RANKING FROM COUNTEREXAMPLES: THE EXACT QUERY COMPLEXITY FOR AT MOST SEVEN ITEMS, AND THE FAILURE OF THE SORTING PATTERN AT SEVEN

KOREA SUPERINTELLIGENCE LABS

Abstract. In the ranking-from-counterexamples game of Alon, Moran and Moran, a learner repeatedly proposes a linear order on n items and an adversarial oracle either declares the proposal correct or names an unordered pair whose relative order the proposal gets wrong. Write $Q(n)$ for the number of proposals a worst-case-optimal deterministic learner needs when every counterexample is truthful, counting the confirming proposal itself. We determine $Q(n)$ for every $n \le 7$:

$$Q(1), \dots, Q(7) \;=\; 1,\ 2,\ 3,\ 5,\ 7,\ 10,\ 12.$$

The values for $n \le 6$ turn out to be published mathematics in a different language: they are $h^{(4)}(S_n) + 1$, where $h^{(4)}(S_n)$ is the minimum depth of a decision tree with proper hypotheses for the sorting decision table, computed by Azad, Chikalov, Hussain and Moshkov in 2021. Neither that work nor the 2026 source cites the other; we identify the two models and re-derive the values. The value at $n = 7$ is new, and it breaks a pattern. For every $n \le 6$ one has $Q(n) = \lfloor \log_2 n! \rfloor + 1$, and for $3 \le n \le 6$ this is exactly the number of comparisons needed to sort n items, whereas $Q(7) = 12 < 13 = \lfloor \log_2 7! \rfloor + 1$. Eleven counterexamples and a confirming round beat the thirteen comparisons that sorting seven items requires, even though the learner never chooses which pair it is told about. We also settle, in the negative and at the level of exact values, a question the source poses about the game started from a poset P of already-known comparisons: two posets on six points, each with 8 linear extensions and width 2, have values 3 and 4. Every value stated as a theorem below is proved in both directions and machine-checked in Lean 4; Section 8 says exactly what is and what is not formalised.

1. Introduction

The game. Alon, Moran and Moran [1] study the following interaction, which they call *ranking from counterexamples*. We quote their model box.

> **Ranking from counterexamples.** There is an unknown target ranking $\pi^\star$ over n items. On each round:
> (1) The learner proposes a linear order π.
> (2) The oracle either declares that π is correct, in which case $\pi = \pi^\star$ and the interaction terminates, or returns an unordered pair $\{i, j\}$, indicating that the relative order of i and j in π is wrong.
>
> A returned pair is *truthful* if π and $\pi^\star$ indeed disagree on the relative order of i and j, and *untruthful* otherwise. We allow at most k untruthful counterexamples throughout the interaction, where k is not known to the learner. Declarations that the proposed ranking is correct are always truthful.

This paper is about the truthful case $k = 0$: every returned pair is one that the proposal really gets wrong, and a correct proposal must be declared correct.

Two features separate the game from comparison sorting, and both are the source's emphasis. The learner does not choose which comparison it learns about — the oracle does, and adversarially, among all the pairs the proposal gets wrong; and every query must itself be a linear order. The second restriction is what the source calls *properness*, and it is what makes the Condorcet phenomenon relevant: the pairwise majority of a set of rankings need not be a ranking, so a learner cannot simply query the majority orientation of each pair.

The source's main theorem is that the optimal number of queries is $\Theta(n \log n + nk)$, so that each untruthful counterexample costs an extra factor of n over the improper case; the upper bound is a geometric form of weighted majority resting on Grünbaum's theorem [6], the lower bound combines a sorting argument with a Condorcet-cycle construction. Its open-questions section then asks, verbatim:

> **Optimal constants for unrestricted rankings.** Even for unrestricted rankings, where our bounds are tight up to constant factors, the precise query complexity remains open. In particular, it would be interesting to determine the optimal constants in front of the $n \log n$ and nk terms.

What this paper does. We compute the exact value at $k = 0$ for every $n \le 7$. Write $Q(n)$ for the number of proposals a worst-case-optimal deterministic learner needs, *counting the confirming proposal*: the interaction ends when the oracle confirms, so a learner that already knows the target still spends one round, and $Q(1) = 1$. The convention matters for comparison with the literature and is fixed precisely in Section 2; it is also one of the machine-checked control statements (Proposition 3.4).

n	1	2	3	4	5	6	7
$Q(n)$	1	2	3	5	7	10	**12**
$\lfloor \log_2 n! \rfloor + 1$	1	2	3	5	7	10	13

The headline is the last column. Nothing forces one bit per round in this game — a single proposal has up to $\binom{n}{2}$ possible answers, so a round can be worth much more than a bit, and $\lceil \log_2 n! \rceil$ is not a valid lower bound for the number of proposals. What is surprising is that the coincidence survives six terms and then breaks. Put beside sorting, which is how the source frames the noiseless case (its abstract says that "the noiseless complexity matches the classical complexity of sorting"), the comparison is sharper than the statement up to constants. Let $S(n)$ be the minimum number of comparisons that sorts n items in the worst case, so that $S(3), \dots, S(7) = 3, 5, 7, 10, 13$ [8, 10]. For $3 \le n \le 6$ we get $Q(n) = S(n)$ on the nose: the counterexample learner needs $S(n) - 1$ informative answers and then one round to have its guess confirmed, and the two effects cancel. At $n = 7$ they do not.

Eleven counterexamples plus a confirmation beat thirteen chosen comparisons —

even though the learner never chooses which pair it hears about. That is Theorem 5.2 and Corollary 5.3.

An older literature computing the same numbers. The values for $n \le 6$ are not new, though it takes an identification of two models to see it. Azad, Chikalov, Hussain and Moshkov [3, 4] study *decision trees with hypotheses* for a decision table T with attributes $f_1, \dots, f_m$. Besides asking for the value of one attribute, such a tree may ask whether a *hypothesis* $\{f_1 = \delta_1, \dots, f_m = \delta_m\}$ is true; the answer is a confirmation or a counterexample $f_i = \sigma$ with $\sigma \ne \delta_i$. In their words, such a hypothesis is proper when the tuple $(\delta_1, \dots, \delta_m)$ "is a row of the table T" [3, §2]. They write $h^{(4)}(T)$ for the minimum depth of a decision tree for T that uses proper hypotheses only.

For the sorting decision table S_n — attributes the $\binom{n}{2}$ comparisons, rows the $n!$ permutations — a proper hypothesis is exactly a linear order, and a counterexample names a misordered pair. That is the game above at $k = 0$. Their published table [3, Table 1] (reproduced as [4, Table 6]) reads

$$h^{(4)}(S_n) \;=\; 2,\; 4,\; 6,\; 9 \qquad (n = 3, 4, 5, 6),$$

which is $Q(n) - 1$; the difference of one is the confirming round, which a decision tree does not pay because it may output the answer at a node without asking. We prove the identification as Proposition 4.3 and record here that neither literature cites the other: the source's related-work section names Angluin's equivalence queries [2], Vaandrager's survey and Braverman et al., but not Moshkov, and Moshkov's papers do not mention ranking from counterexamples. So $Q(n)$ for

$n \le 6$ is published mathematics, newly machine-checked here, and $n = 7$ — one step past where the published table stops — is new.

Results.

- *The published range* (Section 4). $Q(n) = 1, 2, 3, 5, 7, 10$ for $n \le 6$, each proved in both directions against the unreduced model. These re-derive $h^{(4)}(S_n)$ for $n = 3, \dots, 6$ and are stated here because everything past $n = 6$ rests on the same machinery, and because $n \le 6$ is where that machinery can be checked against a published table.
- *Seven items* (Section 5). $Q(7) \ge 12$ (Theorem 5.1) and $Q(7) = 12$ (Theorem 5.2); hence the pattern break (Corollary 5.3). For scale: the source's $\Omega(\log L)$ lower bound for the game started from a poset with L linear extensions rests on the Kahn–Saks fraction $3/11$ [7], and running that recursion at the seven-element antichain gives $\lceil \log_{11/3} 5040 \rceil = 7$ informative rounds, against the eleven that the truth requires. The evaluation is ours: the source prints the constant and the asymptotic statement, not this number, and its $\Omega(n \log n)$ bound for unrestricted rankings is proved differently, by forcing the learner along a root-to-leaf path of a balanced comparison tree for sorting. The source's bounds are stated up to constant factors and were never meant to give the exact value; that is exactly what its open question asks for.
- *Starting from a poset* (Section 6). The source also asks about the game started from a poset P of known comparisons with L linear extensions and width x, and whether the complexity "is determined by L and x alone, or whether additional structural parameters of the poset are needed". Theorem 6.1 exhibits two posets on six points, both with $L = 8$ and $x = 2$, whose exact values are 3 and 4. As Section 6 stresses, this settles the exact-value form of the question and *not* the up-to-constant-factors form that the source actually poses.
- *Method* (Section 3). Two certificate formats with soundness proofs against the model as stated, plus invariance of the game under renaming the items. The invariance is what makes $n = 7$ affordable: it takes the strategy certificate from about half a million entries to 229.

Section 7 records two computations that are *not* part of the formal development and are labelled as such throughout: the first exact values of the game with lies, and the fact that dropping properness does not change the value at $n = 7$. Section 8 describes the formal verification, and Section 9 the frontier.

2. Preliminaries

Fix $n \ge 1$ and let the items be $1, \dots, n$. A *ranking* is a linear order on the items, written as the list of items from first to last; $i \prec_\pi j$ means that π ranks i ahead of j. Two rankings π, σ *disagree* on the unordered pair $\{i, j\}$ when $i \prec_\pi j$ and $j \prec_\sigma i$, or the other way round; a truthful oracle may return $\{i, j\}$ as a counterexample to the proposal π exactly when the target disagrees with π there.

We describe a position of the game by the set V of rankings still consistent with everything returned so far, and define winning within a budget by recursion on the budget.

Definition 2.1. For a set V of rankings of n items and $d \in \mathbb{N}$, say that the learner *wins* V *within* d when

- $d = 0$ and $V = \emptyset$; or
- $d = e + 1$ and there is a ranking π such that for every pair $\{i, j\}$ of distinct items, *either* no $\sigma \in V$ disagrees with π on $\{i, j\}$, *or* the learner wins $\{\sigma \in V : \sigma \text{ disagrees with } \pi \text{ on } \{i, j\}\}$ within e.

Winning within d implies winning within $d + 1$, so $Q(V) := \min\{d : \text{the learner wins } V \text{ within } d\}$ is well defined whenever the minimum exists, and it does for every V occurring below. We write $Q(n)$ for $Q(V)$ with V the set of all $n!$ rankings.

Two readings of Definition 2.1 are worth spelling out. First, if no pair is available at all — that is, every $\sigma \in V$ agrees with π on every pair, so $V \subseteq \{\pi\}$ — then the first alternative holds for

every pair, the oracle must declare π correct, and the round in which that happens is counted. In particular $Q(\{\pi\}) = 1$, not 0: the interaction ends when the *oracle* confirms. Second, the adversary is not required to commit to a target in advance; it must only keep some consistent target alive, which is the "adversary does not commit" device the source itself uses in its lower bounds. The definition also quantifies the learner's proposal over *all* rankings and the oracle's answer over *all* pairs; no reduction to a smaller move set is assumed anywhere below.

States are posets. A truthful counterexample $\{i, j\}$ to π says that the target orders i and j the other way round from π. So after any history the surviving set V is the set $\mathcal{L}(P)$ of linear extensions of the poset P generated by the relations returned so far, and the game is a game on posets: from P, a proposal π and an answer $\{i, j\}$ lead to P together with the reverse of the orientation π gives $\{i, j\}$. We write $Q(P) := Q(\mathcal{L}(P))$ and $e(P) := |\mathcal{L}(P)|$, so that $Q(n) = Q(A_n)$ for A_n the n-element antichain, and

$$Q(P) \;=\; \begin{cases} 1, & e(P) = 1, \\ 1 + \min\limits_{\pi \in \mathcal{L}(P)} \; \max\limits_{\{a,b\} \text{ incomparable in } P} Q\big(P + (\text{reverse of } \pi \text{ on } \{a,b\})\big), & \text{otherwise.} \end{cases} \tag{1}$$

Formula (1) restricts the learner to proposals inside $\mathcal{L}(P)$; that is a restriction we never rely on. A proposal violating a relation of P is answered by that relation's pair, which every consistent target disagrees with, so the version space does not move — and in the formal development this is not even a lemma, because the certificates of Section 3 are checked against Definition 2.1 directly, with the proposal ranging over all rankings.

The inner minimum–maximum is an acyclicity test. The child of a proposal π and an answer $\{a, b\}$ depends on π only through the orientation π gives that pair. So for a level M, call an ordered pair (u, v) *good at level* M when $Q(P + (u \prec v)) \geq M + 1$, meaning the adversary is content to answer $\{u, v\}$ against a proposal that puts v before u. A proposal escapes level M precisely when it orients every good pair the other way; such a proposal exists exactly when the digraph of good pairs has a linear extension, i.e. is acyclic. Hence the inner minimum–maximum in (1) is an $O(n^2)$ acyclicity test per level rather than an enumeration of $\mathcal{L}(P)$. This is what makes the search of Section 5 feasible: it visits one state per labelled poset on n points — 19, 219, 4231, 130,023, 6,129,859 for $n = 3, \ldots, 7$ [10, A001035] — and does $O(n^2)$ work at each. It also explains the shape of the two certificates: an upper bound needs a linear extension of the forced digraph, a lower bound needs a *cycle* in the good-pair digraph.

3. The two certificate formats and relabelling invariance

Nothing in this section is new mathematics; it is the apparatus every value below rests on, and it is described because the exactness of the values depends on what exactly the machine checks. A state is stored as a *constraint mask*: a set P of ordered pairs, with $\mathcal{L}(P)$ the rankings obeying all of them.

Proposition 3.1 (Strategy certificates)**.** *Let T be a finite table whose entries are quadruples (P, π, m, c), where P is a constraint mask, π a ranking, $m \geq 1$ a budget, and c assigns to each pair $\{x, y\}$ of distinct items either the token* FIXED *or an index into T. Suppose that for every entry and every pair $\{x, y\}$:*

- *if $c(\{x, y\}) =$* FIXED*, then P already forces $\{x, y\}$ the way π orders it, so no consistent target disagrees with π there; and*
- *if $c(\{x, y\})$ is the index j, then the budget of entry j is at most $m - 1$, and the mask of entry j is contained in P together with the reversed pair and its transitive consequences.*

Then for every entry (P, π, m, c) of T the learner wins $\mathcal{L}(P)$ within m; in particular $Q(P) \leq m$.

Proposition 3.2 (Adversary certificates)**.** *Let T be a finite table whose entries are quadruples (P, m, σ, c), where P is a constraint mask, $m \geq 1$ a budget, σ a ranking, and c a cyclic list $c_1 \to c_2 \to \cdots \to c_k \to c_1$ of items. Suppose that for every entry: $\sigma \in \mathcal{L}(P)$; and for every edge (u, v) of the cycle, either $m = 1$, or the mask of some entry of budget at least $m - 1$ contains P together with the relation $u \prec v$. Then for every entry and every d, if the learner wins $\mathcal{L}(P)$ within d then $m \leq d$; in particular $Q(P) \geq m$.*

The two containment directions are opposite on purpose, and that is what keeps the tables small: an upper-bound entry may be *more general* than the state it serves (fewer constraints), a lower-bound entry *more specific*, so one generic entry serves many states. The argument behind Proposition 3.2 is the Condorcet mechanism the source itself uses: no ranking orders a cycle consistently, so whatever the learner proposes, at least one edge (u, v) of the cycle is oriented v before u by the proposal, the oracle may answer $\{u, v\}$, and that answer costs the learner a round while leaving a position of value at least $m - 1$.

Proposition 3.3 (Relabelling invariance)**.** *Let τ be a permutation of the items and P a constraint mask. If the learner wins $\mathcal{L}(P)$ within m, then the learner wins $\mathcal{L}(\tau P)$ within m, where τP is the image of P under τ. Consequently $Q(\tau P) = Q(P)$.*

Proposition 3.3 is proved by transporting a strategy along $\sigma \mapsto \tau \circ \sigma$: relabelling preserves positions, hence the relation $\prec$, hence the disagreement relation, and the pair $\{x, y\}$ of the transported game is the pair $\{\tau^{-1}x, \tau^{-1}y\}$ of the original. It is not a nicety. With it, a table entry may be used for any state that contains (upper bound) or is contained in (lower bound) a *relabelling* of the entry's mask, so one entry covers an entire isomorphism class and everything that class generalises. Measured on the searches of the next two sections:

	strategy table		adversary table	
n	labelled	up to isomorphism	labelled	up to isomorphism
5	$\sim 10^3$	18	127	12
6	$\sim 10^4$	61	1,023	13
7	$\sim 5 \cdot 10^5$	**229**	4,095	**36**

The labelled strategy sizes are given only to an order of magnitude, and deliberately. Unlike the other three columns they are not pinned down by the game: they depend on how hard the generator looks for an already-stored entry general enough to cover a child, and widening or narrowing that search moves them: at $n = 7$, widening it from the narrowest setting to a moderate one already takes the count from 529,194 entries to 485,746, and our working notes disagree with each other at $n = 5$ and $n = 6$. No statement in this paper uses these figures. The isomorphism-aware sizes, by contrast, are the sizes of the tables actually shipped and re-checked inside the kernel, and the labelled adversary table comes out as exactly $2^{Q(n)} - 1$ entries at every $n \leq 7$ — see Remark 5.4. At $n = 7$ the strategy table therefore shrinks by a factor of more than 2,000, and the adversary table from 4,095 entries to 36; that is the difference between a table too large to check and a check that takes seconds. Each child pointer of an isomorphism-aware table stores the relabelling it uses; the inverse permutation is computed and then *checked* to be an inverse, so nothing rests on the computation being right.

Finally, the conventions and the negative controls.

Proposition 3.4. (i) *With a single candidate left the learner still wins within one proposal and does not win within zero; equivalently, the confirming round is counted.* (ii) *Six proposals do not suffice on five items, and eight do; so an exact value is a two-sided claim and neither side of Theorem 4.2 is vacuous.* (iii) *A duplicate-free list of all the items is recognised as a ranking and a list with a repeat is not.*

Part (i) is the statement that separates $Q(n)$ from the decision-tree depth of Proposition 4.3, and it is machine-checked rather than asserted.

4. The published range: at most six items

Theorem 4.1. *$Q(1) = 1$, $Q(2) = 2$, $Q(3) = 3$ and $Q(4) = 5$.*

Theorem 4.2. *$Q(5) = 7$ and $Q(6) = 10$.*

Proof of Theorems 4.1 and 4.2. Each value is a pair of bounds. The upper bound is a strategy table checked by Proposition 3.1, the lower bound an adversary table checked by Proposition 3.2, and in both cases the table's first entry carries the empty constraint mask and the stated budget. For $n \le 4$ the tables are stored with states carried by their labels (75 and 31 entries at $n = 4$); for $n = 5, 6$ they are stored up to isomorphism, at 18 and 12 entries for $n = 5$ and 61 and 13 for $n = 6$, using Proposition 3.3. The tables themselves were produced by an exhaustive search of the recursion (1) over all labelled posets on n points, and their correctness is not assumed: the soundness statements are proved for the format, and the Boolean condition of the format is evaluated on the data inside the proof kernel. □

These values were reproduced from scratch, before the identification with the published table was found, by three independent programs: a version-space search whose state is a bitmask over all $n!$ rankings and which performs the plain minimum–maximum of Definition 2.1 with no poset reformulation; a poset search using the acyclicity test of Section 2; and the same poset search written in C. The three agree at every $n \le 6$, and the first shares neither code nor state representation with the other two, which is what makes it a check on the reformulation rather than on the implementation. All three agree with $h^{(4)}(S_n) + 1$ for $n = 3, \dots, 6$.

The identification with decision trees with proper hypotheses.

Proposition 4.3. *For every $n \ge 2$, $Q(n) = h^{(4)}(S_n) + 1$, where $h^{(4)}(S_n)$ is the minimum depth of a decision tree using proper hypotheses only for the sorting decision table S_n in the sense of* [3, 4].

Proof. The table S_n has the $\binom{n}{2}$ comparisons as attributes and the $n!$ permutations as rows, each row labelled by its own permutation. A hypothesis assigns a value to every attribute, and is proper when the assignment is a row, i.e. a linear order; the answer is a confirmation or a counterexample naming an attribute together with a different value, and since the attributes are two-valued, naming the attribute is naming a misordered pair. That is the game of Definition 2.1.

($\le$) Given a tree of depth h, the learner walks it, proposing the hypothesis at each node it reaches and following the edge labelled by the counterexample it receives. After at most h proposals it stands at a terminal node, where the target is determined; one further proposal has it confirmed, so $Q(n) \le h + 1$.

($\ge$) Given a learner that always finishes within Q proposals, build the tree whose nodes are its proposals and whose edges are the counterexamples. Along any path, after $Q - 1$ answers the surviving set must be a singleton — otherwise the oracle has a further counterexample available and the learner needs a $(Q + 1)$-st proposal — so the tree has depth at most $Q - 1$. □

Proposition 4.3 is proved here in the ordinary way and is *not* part of the formal development: it relates two definitions, one of which (decision trees with hypotheses) is not formalised. What is machine-checked is the numerical consequence at each $n \le 7$ (Theorems 4.1, 4.2 and 5.2) and the convention that produces the $+1$ (Proposition 3.4(i)). Numerically the identity holds at $n = 3, 4, 5, 6$, where our $3, 5, 7, 10$ meet the published $2, 4, 6, 9$.

Remark 4.4. The same published table gives $h^{(1)}(S_n) = 3, 5, 7, 10$ for $n = 3, \dots, 6$, the minimum depth of an ordinary comparison tree, which is the classical minimum-comparison sorting count $S(n)$; that column is an independent check that the table's decision tables are the ones we think they are.

5. Seven items

Theorem 5.1. *$Q(7) \geq 12$: no learner identifies an unknown ranking of seven items within eleven proposals.*

Theorem 5.2. *$Q(7) = 12$.*

Proof. The lower bound is Theorem 5.1, whose certificate is an adversary table in the sense of Proposition 3.2 with 36 entries stored up to isomorphism; the first entry carries the empty constraint mask and the budget 12. The upper bound is a strategy table in the sense of Proposition 3.1 with 229 entries stored up to isomorphism, whose first entry likewise carries the empty mask and the budget 12; the same strategy needs about half a million entries when states are carried by their labels, and it is Proposition 3.3 that licenses the reduction. Both tables are checked entry by entry inside the proof kernel; the soundness propositions do the rest. □

Corollary 5.3. *$Q(n) = \lfloor \log_2 n! \rfloor + 1$ for every $n \leq 6$, and this fails at $n = 7$, where $\lfloor \log_2 7! \rfloor + 1 = 13$ and $Q(7) = 12$. Moreover $Q(7) = 12 < 13 = S(7)$, whereas $Q(n) = S(n)$ for $3 \leq n \leq 6$: eleven counterexamples together with a confirming round identify a ranking of seven items, while thirteen comparisons are needed to sort them.*

Proof. The first sentence is Theorems 4.1, 4.2, 5.2 together with the arithmetic of $\lfloor \log_2 n! \rfloor$. The second uses in addition the classical values $S(3), \ldots, S(7) = 3, 5, 7, 10, 13$ [8, 10], of which $S(n)$ for $n \leq 6$ also appears as the $h^{(1)}$ column of [3, Table 1]. The machine-checked input is the list of values $Q(n)$; the two comparisons are arithmetic and a citation, as Section 8 records. □

What the search was, and what the theorem rests on. The value 12 was found by exhaustive search, and the search is what we describe here; the *theorem* rests on the certificates alone, which are re-verified from scratch inside the proof kernel, so the correctness of no search program is used anywhere.

The search evaluates the recursion (1) with the acyclicity test of Section 2, keeping one memo entry per labelled poset on seven points. There are 6,129,859 of those [10, A001035]. In C, with 64-bit poset keys and an open-addressing memo, this takes minutes and 0.30 GB, and the same search in Python roughly ten times as long; the value was recomputed with two different hash-table sizes and reproduced. Both implementations agree with the version-space search at every $n \leq 6$, hence with $h^{(4)}(S_n) + 1$. The certificate generator walks the same search and emits the two tables, canonicalising each state so that one entry stands for its whole isomorphism class.

Remark 5.4. The adversary certificate at $n = 7$ has a striking shape, visible once labels are restored: it is a complete binary tree with $2^{12} - 1 = 4{,}095$ nodes. At every node of budget m there is a pair $\{u, v\}$ such that *both* answers — "the target really has u before v" and the reverse — lead to a position from which the adversary still forces $m - 1$ further rounds; neither answer collapses the value. This is exact halving, twelve times over, where Kahn and Saks [7] guarantee only a 3/11 fraction and the 1/3–2/3 conjecture would give 1/3. It is not a contradiction: those statements are about some pair being balanced in the count of linear extensions, this one about some pair being balanced in the game value. The same shape appears in the adversary certificates at every $n \leq 7$, with $2^{Q(n)} - 1$ nodes.

6. Starting from a poset: the extension count and the width do not determine the exact value

The source's open-questions section also poses the game started from partial information. We quote the relevant part.

> **Starting from partial information.** … the known comparisons form a poset P, and the unknown target ranking is one of its L linear extensions. … It would be interesting to characterize, up to constant factors, the optimal query complexity for

every starting poset P and every k. In particular, it is not clear whether this complexity is determined by L and x alone, or whether additional structural parameters of the poset are needed.

Here x is the width of P, the largest size of an antichain in it. Our certificate machinery accepts an arbitrary starting mask, not only the empty one, so exact values $Q(P)$ are within reach at small size.

Theorem 6.1. *The exact value is not a function of (L, x). Let*

$$P_a = \{\, 0<1,\ 0<2,\ 0<3,\ 0<4,\ 0<5,\ 1<2,\ 1<3,\ 1<4,\ 2<3,\ 5<3,\ 5<4 \,\},$$
$$P_b = \{\, 0<1,\ 0<2,\ 0<3,\ 0<4,\ 1<2,\ 1<3,\ 1<4,\ 2<3,\ 2<4,\ 5<3,\ 5<4 \,\}$$

be posets on the six points $0, \dots, 5$ (the lists are the transitive closures). Both have 8 linear extensions and width 2, and

$$Q(P_a) = 3, \qquad Q(P_b) = 4.$$

In particular the learner wins from P_a within three proposals and does not win from P_b within three.

Proof. Each value is again two-sided: a strategy table of 5 entries and an adversary table of 3 for P_a, a strategy table of 9 entries and an adversary table of 5 for P_b, all up to isomorphism, each with the relevant starting mask in its first entry, all checked entry by entry inside the proof kernel against Propositions 3.1 and 3.2. The parameters $L = 8$ and $x = 2$ are a linear-extension count and a scan of the 2^6 subsets for antichains; they are computed rather than formalised, being an arithmetic anyone can redo in seconds. □

Remark 6.2. This does *not* settle the source's question, which asks for a characterisation *up to constant factors*, and a difference of one proposal is invisible at that resolution. What it settles is that no formula in L and x gives the exact value.

The pair was found by an exhaustive census: all 130,023 labelled posets on six points were grouped by (L, x) and the exact value computed for each. Exactly four of the resulting classes carry two different values — those with $(L, x) = (8, 2)$, $(9, 2)$, $(18, 3)$ and $(36, 3)$ — and none does at $n \le 5$, where all 10 classes at $n = 4$ and all 27 at $n = 5$ are single-valued. In every two-valued class found, the two values differ by exactly one, which is why Remark 6.2 is not a formality: a family of pairs with a growing ratio, not a fixed gap, is what the source's form of the question needs.

7. Two computations outside the formal development

The two statements in this section were computed and are *not* machine-checked; we mark them as such and state exactly what was run.

Remark 7.1 (Properness is free through seven items). Drop the requirement that a query be a linear order, so that the learner may propose an arbitrary orientation of the pairs — the *type-2* hypotheses of [3] — and keep everything else. Running the search of Section 5 with the acyclicity requirement on the proposal removed reproduces the published $h^{(2)}(S_n) = 2, 4, 6, 9$ for $n = 3, \dots, 6$ [3, Table 1], and returns $11 = Q(7) - 1$ at $n = 7$, where no published value exists. So the improper value at seven items is also 12: in the truthful game, properness costs nothing up to seven items. This sits beside the source's headline that properness is exactly what turns the cost of an untruthful counterexample from $O(1)$ into $\Theta(n)$. One implementation, validated against the published column at $n \le 6$; formalising it would need a second model in which a proposal is an arbitrary orientation rather than a ranking, which we have not built.

Remark 7.2 (First exact values with lies). Let $Q(n, k)$ be the value of the same game when at most k of the returned counterexamples may be untruthful and the learner knows k — a lower bound for the parameter-free learner of the source, which does not know k. The state is now a vector of lie counts, one per ranking, and a returned pair costs a lie to every ranking that *agrees* with the proposal on that pair. Computed values:

$Q(n,k)$	$k=0$	$k=1$	$k=2$	$k=3$
$n=2$	2	4	6	—
$n=3$	3	6	9	12
$n=4$	5	8	12	—
$n=5$	7	10	—	—

So $Q(3,k) = 3(k+1)$ over the range computed, and the per-lie increment is 2 at $n=2$, 3 at $n=3$, and 3 then 4 at $n=4$ — data on the second of the two constants the source leaves open, the one in front of nk, consistent with a slope approaching n but not pinning it. Neither source computes any exact $Q(n,k)$. The caveat is real: the $k \geq 1$ column rests on one program, run in two modes that visit different state sets — proposals restricted to rankings still consistent with the history, and all $n!$ proposals allowed with a rule that stalls the adversary when no progress is made — which agree in every cell computed. Its $k=0$ column reproduces the three independent programs of Section 4, and $Q(2,1)=4$ is checkable by hand. The lie model is not formalised, so nothing in this remark is machine-checked.

8. Formal verification

Theorems 4.1, 4.2, 5.1, 5.2 and 6.1 and Propositions 3.1, 3.2, 3.3 and 3.4 are formalised and machine-checked in Lean 4 [9]. The exceptions, stated as such above, are: Proposition 4.3 (a statement about two definitions, one of which is not formalised); in Corollary 5.3 everything except the values $Q(n)$ themselves — the comparisons with $\lfloor \log_2 n! \rfloor + 1$ and with $S(n)$ are arithmetic and a citation, not formal statements; the parameters L and x of Theorem 6.1 and the census that found the pair; the table sizes and search costs quoted throughout; the census counts and the halving observation of Section 9; and Remarks 4.4, 5.4, 7.1 and 7.2. The development is eleven modules and 1,729 lines, declaring 94 theorems and 97 definitions, 58 of the latter being the certificate table data itself; it imports no mathematical library, only the Lean core, which is why the whole development elaborates in a couple of minutes and never exceeds 1.7 GB. The reasoning layer is free of compiled evaluation: the model, both soundness propositions and relabelling invariance are ordinary proofs whose axiom sets were printed and are contained in the standard triple — propositional extensionality, quotient soundness and choice. Compiled evaluation (`native_decide`, which adds one native-evaluation axiom per Boolean check) is used only to evaluate the checker of a certificate format on the certificate data, and for the three side conditions of each bound ($0 < |T|$, the first entry's mask is the intended starting mask, and its budget is the claimed value); an audit of the axiom set of every displayed statement confirms that these are the only non-standard axioms, and that no proof is incomplete. A reader who does not wish to trust compiled evaluation should read the theorems as: *the strategy and adversary tables published with the development certify the stated bounds, given that the tables satisfy their format's Boolean condition*, the second half being a finite check on explicit data. The repository is private: repository reference to be added at submission.

9. The frontier

Question 9.1. What is $Q(8)$?

The certificate side is no longer the obstacle: by Proposition 3.3 the strategy table is bounded by the number of unlabelled posets on eight points, 16,999 [10, A000112], and in practice by far less — at $n=7$ the walk needed 229 of a possible 2,045. The *search* is the whole bill: one memo entry per labelled poset on eight points, 431,723,379 of them, which by extrapolation from $n=7$ is roughly 9.7 GB and about 40 CPU-minutes. That is at the memory budget we allowed ourselves rather than inside it, and an attempt was stopped after 61 minutes of wall time on a heavily loaded machine with no output. The prediction to beat is $\lfloor \log_2 8! \rfloor + 1 = 16$, and Corollary 5.3 says the truth may well be smaller. If memory is the wall, the obvious rewrite is a search whose states are isomorphism classes, 16,999 instead of 431 million, which needs only the canonical form the certificate generator already computes.

Question 9.2. Does every finite poset that is not a total order have an incomparable pair $\{u, v\}$ such that neither of the two orientations of $\{u, v\}$ drops the game value by more than one?

Question 9.2 is Remark 5.4 asked in general; it holds for every poset on at most seven points (verified in the course of the searches above, and not formalised), and it is why every adversary certificate in this paper is a complete binary tree. The "by more than one" is not a hedge: read with equality in place of it, the statement is already false at $n = 3$, where the poset $\{0 \prec 1\}$ on three points has value 2 and each of its two incomparable pairs has one orientation of value 2 and one of value 1.

A separate computed fact, again for posets on at most seven points only: $Q(P) \le \lfloor \log_2 e(P) \rfloor + 1$ always, with equality failing for 120 of the 4,231 posets on five points, 14,660 of the 130,023 on six and 1,241,493 of the 6,129,859 on seven — and in every one of those cases the true value is *smaller*, never larger. The smallest failures have $e(P) = 8$ and $Q(P) = 3$. Whether the inequality holds for every poset is open; it is the exact form of the $O(\log e(P))$ upper bound the source sketches up to a constant factor, and a proof would fix that constant from above. Corollary 5.3 already shows it is not an equality: at the seven-element antichain the truth is one less. The counts in this paragraph are computations, not machine-checked.

Question 9.3. What is the optimal constant in front of the nk term?

Remark 7.2 is the first data on it and is thin: one implementation, and the largest cell computed is $(n, k) = (5, 1)$. An independent implementation and the cells $Q(5, 2)$ and $Q(4, 3)$ would firm up the apparent slope of one lie costing about n rounds. Formalising anything here needs a Lean model of the lie game — whose state is a vector of lie counts, not a version space — and a certificate format for it.

Finally, Theorem 6.1 answers the exact-value form of the source's question about (L, x), and the up-to-constant-factors form it actually asks is untouched: what is needed is a family of poset pairs sharing (L, x) whose values differ by a growing factor, and the six-point census says such pairs are not common at small size, every two-valued class found there differing by exactly one.

References

[1] N. Alon, S. Moran and S. Moran, *Sorting from Counterexamples*, arXiv:2608.21579v1 (21 August 2026), primary classification cs.LG, cross-listed cs.CC, cs.CG, cs.DS and math.CO. All quotations above are from the source of v1, read on 3 September 2026, when the arXiv record showed v1 as the only version and carried no journal reference. Its model box is quoted in Section 1 and its open questions in Sections 1 and 6.

[2] D. Angluin, *Queries and concept learning*, Machine Learning **2** (1988), no. 4, 319–342; DOI 10.1007/BF00116828. The equivalence-query model that the source names as the closest classical framework, and in which the hypotheses of [3] are proper equivalence queries. Cited for context only; no result here depends on it.

[3] M. Azad, I. Chikalov, S. Hussain and M. Moshkov, *Sorting by Decision Trees with Hypotheses (extended abstract)*, in: 29th International Workshop on Concurrency, Specification and Programming (CS&P 2021), CEUR Workshop Proceedings, vol. 2951, pp. 126–130, https://ceur-ws.org/Vol-2951/paper1.pdf. Its §2 defines proper hypotheses and the depths $h^{(k)}$, and its Table 1 gives $h^{(1)}, \ldots, h^{(5)}(T_n)$ for $n = 3, \ldots, 6$, the type-4 column being $2, 4, 6, 9$ and the type-1 column $3, 5, 7, 10$. The paper writes the sorting decision table T_n; we write S_n, following [4]. Retrieved and read in full on 3 September 2026.

[4] M. Azad, I. Chikalov, S. Hussain, M. Moshkov and B. Zielosko, *Decision Trees with Hypotheses for Recognition of Monotone Boolean Functions and for Sorting*, arXiv:2203.08894v1 (16 March 2022), primary classification cs.CC. Its Table 6 gives $h^{(1)}, \ldots, h^{(5)}(S_n)$ for $n = 3, \ldots, 6$ with the same entries as [3, Table 1]; its Table 1 concerns a different decision table (recognition of monotone Boolean functions) and is not the sorting table. Read on 3 September 2026; the arXiv record showed v1 as the only version, with no journal reference and no DOI. Its introduction states its relation to [3]: "This paper is an essential extension of the conference paper […]: for the problem of sorting, we added results related to the use of greedy algorithm. The problem of recognition of monotone Boolean functions was not considered in […]." Note that the author list is not the same — B. Zielosko is a fifth author here — and that the numbering of the sorting table differs between the two papers.

[5] M. Azad, I. Chikalov, S. Hussain and M. Moshkov, *Optimization of Decision Trees with Hypotheses for Knowledge Representation*, Electronics **10** (2021), no. 13, article 1580; DOI 10.3390/electronics10131580. Named by [3] as the reference carrying the complete definitions of the notions used there. Cited for that reason only.

[6] B. Grünbaum, *Partitions of mass-distributions and of convex bodies by hyperplanes*, Pacific J. Math. **10** (1960), 1257–1261; DOI 10.2140/pjm.1960.10.1257. One of the two constants the source's $\Theta(\log L)$ rests on; cited here only as the source cites it.

[7] J. Kahn and M. Saks, *Balancing poset extensions*, Order **1** (1984), 113–126; DOI 10.1007/BF00565647. The 3/11 balanced-pair theorem, which the source uses for its $\Omega(\log L)$ lower bound and which is the published bound our Remark 5.4 should be read against.

[8] D. E. Knuth, *The Art of Computer Programming, Volume 3: Sorting and Searching*, second edition, Addison–Wesley, 1998, §5.3.1 (minimum-comparison sorting). The source of the classical values $S(n)$ used in Corollary 5.3; [10] cites the same edition and section for the same sequence. We give the section, not a page range.

[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, Springer, 2021, pp. 625–635; DOI 10.1007/978-3-030-79876-5_37.

[10] The On-Line Encyclopedia of Integer Sequences, https://oeis.org, entries A001035 (posets with n labelled elements), A000112 (posets with n unlabelled elements), A036604 (sorting numbers: minimal number of comparisons needed to sort n elements) and A003070 ($\lceil \log_2 n! \rceil$); all consulted on 3 September 2026. A search for the sequence $1, 2, 3, 5, 7, 10, 12$ of Corollary 5.3 returns twenty-one entries on that date, none of which is this quantity; a reader running that search should know that two of them, A119565 and A119592, defined by the unrelated recursions $a(n) = \lfloor a(n-1) + 1 + a(n-2)/6 \rfloor$ and $a(n) = \lfloor a(n-1) + 1 + (a(n-2)+1)/6 \rfloor$ with the same initial values $1, 2, 3, 5, 7, 10$, both agree with $Q(n)$ on all seven terms and then continue $14, 17, 20$.