is what is proved. The construction is standard; it is recorded because every upper bound below rests on it.

Both measures are stated above over the alphabet of the word. The following makes the lower bounds independent of the ambient alphabet, which is the setting of the source's census.

Proposition 3.5 (Alphabet restriction)**.** *Let $\Sigma \subseteq \Sigma'$ and let M be an NFA over Σ'. Let $M \restriction \Sigma$ be M with its transitions on letters outside Σ deleted. If M exactly (resp. uniquely) accepts a word $x \in \Sigma^*$, then so does $M \restriction \Sigma$. Consequently, if no k-state automaton over Σ exactly (resp. uniquely) accepts x, then no k-state automaton over any larger alphabet does either.*

The proof is immediate: restricting keeps every computation on words over Σ and can only shrink the set of words that must be rejected. We record it because it is what matches the values below to the source's setting, which is explicitly "over any alphabet".

4. Separating words at lengths 15 and 16

Theorem 4.1. *Let $w = \mathit{0123444444445126} = \mathit{01234}^{8}\,\mathit{5126}$, a word of length 16 over the alphabet $\{0,\dots,6\}$. Then*

$$A_{Ne}(w) = 8 \;<\; 9 = A_N(w).$$

The upper bounds are witnessed by two explicit automata. Let E have states $q_0,\dots,q_7$, initial state q_0, sole final state q_7, and the ten transitions

$$q_0 \xrightarrow{0} q_1,\quad q_1 \xrightarrow{1} q_2,\quad q_2 \xrightarrow{2} q_3,\quad q_3 \xrightarrow{3} q_4,\quad q_4 \xrightarrow{4} q_5,$$
$$q_5 \xrightarrow{4} q_6,\quad q_6 \xrightarrow{4} q_4,\quad q_6 \xrightarrow{4} q_6,\quad q_5 \xrightarrow{5} q_1,\quad q_3 \xrightarrow{6} q_7,$$

and let U have states $q_0,\dots,q_8$, initial state q_0, sole final state q_8, and the nine transitions

$$q_0 \xrightarrow{0} q_1,\quad q_1 \xrightarrow{1} q_2,\quad q_2 \xrightarrow{2} q_3,\quad q_3 \xrightarrow{3} q_4,\quad q_4 \xrightarrow{4} q_4,$$
$$q_4 \xrightarrow{5} q_5,\quad q_5 \xrightarrow{1} q_6,\quad q_6 \xrightarrow{2} q_7,\quad q_7 \xrightarrow{6} q_8.$$

Both are the automata induced, in the sense of Lemma 3.1, by an accepting path: E by

$$0, 1, 2, 3, 4, 5, 6, 4, 5, 6, 6, 4, 5, 1, 2, 3, 7$$

and U by $0, 1, 2, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 6, 7, 8$. In E the word is accepted along more than one path — necessarily so, since by the theorem no eight-state automaton accepts it uniquely — which is exactly why E witnesses only the smaller measure: the run of eight fours can be spelled by more than one walk through q_4, q_5, q_6.

Proof of Theorem 4.1. For $A_{Ne}(w) \le 8$: the path above is an accepting path of E for w, and the subset construction of Proposition 3.4 reaches no final state of E by any other word of length 16, so E exactly accepts w. For $A_N(w) \le 9$: the second path is an accepting path of U for w, and both constructions of Proposition 3.4 avoid the final state of U, so U uniquely accepts w. For the lower bounds, $\mathrm{search}(7, \mathrm{exact}, w)$ and $\mathrm{search}(8, \mathrm{unique}, w)$ both return false, exhausting 7,767 and 2,526 nodes respectively, so Proposition 3.3 gives $A_{Ne}(w) \ge 8$ and $A_N(w) \ge 9$. □

Both searches are finite and each was run to exhaustion as part of the proof; the lower bounds rest on that exhaustion together with Proposition 3.3, and on nothing else. The two subset constructions are likewise finite evaluations.

Corollary 4.2. *No automaton with at most seven states, over any alphabet whatsoever, exactly accepts 0123444444445126; and no automaton with at most eight states, over any alphabet, uniquely accepts it.*

Proof. Proposition 3.3 rules out the automata over the word's own seven letters, and Proposition 3.5 transfers the two statements to every larger alphabet. □

Theorem 4.3. *Let $w' =$ 012344444444512 $=$ 0123$4^8$512, a word of length 15 over $\{0,\dots,5\}$, and let $w'' =$ 012333333334015 be the canonical form of its reversal. Then*

$$A_{Ne}(w') = A_{Ne}(w'') = 7 \ < \ 8 = A_N(w') = A_N(w'').$$

Proof. As for Theorem 4.1. For w' the exactly accepting automaton is E above with the transition $q_3 \xrightarrow{6} q_7$ and the state q_7 deleted and q_3 taken as the final state, and the uniquely accepting one is U with $q_7 \xrightarrow{6} q_8$ and q_8 deleted and q_7 taken as final; the accepting paths are the ones displayed above with their last entry removed, and $\mathrm{search}(6, \mathrm{exact}, w')$ and $\mathrm{search}(7, \mathrm{unique}, w')$ exhaust 941 and 967 nodes. For w'' a seven-state and an eight-state automaton, each with its accepting path, are exhibited separately, and the same two searches return false. □

The relation between the two theorems is the source's own hint that "the witnessing NFA simply gets a head and/or tail segment appended": $w = w' \cdot \mathtt{6}$, and the automata for w are those for w' with one fresh state and one fresh transition appended. The same operation, applied to the source's length-13 word x^*, is how these words were found: w is x^* with its run of fours lengthened from five to eight. We stress that the values $(8, 9)$ and $(7, 8)$ are the source's; what Theorems 4.1 and 4.3 add is a word realising each row, an automaton realising each upper bound, and a proof.

Theorem 4.4. *The words 0123455555555623 and 0123333333340156, of length 16 over seven letters and reversals of one another up to renaming, both satisfy $A_{Ne} = 8 < 9 = A_N$.*

Remark 4.5. That there should be exactly two separating words of length 15, forming a reversal pair, is corroborated by the source itself, though not by its printed text: the only commented-out mathematical line in the LaTeX source of its e-print — the last line of the caption of its Figure 1, immediately after the sentences describing the $n = 14$ diagram — reads "Further below: the NFA for $n = 15$ (its reverse is the other example)." It names no word, and, being a comment, it does not appear in the compiled paper; we record it as corroboration and nothing more. The census of Remark 6.1 finds exactly two parents at length 15, and they are 012344444444512 and its reversal.

5. Length 14: every separating word

Theorem 5.1. *Each of the following twenty canonical words of length 14 satisfies $A_{Ne} = 6$ and $A_N = 7$:*

01011111111001	01011111111201	01022222222001	01022222222101
01022222222301	01100000000101	01100000000201	01122222222001
01122222222101	01122222222301	01200000000101	01200000000201
01200000000301	01211111111001	01211111111201	01211111111301
01233333333001	01233333333101	01233333333201	01233333333401

Two of them use two letters, ten use three, seven use four and one uses five.

Proof. Forty explicit automata — one six-state exactly accepting and one seven-state uniquely accepting automaton per word, each with an accepting path — give the upper bounds through Proposition 3.4; forty exhausted searches, $\mathrm{search}(5, \mathrm{exact}, \cdot)$ and $\mathrm{search}(6, \mathrm{unique}, \cdot)$, give the lower ones through Proposition 3.3. □

Remark 5.2 (The census at length 14; a computation outside the formal development). These twenty are all of them. Following the aggregate method of [8, §6] — for each state sequence, group the positions $1, \ldots, n$ into the classes forced by the sequence, and match every canonical word against the resulting position partitions — an independent implementation in C enumerated the

$$\textstyle\sum_{k \le 8} S(15, k) \;=\; 1{,}301{,}576{,}801$$

restricted growth state sequences over at most 8 states (S the Stirling numbers of the second kind), found 918 position partitions of unique type and 715 of exact type (1,350 distinct partitions in all: a partition can be of both types, and 283 are), and 64,531 candidate words, of which exactly 20 separate the two measures, namely those of Theorem 5.1, each with $(A_{Ne}, A_N) = (6, 7)$. All twenty are letter-mergings of `01233333333401`: of the $B_5 = 52$ ways to merge its five letters, exactly 20 yield separating words. Exactly two are binary, and they are a reversal pair, `01011111111001` and `01100000000101`. The enumeration takes well under an hour of CPU time.

Three checks were run on it. All twenty words were re-confirmed by the per-word searches of Section 3, a different algorithm on a different data path. All 52 mergings of `01233333333401` were run through those searches, and exactly the twenty separate. And the set of twenty is closed under reversal-and-recanonicalisation, as the reversal invariance of both measures demands, with four fixed points; nothing in the census enforces this.

Remark 5.3 (The Hyde–Kjos-Hanssen bound at length 14; a computation outside the formal development). The same enumeration verifies, rather than assumes, the bound $A_N(x) \le \lfloor n/2 \rfloor + 1$ at length 14: every one of the $B_{14} = 190{,}899{,}322$ canonical words of length 14, over every alphabet, is matched by a unique-type position partition using at most 8 states, so $A_N(x) \le 8$ throughout. There are no exceptions; the check took 12.4 CPU-seconds. The source performs the same verification for every length $n \le 13$, and that verification was reproduced here first (see Remark 7.5). We note that the bound itself is already formalised in Lean 4 by the first author of [8] [1]; what is added here is only its numerical confirmation at one further length.

6. Length 15: the census

Remark 6.1 (The census at length 15; a computation outside the formal development). Exactly 74 canonical words of length 15, over all alphabets, satisfy $A_{Ne} < A_N$, and all of them have $(A_{Ne}, A_N) = (7, 8)$. The same implementation enumerated the

$$\textstyle\sum_{k \le 7} S(16, k) \;=\; 7{,}291{,}973{,}067$$

restricted growth state sequences over at most 7 states in three shards, about 3.5 CPU-hours at a measured 576 thousand sequences per second, and found 201 position partitions of unique type and 138 of exact type (the two counts overlap) and 24,720 candidate words. The cap of seven states is legitimate here because a separating word of length 15 has $A_{Ne} < A_N \le \lfloor 15/2 \rfloor + 1 = 8$ by the Hyde–Kjos-Hanssen bound [4], hence $A_{Ne} \le 7$; unlike at lengths ≤ 14, that bound is *cited* here rather than re-derived, and Section 9 prices the re-derivation.

The 74 form 37 reversal pairs, none of them a fixed point of reversal. They split cleanly into the 37 letter-mergings of `012344444444512` that separate and the 37 of its reversal `012333333334015`, the two sets being disjoint. *None of the* 74 *is binary*: they use 3, 4, 5 and 6 letters, with 16, 38, 18 and 2 words respectively. Lengths 13 and 14 both carry binary witnesses — four and two canonical ones — so the least alphabet size admitting a separating word is $2, 2, 3$ at lengths $13, 14, 15$.

Three checks, as before: all 74 were re-confirmed by the per-word searches; all $B_6 = 203$ letter-mergings of `012344444444512` were run through those searches, exactly 37 separating and 166 not, in complete agreement; and the set is closed under reversal-and-recanonicalisation. Two of the 74 are the words certified in Theorem 4.3.

7. The published values, reproduced and certified

Before claiming anything new we recomputed, from an independent implementation, every number that [8] publishes. The values in this section are the source's; the machine-checked proofs are ours.

Proposition 7.1 ([8, Theorem 1.1])**.** *For the binary word w_{13} = 1101000000100, $A_{Ne}(w_{13}) = 6$ and $A_N(w_{13}) = 7$.*

The six-state automaton certifying $A_{Ne}(w_{13}) \leq 6$ is the source's: our search finds as its first solution the state sequence $0, 1, 2, 3, 4, 5, 4, 5, 5, 4, 1, 2, 3, 0$, which is the path [8] names, and the automaton it induces has exactly the nine transitions of the source's Figure 1. The search establishing $A_N(w_{13}) \geq 7$ visits 473 nodes with the depth profile $1, 2, 5, 9, 18, 36, 65, 107, 97, 71, 34, 23, 5, 0$, which is the profile [8, §4] prints; the two auxiliary counts 12,609,265 and 42,019,440 of label-consistent restricted growth sequences over five and six states were reproduced as well.

Proposition 7.2 (after [8, Proposition 3.1])**.** *The six-state automaton witnessing $A_{Ne}(w_{13}) \leq 6$ has an accepting path for w_{13} other than the one exhibited; hence it does not witness $A_N(w_{13}) \leq 6$.*

The source's Proposition 3.1 says more, namely that there are exactly three accepting paths; we certify only that there is more than one, which is what the separation needs. This doubles as the non-vacuity check for the second construction of Proposition 3.4: the set $D(x, p)$ there is not always disjoint from the final states.

Proposition 7.3. *$A_{Ne} = 6 < 7 = A_N$ for 01233333333401 and for 01011111111001, both of length* 14*.*

The first word is printed in the caption of the source's Figure 1 ("All examples are homomorphic images of the word `01233333333401`") and its values are the table's row 14. The second is not printed anywhere in the source as a word, but it is recoverable from it: the source's $n = 14$ diagram is a fully labelled binary automaton, and decoding it — states $q_0, \dots, q_5$, initial q_0, final q_2, transitions $q_0 \xrightarrow{0} q_1$, $q_1 \xrightarrow{1} q_2$, $q_2 \xrightarrow{0} q_3$, $q_3 \xrightarrow{1} q_4$, $q_4 \xrightarrow{1} q_5$, $q_5 \xrightarrow{1} q_3$, $q_5 \xrightarrow{1} q_5$, $q_4 \xrightarrow{0} q_0$ — yields exactly this word. The census of Remark 5.2 finds it independently, as one of the two binary words of that length.

Proposition 7.4. *A_{Ne}(01234555555567128) $= 8 < 9 = A_N$(01234555555567128), a word of length* 17 *over nine letters.*

This word is likewise decoded from a published diagram, the source's $n = 17$ automaton, an eight-state automaton over $\{0, \dots, 8\}$ with initial and final state q_0. That the decoded word reproduces the source's table row 17 exactly is the strongest available check that the decoding is right.

Remark 7.5 (The census at length 13, reproduced; a computation outside the formal development). Every published number of [8, Theorem 1.2, §6] was reproduced by the independent implementation before anything above was claimed: the $\sum_{k \leq 7} S(14, k) = 164{,}029{,}595$ state sequences; the 333 position partitions of unique type and the 242 of exact type (475 distinct, 100 of both types); the $B_{13} = 27{,}644{,}437$ canonical words matched against them; the 365 separating words, distributed over alphabet sizes $2, \dots, 7$ as $4, 75, 170, 97, 18, 1$; the fact that all 365 are letter-mergings of x^*; and the absence of any separating word at every length $4 \leq n \leq 12$. The whole length-13 computation takes 262 CPU-seconds and 4 megabytes, which matches the source's "the whole computation takes a few minutes".

Remark 7.6 (Four or eight binary words at length 13?). The source's Theorem 1.2 states that "up to injective renaming of letters there are exactly 4 separating words of length 13 using exactly 2 letters, namely `1101000000100` and `0101000001100` and their reversals", while its abstract says of the same 365 words that "exactly eight of them are binary". Our reproduction agrees with the theorem: there are four canonical binary words of length 13, namely `0010000001011`, `0010111111011`, `0011000001010` and `0101000001100`, which are the canonical forms of the four words the theorem names. Each canonical class is realised by exactly two words over a fixed two-letter alphabet, so the two counts are consistent once the abstract's "eight" is read as counting words over $\{0,1\}$ rather than classes up to renaming.

8. Controls

A development whose lower bounds all come from one search needs evidence that the search can fail to fail, and that the certificates can fail to pass.

Proposition 8.1. $A_{Ne}(\mathtt{01234}^6\,\mathtt{5126}) = A_N(\mathtt{01234}^6\,\mathtt{5126}) = 7$.

So lengthening the run of fours in the source's x^* by one destroys the separation: the shape of the witness alone does not produce it. The same seven-state automaton serves for both upper bounds, and the two searches search$(6,\cdot,\cdot)$ both return false.

Proposition 8.2. (i) search(8, *exact*, `0123444444445126`) *and* search(9, *unique*, `0123444444445126`) *both return* true. (ii) *For the six-state automaton of Proposition 7.1 and the word* `1101000000101` *— the length-13 witness with its last letter flipped — the set* $R(\cdot)$ *of Proposition 3.4 does contain a final state.*

Part (i) rules out the failure mode in which the searches of Theorem 4.1 return false because their enumeration is empty or their prune is unsound in the wrong direction: one state higher, the same enumeration succeeds. Part (ii) rules out the failure mode in which the exact-acceptance certificate is passed by every automaton: here a final state *is* reached by a different word of the same length, namely `1101000000100`, which that automaton accepts. Together with Propositions 8.1 (a word where the two measures agree) and 7.2 (an automaton that is exactly but not uniquely accepting) these are the negative controls of this work.

Remark 8.3 (The run family; five of its eight columns are computations outside the formal development). Write $x(m) = \mathtt{01234}^m\,\mathtt{5126}$, a word of length $m+8$, so that $x(5) = x^*$ and $x(8) =$ `0123444444445126`. The per-word searches give

m	5	6	7	8	9	10	11	12
A_{Ne}	6	7	7	8	7	8	9	9
A_N	7	7	7	9	7	8	9	9

so the separation holds at $m=5$ and $m=8$ and fails at the other six values. The entry $m=5$ is the source's; $m=6$ is Proposition 8.1 and $m=8$ is Theorem 4.1, both machine-checked; the remaining five columns were computed by the reference implementation only. No pattern is claimed, and the table is evidence against any easy one.

9. What remains

Is the binary case sporadic? The least alphabet size carrying a separating word is 2 at length 13, 2 at length 14 and 3 at length 15. Length 16 would begin to answer it, and the per-word searches can hunt for a binary length-16 witness far more cheaply than a full census can.

The Hyde–Kjos-Hanssen bound at length 15. The census of Remark 6.1 takes its cap from the published bound rather than re-deriving it, as was done at every length up to 14. Closing that gap is the eight-state layer: $\sum_{k\le 8} S(16,k) = 9{,}433{,}737{,}120$ sequences, about 4.5 CPU-hours at the rate measured above.

The census at length 16. $\sum_{k\le 8} S(17,k) = 69{,}998{,}462{,}014$ sequences, about 34 CPU-hours. The head-and-tail extension used in Section 4 is the cheap substitute, and is how the words of Theorems 4.1 and 4.3 were found.

The counts. The number of canonical separating words is 0 at every length up to 12, then 365, 20, 74 at lengths 13, 14, 15. The sequence is in no database we searched; its next term is length 16 and is out of reach by the census route. The non-monotonicity is itself striking, and so is the structure behind it: at lengths 13 and 14 all separating words are the letter-mergings of a single parent that is a palindrome up to renaming, while at length 15 there are two parents forming a reversal pair. Is there always a finite set of parents, and how does its size grow?

A proof rather than a computation. Every separation known at any length is a finite check. A proof that some infinite family of words separates the two measures would be the first result here that is not a computation; Remark 8.3 shows that the most obvious family is not one.

10. Formal verification

Every theorem, proposition, lemma and corollary stated above is formalised and machine-checked in Lean 4 [9] with Mathlib [10], with the explicitly labelled exceptions: Remarks 5.2, 5.3, 6.1 and 7.5, which report the aggregate computations run in C, the last five columns of the table in Remark 8.3, and the auxiliary counts quoted after Proposition 7.1. The node counts reported for the searches are likewise measurements of the reference implementation and not part of any certified statement; what is certified is that each search returns false. The development is six files and about 2,250 lines: the definitions and the reduction of Section 3; the certificates of Proposition 3.4; seven words with fourteen explicit automata, covering Theorems 4.1 and 4.3 and Sections 7 and 8; the twenty words of Theorem 5.1 with their forty automata; three further witnesses with six more automata, for Theorems 4.3 and 4.4; and the alphabet restriction of Proposition 3.5. There is no incomplete proof anywhere in it.

The axiom set of every statement was printed. No part of the reasoning layer is proved by compiled evaluation: Proposition 3.5 depends on propositional extensionality alone, Proposition 3.4 on that together with quotient soundness, and Lemmas 3.1 and 3.2 and Proposition 3.3 on those two together with the axiom of choice. The value statements additionally depend on named compiled-evaluation axioms, one for each finite evaluation: for each word, one for the exact-acceptance certificate, two for the unique-acceptance certificate and one for each exhausted search. This is the split we consider the right one: what is trusted to a compiled program is the value of a decidable predicate on a fixed input, while every step from those values to the statements about all automata is a kernel-checked proof.

Two measurements bear on the size of the searches. The normalisation of Lemma 3.2 is what makes them feasible: the two searches of Theorem 4.1 exhaust 7,767 and 2,526 nodes, against 5,533,592 and 11,590,651 without it — a factor of 712 on the first, close to the $6! = 720$ relabellings it removes — and the searches at the other lengths behave the same way. That the search establishing $A_N(\texttt{1101000000100}) \geq 7$ exhausts exactly 473 nodes, with the depth profile printed in [8, §4], is an independent check on the whole reduction. The development elaborates in about 2.7 CPU-minutes on one core, on top of the library import. The repository is private; repository reference to be added at submission.

References

[1] B. Kjos-Hanssen, *ac-exercises*, Lean 4 repository, `https://github.com/bjoernkjoshanssen/ac-exercises`, branch `main`, head commit `146741f` of 20 June 2026. Its file `Acmoi/HydePrelim.lean` defines the two measures through the predicates `A_Ne_at_most` and `A_N_at_most` and proves $A_{Ne} \leq A_N$; `Acmoi/HydeTheorem.lean` defines A_N and A_{Ne} by `Nat.find` and proves the bound of [4] as `A_N_bound`. We searched the tree and found no separating word, no folding lemma, no exhaustive search, no `native_decide` and no computed value of either measure. The repository is public but not archived, so we pin it by commit: it was cloned and searched twice on 3 September 2026 and stood at `146741f` both times.

[2] J. Chen, B. Kjos-Hanssen, I. Koswara, L. Richter and F. Stephan, *Languages of words of low automatic complexity are hard to compute*, in: 45th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2025), C. Aiswarya, R. Mehta and S. Roy (eds.), LIPIcs 360, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025, Article 24, pp. 24:1–24:19, `doi:10.4230/LIPIcs.FSTTCS.2025.24`; arXiv: 2510.07696. Volume, editors, article number and page range confirmed against the publisher's record and against dblp on 3 September 2026; the question quoted in Section 1 is Question 48, on p. 24:16 of the published article.

[3] K. Hyde, *Nondeterministic finite state complexity*, Master's thesis, Department of Mathematics, University of Hawaiʻi at Mānoa, 2013, `http://hdl.handle.net/10125/29507`. Title, author, degree, department, institution and year confirmed against the university's repository record on 3 September 2026.

[4] K. K. Hyde and B. Kjos-Hanssen, *Nondeterministic automatic complexity of overlap-free and almost square-free words*, Electron. J. Combin. **22**(3) (2015), Paper 3.22, 18 pp., `doi:10.37236/4851`; arXiv:1402.3856. The source of the bound $A_N(x) \leq \lfloor n/2 \rfloor + 1$ used in Remarks 5.3 and 6.1. Title, authors and journal reference confirmed against the arXiv metadata record and against Crossref on 3 September 2026.

[5] B. Kjos-Hanssen, *Few paths, fewer words: model selection with automatic structure functions*, Exp. Math. **28**(1) (2019), 121–127, `doi:10.1080/10586458.2017.1368048`; arXiv:1608.01399. Title, author and identifier confirmed against the arXiv metadata record, which carries no journal reference; journal, volume, issue, pages and year of publication confirmed against Crossref. Both checked 3 September 2026.

[6] B. Kjos-Hanssen, *Conditional automatic complexity and its metrics*, arXiv:2308.16292. Records the question answered by [8] as open. Title, author and identifier confirmed against the arXiv metadata record on 3 September 2026; no journal data is claimed.

[7] B. Kjos-Hanssen, *Automatic complexity: a computable measure of irregularity*, De Gruyter Series in Logic and its Applications 12, De Gruyter, Berlin, 2024, `doi:10.1515/9783110774870`.

[8] B. Kjos-Hanssen and T. Rivera Petit, *Exact versus unique nondeterministic automatic complexity*, arXiv: 2608.22271v1, submitted 23 August 2026 (the manuscript is dated 21 August 2026), cs.FL with a math.LO cross-list. All quotations and all statements about its figures are from the LaTeX source of the e-print of v1 (one file, `exact-vs-unique_1.tex`, 30,284 bytes), retrieved twice on 3 September 2026 and byte for byte identical both times; v1 was the only version on both occasions.

[9] L. de Moura and S. Ullrich, *The Lean 4 theorem prover and programming language*, in: Automated Deduction — CADE 28, Lecture Notes in Computer Science 12699, Springer, 2021, pp. 625–635.

[10] The mathlib Community, *The Lean mathematical library*, in: Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2020), ACM, 2020, pp. 367–381.

[11] J. Shallit and M.-W. Wang, *Automatic complexity of strings*, J. Autom. Lang. Comb. **6**(4) (2001), 537–554, `doi:10.25596/jalc-2001-537`. Confirmed against dblp on 3 September 2026.