An Optimal Deterministic Adversary for Sorting

Nicholas Nash
ORCID: 0009-0003-0475-7852

7 November 2026

Abstract

This paper describes an optimal deterministic, polynomial-time adversary for ordinary sorting. For any correct sorting algorithm, the adversary forces that algorithm to make $(1 - o(1))n \log_2 n$ comparisons. The best previously known adversary forced $\frac{37}{64} n \log_2 n$ comparisons. The adversary recursively moves items into ordered subproblems, extending the classical framework in which items move down a binary tree.

hexagon:2611.00196v1

Methodological disclosure

Except for the abstract, the text of this paper is AI-generated using GPT 6 Astra. The author is working on a clear, human-readable exposition.

The organizing idea Keep comparisons inside a group for as long as they remain inexpensive to handle. When necessary, fix a position in that group and continue the unresolved order in a child.

The paper first specifies the algorithm and its numerical subroutines. Sections 3–5 establish legality and the refinement bound; Sections 6–8 give the repairs and polynomial-time implementation. The appendices prove the numerical subroutines independently of sorting.

1 From Brodal's tree to unfinished comparisons

There are n distinct labelled keys. A sorting algorithm asks which of two keys is smaller. Its next question can depend on previous answers. An adversary answers the questions, always consistently with some strict total order. A sorter correct on every input can stop only when its comparison transcript determines that order uniquely.

Throughout, $\log = \log_2$, while ln denotes the natural logarithm. The cases $n \leq 1$ are immediate; assume $n \geq 2$, and put

$$T = n \log n.$$

The objective is an efficient deterministic adversary that forces almost T comparisons.

1.1 Brodal: a comparison makes a spatial commitment

The binary-tree adversary described by Brodal, Chaudhuri, and Radhakrishnan [1], and reviewed by Kaplan, Zamir, and Zwick [2], keeps items at nodes of a binary tree. In the basic same-node

case, comparing A and B sends one to the left child and the other to the right. Dyadic intervals give an equivalent picture: halve the common interval and assign opposite halves.

After answering $A < B$ and $C < D$ at the same node, this rule can place A, C on the left and B, D on the right. That representation also commits to the cross-relations $A < D$ and $C < B$. These extra commitments are the cost of a simple representation.

There is an elegant coding proof of its bound. Once sorting is complete, the occupied nodes are distinct leaves: an occupied ancestor and descendant, or two items at the same node, would leave an unresolved pair. The tree's partial order contains all answered comparisons, so such an incomparable pair admits total orders in both directions consistent with the transcript. The root-to-leaf binary addresses are therefore *prefix-free*: no address begins with another complete address.

If their lengths are $d_1, \dots, d_n$, the corresponding dyadic subintervals are disjoint, so

$$\sum_{i=1}^{n} 2^{-d_i} \le 1.$$

This is Kraft's inequality. Convexity of 2^{-x} gives

$$2^{-\frac{1}{n}\sum_i d_i} \le \frac{1}{n}\sum_i 2^{-d_i} \le \frac{1}{n}, \qquad \text{hence} \qquad \sum_i d_i \ge n \log n.$$

Each query increases total depth by at most two. Thus at least $\frac{1}{2} n \log n$ queries are needed. The tree and its prefix code make the proof transparent.

1.2 Kaplan–Zamir–Zwick: retain a pair before separating it

Kaplan, Zamir, and Zwick (KZZ) already postpone some spatial commitments. Their nodes can retain ordered pairs, together with carefully specified interval and intermediate-node states. Thus $A < B$ and $C < D$ can remain as two pairs without immediately imposing the cross-relations. Their rules eventually resolve interactions between such structures by moving items; their best adversary gives the coefficient $37/64$ [2].

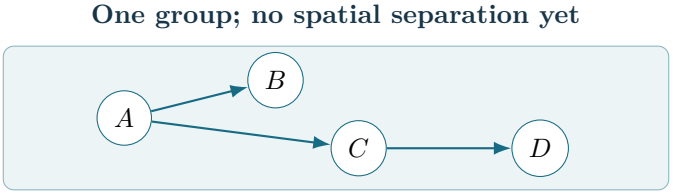
The idea of going further is also present in their paper. Section 6 discusses continuous intervals and richer dependent structures, and asks whether maintaining suitable small dependencies could yield an efficient adversary with coefficient tending to one. The present construction follows that program. Its task is to make more general retained dependencies both computationally manageable and inexpensive to simplify.

1.3 The next step: keep an evolving graph inside a node

After

$$A < B, \qquad C < D, \qquad A < C,$$

we can retain all four items in one node, with just the indicated constraints and their consequences:



Each item has a tentative real position. An arrow $A \to B$ means $X_A < X_B$. The adversary chooses answers by estimating what fraction of the allowed position assignments survives. When the graph becomes difficult to handle, it fixes one item's position and moves that item's unresolved order into a child.

The structure can retain more than a pair, but it does not retain arbitrary complexity indefinitely. Bounded degree, bounded directed-path length, and controlled interval widths are the conditions that make its calculations local.

Theorem 1.1 (Main result)**.** *There is one deterministic legal adversary, initialized using only n, with a fixed polynomial bound on the number of bit operations per comparison, that forces every correct comparison sorter to make*

$$(1 - o(1))n \log n$$

comparisons. The term $o(1)$ tends to zero as $n \to \infty$.

Merge sort uses at most $n \lceil \log n \rceil$ comparisons, so the leading coefficient is optimal. The construction below has very large asymptotic thresholds; this is an existence result, not a claim of practical performance.

2 The algorithm before its proofs

2.1 What one node stores

A node is a group of items whose relative order is still being resolved there. Some items have a free real coordinate X_x in an open interval (a_x, b_x). Directed arrows between free items impose $X_x < X_y$. Other items have already been assigned a fixed parent position and delegated to a child at that position. Call that position the child's *anchor*. Each child has its own fresh coordinate system.

Every original item has exactly one free coordinate, at exactly one node. Its appearances at ancestors are fixed anchors, not additional free coordinates. Different children of one parent have distinct anchors.

At a node v, the intervals and arrows define

$$K_v = \{X : \ a_x < X_x < b_x \text{ for all free } x, \quad X_u < X_w \text{ for every arrow } u \to w\}.$$

Write $V_v = \mathrm{vol}(K_v)$. This is the volume of a constrained region, not generally the product of interval lengths. For example, two coordinates in $(0, 2)$ give a square of area 4; the constraint $X_A < X_B$ leaves a triangle of area 2.

Our auxiliary quantity is

$$Z = \prod_v V_v. \tag{1}$$

Different nodes have separate coordinates, so their volumes multiply. A node with no free coordinates contributes 1. Initially there is only the root, all n intervals are $(0, n)$, and $Z = n^n$. The adversary does not need to calculate the global product.

2.2 An answer restricts one node

To compare two original items, descend while they belong to the same child. At the first node where their representations differ, each is either a free coordinate or a child anchor. Different anchors decide the answer directly. Otherwise at least one coordinate is free.

For example, if the representations are X_A and X_B, consider

$$p = \mathbb{P}(X_A < X_B) = \frac{\mathrm{vol}\{X \in K_v : X_A < X_B\}}{V_v}.$$

The probability notation means a ratio of volumes. It introduces no random choice into the algorithm. Equality has zero volume, so the reverse answer has probability $1 - p$.

An estimate $\widehat{p}$ with $|\widehat{p} - p| \le \eta < 1/4$ suffices. Answer $A < B$ if $\widehat{p} \ge 1/2$, and $B < A$ otherwise. The chosen region has at least $1/2 - \eta$ of the old volume. When one representation is an anchor c, use the same rule with a threshold event $X_A < c$.

Subroutine 1: estimate On a suitably maintained graph, return a deterministic additive-η estimate of a comparison or threshold probability. Section 4 states the precise conditions; Appendices B and C prove the procedure.

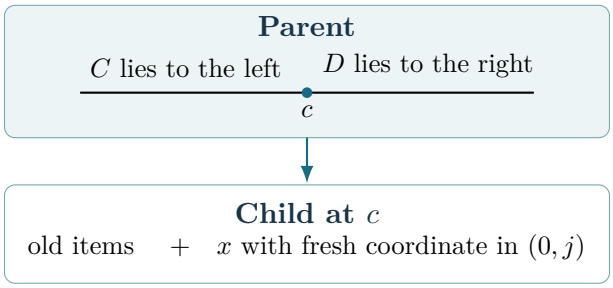
2.3 A refinement fixes a position and renews its coordinate

Suppose we choose an interior feasible position c for a free item x. Fix $X_x = c$, remove x from the parent's free graph, and turn its incident arrows into interval restrictions on their other endpoints. For instance,

$$X_C < X_x < X_D \quad \longrightarrow \quad X_C < c < X_D.$$

The strict comparisons remain true.

Then admit x to the child anchored at c, creating that child if necessary. If this is the child's j-th admission, give x a fresh, independent coordinate in $(0, j)$. Leave every old child coordinate, interval, arrow, and anchor unchanged.



Here j counts *all admissions*, including items that later move to grandchildren. If $f_x(c)$ is the density of the parent coordinate at c, this operation satisfies the exact identity

$$\boxed{\frac{Z_{\text{after}}}{Z_{\text{before}}} = j f_x(c).} \tag{2}$$

The parent loses one dimension by taking a section; the child gains one dimension of length j. Thus total dimension remains n.

For intuition, in the triangle $0 < X_A < X_B < 2$, the section at $X_A = c$ has length $2 - c$, so $f_A(c) = (2-c)/2$. If a child at position 1 already contains one item, fixing A there and admitting it as the second item gives factor $2 \cdot (1/2) = 1$: the child's new freedom exactly pays for the parent restriction.

Subroutine 2: choose a position Find an interval on which the item's density has a guaranteed lower bound, and select a prescribed dyadic midpoint inside it. The ordinary choice needs the local estimate; an emergency central-interval choice works before the width conditions are restored. The guarantees are stated in Section 4.

2.4 The two procedures

All choices and ties use fixed label order. The top-level algorithm is:

```
COMPARE(A, B):
  Descend through common children to the responsible node.
  If both representations are different anchors, return their order.
  Estimate one direction; choose the estimated larger side.
  Store its arrow or interval bound.
  RESTORE the conditions needed by the numerical subroutines.
  Return the chosen answer.
```

Its central private operation is:

```
REFINE(x, node, reason):
  Obtain a good interval by the method appropriate to the reason.
  Choose its prescribed dyadic midpoint c.
  Fix x at c; replace incident arrows by permanent interval bounds.
  Admit x as an isolated free coordinate to the child at c.
```

Subroutine 3: restore Cut a cheap problematic edge, or refine selected troublesome vertices. Restore the numerical hypotheses, preserve all public answers, and perform sufficiently few refinements. Section 6 specifies every rule and proves these promises.

The remainder discharges these contracts. Legality comes first. Then we prove that rare refinements are affordable *before* specifying which ones the algorithm makes.

3 Why the representation is legal

3.1 Reading an actual order

Choose a feasible point at every node, avoiding equalities between free coordinates and between a free coordinate and an anchor. These exceptional equalities have volume zero. Read positions from left to right; replace each child anchor by the order read recursively inside that child.

The invariant is:

> Every order represented in this way respects every comparison already answered.

We do not require the converse. Private repairs may discard compatible orders. Their cost is accounted for through Z. A positive-volume state always represents at least one order, so the invariant proves legality.

An actual comparison restricts one node to a positive-volume side. It therefore preserves the invariant and imposes the requested order. An answer contradicting an earlier implied relation would have probability zero, and cannot be chosen by an additive-η majority estimate with $\eta < 1/4$. No separate public-order safety test is needed.

3.2 Effective intervals

Propagate lower bounds along arrows and upper bounds against them. Writing $y \preceq x$ when $y = x$ or a directed path leads from y to x, define

$$\ell_x = \max_{y \preceq x} a_y, \qquad r_x = \min_{y \succeq x} b_y, \qquad W_x = r_x - \ell_x.$$

The interval (ℓ_x, r_x) is the coordinate's *effective interval*. Store these tightened bounds permanently, including after arrows are removed. Along $x \to y$, both $\ell_x \le \ell_y$ and $r_x \le r_y$.

Lemma 3.1 (Interior fixing)**.** *For an acyclic arrow graph, the feasible region has positive volume if and only if every $W_x > 0$. Every $c \in (\ell_x, r_x)$ is feasible for X_x, and fixing it leaves a positive-volume section in the remaining coordinates.*

Proof. Necessity is immediate. For sufficiency, choose numbers $0 < \theta_x < 1$ increasing in a topological order, a listing in which every arrow points forward, and put

$$X_x = (1 - \theta_x)\ell_x + \theta_x r_x.$$

This expression increases with either endpoint and strictly with θ_x when $W_x > 0$. All arrows are therefore strict, and a small open neighborhood is feasible. A topological order exists by repeatedly removing a vertex with no incoming arrow.

After fixing x at interior c, only predecessors get smaller upper bounds and only successors get larger lower bounds. A predecessor y still has upper bound at least $\min(r_y, c) > \ell_y$, since $\ell_y \le \ell_x < c$. The successor argument is symmetric. Apply the same positive-volume criterion to the remaining coordinates. □

Lemma 3.2 (Legal admission and the refinement identity)**.** *An item fixed at an interior position c is incomparable in the public transcript with every existing item in the child at c. It may enter that child as an isolated coordinate, and (2) holds.*

Proof. Choose a point in the positive-volume parent section $X_x = c$ at which every other free coordinate is distinct from c and from all anchors. Move x a little to the left, or a little to the right, keeping everything else fixed. Strict inequalities permit both changes. The two represented orders put x immediately before or immediately after the entire old child block. Both respect the transcript. Thus no old item in that child has a publicly determined relation to x.

Any new position of x among the old child items is legal: comparisons among old labels are unchanged, there were no answered comparisons between x and those child labels, and x's relations to all outside blocks are the same as in the two surrounding orders.

Let $S_x(c)$ be the parent section volume. By the definition of coordinate density, $f_x(c) = S_x(c)/V_v$. Fixing replaces V_v by $S_x(c)$; the new independent child coordinate multiplies its volume by j. This proves (2). □

A real point has probability zero, but its section has a well-defined lower-dimensional volume. The identity compares that section with the original volume and then restores the missing dimension. A density can exceed one; a refinement can increase Z. These gains must not be discarded in the accounting.

3.3 A fixed rule for the anchor

Suppose a closed interval $I = [a, a + m]$ has density at least $1/(\beta m)$ throughout. Choose the unique power of two b such that

$$\frac{m}{4} < b \le \frac{m}{2}, \qquad k = \lceil a/b \rceil, \qquad c = (k + \tfrac{1}{2})b.$$

The cell $[kb, (k+1)b]$ lies inside I; its midpoint is strictly inside I. Consequently

$$f_x(c) \ge \frac{1}{4\beta b}. \tag{3}$$

Finding the scale uses the binary length of a rational number, not a search through arbitrarily fine grids.

One midpoint determines one cell and one scale: equality of $(2k+1)2^{z-1}$ at two different powers 2^z would equate an odd integer with an even one. This uniqueness lets repeated choices reuse the same child, and lets us count all possible destinations.

4 A numerical theorem independent of sorting

This section's objects are just an acyclic directed graph and intervals. It has no children, refinements, or sorting transcript. For the algorithms, endpoints and queried thresholds are dyadic rationals; the geometric facts apply also to arbitrary real endpoints.

The graph's *degree* is the largest number of neighbors, ignoring arrow directions. Its *height* is the maximum number of vertices on a directed path. Components and distances also ignore directions. Two vertices are comparable if a directed path joins them in either direction.

Theorem 4.1 (Local interval probabilities)**.** *Fix integers $d, h \geq 1$, a rational ratio $R \geq 1$, and rational $0 < \varepsilon < 1/4$. There is a computable radius $\rho = \rho(d, h, R, \varepsilon)$ with the following property.*

Let K be a positive-volume interval-and-arrow region whose graph has degree at most d and height at most h. In every component, suppose the largest effective width is at most R times the smallest. For one or two queried coordinates, retain the induced graph within distance ρ, together with the original effective intervals.

The resulting local probabilities approximate $\mathbb{P}(X_x < c)$ and $\mathbb{P}(X_x < X_y)$ to additive error at most ε. The same local region works for every threshold c. It has at most

$$b \leq 2(d+1)^{\rho+1}$$

vertices. Its probabilities are calculated exactly using $O(b^3 2^b)$ arithmetic operations on rational numbers. No minimum positive width is required.

The radius is independent of the total graph size. The local calculation is exact; using it in place of the full probability introduces the error. Appendix B proves locality and gives explicit constants. Appendix C gives the calculation.

Lemma 4.2 (Two density windows)**.** *Suppose x has at most B other comparable vertices and effective interval (ℓ_x, r_x) of width W.*

1. *Its density satisfies $f_x(t) \leq (B+1)/W$. On*

$$I_x = [\ell_x + W/4,\ r_x - W/4]$$

   it satisfies $f_x(t) \geq 4^{-B}/W$. This window needs no probability calculation.
2. *Given local cumulative probabilities accurate to $1/32$, one can find dyadic $a < b$, strictly inside the effective interval, with $m = b - a$ and*

$$f_x(t) \geq \frac{1}{16m} \qquad (a \leq t \leq b).$$

   If the local region contains b_0 vertices, each endpoint requires at most $\lceil \log(32b_0) \rceil + 1$ bisections.

Appendix A proves this lemma. Applied to the rounding rule, the quantile window gives

$$-\log f_x(c) \leq \log(64b).$$

The central window gives $W/8 < b \le W/4$, and hence

$$-\log f_x(c) \le \log(64b) + 2B.$$

The central window is more expensive, but it is available even when width ratios prevent a local calculation. This avoids a circular repair rule.

Relation to linear extensions. When all intervals are $(0,1)$, each coordinate order occupies volume $1/m!$, where m is the graph size. Thus the region's volume is $e(G)/m!$, and comparison probabilities agree with those under a uniformly random linear extension [3]. With unequal intervals, different orders can have different volumes. The theorem concerns those interval-constrained probabilities. It is not a general deterministic counting theorem for linear extensions, nor a claim of a fully polynomial approximation scheme.

5 Why rare refinements are affordable: a coding proof

We now set aside the implementation of repairs. Consider any completed legal run using q comparisons and P refinements. Index the refinement events by $i = 1, \dots, P$; one item can be refined several times. If event i fixes x at c in a cell of length b, assume

$$f_x(c) \ge \frac{1}{64b\,2^{\gamma_i}}, \qquad \gamma_i \ge 0, \qquad \Gamma = \sum_{i=1}^{P} \gamma_i. \tag{4}$$

The ordinary window has $\gamma_i = 0$; the emergency central window has $\gamma_i = 2B$.

Theorem 5.1 (Rare refinements suffice)**.** *Suppose each comparison retains at least $1/2 - \eta$ of Z, or loses nothing, and there are at most q further private cuts, each retaining at least $1 - \eta$, where $0 < \eta \le 1/32$. Then*

$$\boxed{T \le (1 + 10\eta)q + \Gamma + 10P + 3n + 2P \log\left(2 + \frac{T}{P}\right).} \tag{5}$$

For $P = 0$, the last term means zero. In particular,

$$\eta = o(1), \qquad P = o(T), \qquad \Gamma = o(T) \quad \Longrightarrow \quad q \ge (1 - o(1))T.$$

The proof's meaning is that rare refinements have a small *average* cost. It does not require every refinement to be cheap. The coding argument pays for the many possible child destinations; the admission factorial pays for their repeated use.

5.1 The tree's finite budget

Let s_v be the final number of items ever admitted to node v, and r_v the number remaining free there. For a child c of v, write $k_c = s_c$. Every label visits a node at most once. Consequently

$$s_v = r_v + \sum_{c \text{ child of } v} k_c, \qquad \sum_v r_v = n, \qquad \sum_v s_v = n + P, \qquad \sum_{v,c} k_c = P.$$

All coordinates at v lie in $(0, s_v)$: root intervals started in $(0, n)$, and later admissions received $(0, j)$ with $j \le s_v$.

Define the group-size budget

$$\mathcal{B} = \sum_{v,c} k_c \log(s_v/k_c). \tag{6}$$

For one original label visiting nodes of final sizes $n, s_1, \dots, s_*$, its terms telescope to $\log(n/s_*) \le \log n$. Therefore

$$\mathcal{B} \le T. \tag{7}$$

This has a direct coding interpretation. Add a terminal leaf for each free item at its last node. A branch to a child of size k has ideal information cost $\log(s/k)$, and a branch to an individual terminal item costs $\log s$. Every full item path then has cost exactly $\log n$. Thus

$$T = \mathcal{B} + \sum_v r_v \log s_v.$$

These are fractional information costs, not a claim that a binary codeword has noninteger length.

Lemma 5.2 (Distinct destinations spend the budget)**.** *A batch of g refinements at one parent, entering g distinct children, contributes at least $g \log g$ through its own refinement events to $\mathcal{B}$.*

Proof. If the children's final sizes are $k_1, \dots, k_g$, then $\sum_i k_i \le s$. The arithmetic–geometric mean inequality gives $\prod_i k_i \le (s/g)^g$, or $\sum_i \log(s/k_i) \ge g \log g$. □

5.2 Why the raw binary addresses need a header

In Brodal's finished tree, leaf addresses are already prefix-free. Here children are anchored at midpoints of dyadic cells at many scales. Those cells can nest. For example, in a parent span $[0, 8]$, cells $[0, 4]$ and $[0, 2]$ have addresses 0 and 00, and distinct anchors 2 and 1. These raw addresses are not prefix-free.

We must also describe the scale: where the address stops. This can be done with only a logarithmic overhead in the scale number. The following weights express that observation without integer rounding.

For a child with cell length b_c in a parent of final size s, put

$$t_c = \lfloor \log(s/b_c) \rfloor, \qquad \pi_c = \frac{b_c/s}{(t_c+1)(t_c+2)}. \tag{8}$$

There is one dyadic cell length at each scale number t. At that length, at most s/b cells fit in $[0, s]$. Hence

$$\sum_{c:t_c=t} \pi_c \le \frac{1}{(t+1)(t+2)}, \qquad \sum_c \pi_c \le \sum_{t\ge 0} \left(\frac{1}{t+1} - \frac{1}{t+2} \right) = 1.$$

Thus the weights satisfy Kraft's inequality. The ideal description length is

$$-\log \pi_c = \underbrace{\log(s/b_c)}_{\text{location at this scale}} + \underbrace{\log((t_c+1)(t_c+2))}_{\text{scale header}}.$$

For a literal prefix-free code on the finite set of used children, set $\ell_c = \lceil -\log \pi_c \rceil$. Their Kraft sum is at most one. Sort these integers increasingly and place consecutive dyadic intervals of lengths $2^{-\ell_c}$ inside $[0, 1]$. Each new left endpoint is aligned because previous intervals were at least as long; the total length is at most one. Their disjoint binary addresses realize the code. The proof below uses the weights directly and saves the rounding bit. These proof weights depend on final sizes; the online adversary never needs to know them.

5.3 The factorial pays for reuse

At the j-th admission to a child, the refinement identity supplies a factor j. Across its k_c admissions this is $k_c!$, even if that child answers comparisons between arrivals. Therefore the total logarithmic refinement loss L_{ref} satisfies

$$\begin{aligned} L_{\mathrm{ref}} &\le \Gamma + \sum_{v,c} (k_c \log(64 b_c) - \log(k_c!)) \\ &\le \Gamma + 8P + S, \qquad S = \sum_{v,c} k_c \log(b_c/k_c). \end{aligned} \tag{9}$$

Here $k! \ge (k/e)^k$, obtained by comparing $\sum_{j=1}^{k} \ln j$ with $\int_1^k \ln u \, du$, and $6 + \log e < 8$.

Set $w_c = s_v \pi_c = b_c/((t_c+1)(t_c+2))$. Kraft gives

$$W := \sum_{v,c} w_c \le \sum_v s_v = n + P.$$

For $P > 0$, concavity of log, with weights k_c/P, yields

$$\begin{aligned} \sum_{v,c} k_c \log(w_c/k_c) &\le P \log(W/P) \\ &\le P \log(1 + n/P) \le n \log e. \end{aligned} \tag{10}$$

Consequently

$$S \le n \log e + \sum_{v,c} k_c \log((t_c+1)(t_c+2)). \tag{11}$$

If $S < 0$, it already improves our desired upper bound. Otherwise, observe that

$$\sum_{v,c} k_c t_c \le \sum_{v,c} k_c \log(s_v/b_c) = \mathcal{B} - S \le T.$$

The average scale number, over refinement events, is therefore at most T/P. A second use of concavity bounds all scale headers:

$$\sum_{v,c} k_c \log((t_c+1)(t_c+2)) \le 2P \log\left(2 + \frac{\sum k_c t_c}{P}\right) \le 2P \log(2 + T/P).$$

This and (11) cover both signs of S. Substituting into (9) proves

$$L_{\mathrm{ref}} \le \Gamma + 8P + n \log e + 2P \log(2 + T/P). \tag{12}$$

One way to read this proof is $S = \mathcal{B} - \mathcal{A}$, where $\mathcal{A} = \sum k_c \log(s_v/b_c)$ is the binary location information. The cost left after factorial recovery is the excess of group separation information over location information. Kraft bounds that excess by the short scale headers, plus $O(n)$.

5.4 The endpoint is now a one-line estimate

When the transcript determines a total order, the r_v free coordinates at a node must occur in one prescribed order. They all lie in $(0, s_v)$. Thus

$$V_v \le \frac{s_v^{r_v}}{r_v!} \le e^{s_v}, \qquad \boxed{\log Z_{\mathrm{final}} \le (n+P) \log e.} \tag{13}$$

The second inequality uses one term of the exponential series. Multiplication over nodes uses $\sum s_v = n + P$. No multinomial calculation is needed.

Each comparison costs at most $-\log(1/2-\eta)$, and each private cut at most $-\log(1-\eta)$. For $\eta \le 1/32$, their combined cost is at most $(1+10\eta)q$, using $-\ln(1-u) \le 2u$ for $0 \le u \le 1/2$.

Starting at $\log Z = T$, add these costs, (12), and (13). The inequalities $8 + \log e < 10$ and $2\log e < 3$ prove (5). When $P = 0$, the endpoint estimate alone gives the stated weaker bound.

Finally, with $p = P/T$,

$$\frac{2P\log(2+T/P)}{T} = 2p\log(2+1/p) \longrightarrow 0 \quad \text{as } p \downarrow 0.$$

This proves Theorem 5.1 directly. There is no auxiliary parameter to send to zero later.

What the coding argument does, and what it still needs. It explains the cost of choosing and reusing child positions. Our current item paths are not themselves Brodal-style terminal codewords: several free items may remain at an internal node. The endpoint-volume estimate handles that residual freedom. The coding argument also does not calculate comparison probabilities; that is the separate task of the numerical theorem.

6 How to keep the graphs calculable

The numerical theorem asks for three properties: few neighbors, short directed paths, and controlled width ratios. The repair routine has two stages. First repair widths, so the numerical theorem can be used. Then remove excessive degree or height using the ordinary, inexpensive density window.

Fix integers $D, H, J \ge 4$ and $0 < \eta \le 1/32$. Between queries, require degree below D and height below H. One new arrow raises degree by at most one and joins at most two old directed paths. No repair adds an arrow. Thus throughout repairs we may use

$$d = D, \qquad h = 2H, \qquad B = (d+1)^{h+1}, \qquad C = B+1, \qquad \sigma = 2B.$$

The distance-h ball contains at most B vertices, so in particular each vertex has at most B comparable vertices. Set

$$L = \max\{4, \lceil C/\eta \rceil\}, \qquad A = L^{2h+2}, \qquad R = A^J. \tag{14}$$

All propagation permanently stores the tightened intervals. Fixed label order resolves every choice of vertex, edge, or path.

6.1 Width repair: cut an edge when one side is much wider

Suppose an arrow $u \to v$ has $W_u > LW_v$. Impose the stronger bound $X_u < \ell_v$, and remove the arrow. Since $X_v > \ell_v$, every remaining assignment still satisfies $X_u < X_v$.

How much is lost? Along the arrow, $r_u \le r_v$. The discarded part of u's effective interval therefore has length at most W_v. By the density upper bound, its probability is at most

$$\frac{CW_v}{W_u} < \eta.$$

The new interval is positive, because

$$\ell_v - \ell_u = W_u - W_v + (r_v - r_u) > 0.$$

A predecessor whose upper bound decreases still has lower bound at most $\ell_u < \ell_v$; propagation stays feasible.

Symmetrically, when $W_v > LW_u$, impose $X_v > r_u$ and remove the arrow. These are permanent interval bounds. A deleted arrow is replaced by a stronger restriction, not forgotten. Each such cut removes an edge introduced by a comparison, so the entire run has at most q cuts.

6.2 Width repair: separate a batch of very different scales

Suppose no edge cut applies, but one component's largest width exceeds R times its smallest width w. Adjacent widths have ratio at most L. Along a path from a minimum-width vertex to a maximum-width vertex, mark the first vertex reaching width $A^i w$, for $i = 0, \dots, J-1$. The marked widths lie in

$$[A^i w,\ LA^i w].$$

Successive marked widths therefore differ by a factor at least

$$A/L = L^{2h+1}.$$

Any two marked vertices have distance greater than $2h$: a path of length at most $2h$ would bound their width ratio by L^{2h}.

Refine these J vertices in increasing width, using the central interval and without interrupting the batch. Fixing a coordinate changes effective intervals only along directed paths to or from that coordinate, hence within distance h. The chosen vertices' widths are therefore unchanged until they are processed.

Their selected cell lengths belong to $(W/8, W/4]$. The marked widths differ by more than a factor four, so the cell lengths differ. Distinct scales have distinct midpoints. Thus the batch enters J distinct children.

Call these *scale refinements*. Each pays surcharge $\sigma = 2B$. Their advantage is that they require no local probability calculation.

Repeat the two width rules, always giving the edge cut priority, until neither applies. Each cut deletes an edge, and each scale refinement removes a nonisolated vertex. The process terminates. Every remaining component then has width ratio at most R.

6.3 Graph repair: high degree or a long directed path

After width repair, the numerical theorem is available with the safe bounds d, h, R.

If a vertex has degree at least D, refine it using its quantile window. Then repair widths again. This is an *ordinary refinement*, with zero surcharge.

Otherwise, if a directed path has at least H vertices, mark H consecutive vertices. Finish this marked batch before selecting another path:

1. Repair widths before each ordinary refinement.
2. If a marked item remains free, refine it using its quantile window.
3. If an intervening width repair already refined it, skip it.

The marked items are pairwise comparable in the public transcript. Lemma 3.2 therefore forces their departures from this parent into H distinct children, including departures caused by intervening scale repairs.

Continue until degree and height are below their thresholds. No repair adds an edge. A child admission adds only an isolated coordinate, with no change to its old distributions. It therefore requires no further child repair.

Why the order matters Width repair can operate without the local estimator. Once widths are controlled, ordinary refinement may use it. The algorithm never calls the estimator in order to repair the very condition needed to justify that call.

6.4 The complete restoration routine

The pseudocode hides no additional operation:

```
RESTORE(node):
  Repeat:
    Repair widths, giving edge cuts priority over scale batches.
    If a degree is at least D, ordinarily refine that item; continue.
    If there is an H-vertex directed path, finish its marked batch;
      continue.
    Return.
```

The four rules have two different purposes. Edge cuts remove negligible dependencies. Refinements remove a variable while returning freedom in a child. The following count shows why the costly operations are infrequent.

7 Why there cannot be too many refinements

Let N_o and N_s count ordinary and scale refinements. Then

$$P = N_o + N_s, \qquad \Gamma = \sigma N_s.$$

High degree is paid for by already asked comparisons. A degree refinement deletes at least D current edges. Deleted edges cannot be deleted again; every edge was introduced by an actual query at that node. Thus there are at most q/D degree refinements.

Long paths and separated scales spend the tree budget. Each marked path sends H items into different children. By Lemma 5.2, its refinement events contribute at least $H \log H$ to $\mathcal{B}$. Different marked paths at one node use disjoint refinement events, so at most $T/\log H$ ordinary path refinements occur.

Each scale batch similarly contributes at least $J \log J$, and scale batches use disjoint refinement events. Hence

$$\boxed{N_o \le \frac{q}{D} + \frac{T}{\log H}, \qquad N_s \le \frac{T}{\log J}.} \tag{15}$$

A path batch can overlap a scale batch. This causes no double charge in either separate estimate: each family uses its own application of $\mathcal{B} \le T$. A marked item consumed by a scale repair is charged only for the refinement it actually makes.

A rough finite bound for implementation. We also have

$$P \le 2q. \tag{16}$$

Charge each refinement to one incident edge endpoint at the node it leaves. Degree and scale refinements have such an endpoint when selected. For a marked path item, reserve an original incident path endpoint even if another repair removes that edge before the item moves. A label leaves a node only once, so an endpoint cannot be charged twice. There are at most $2q$ introduced endpoints in total.

This last estimate is not the asymptotic rarity argument. It is useful for bounding the amount of state and the number of binary digits created.

8 Choosing the parameters and proving polynomial time

8.1 First choose any fixed accuracy stage

For each integer $k \ge 4$, use

$$D = k + 4, \qquad H = 2^k, \qquad \eta = \frac{1}{100k}.$$

Compute B, σ as above, and set

$$J = 2^{k\sigma}.$$

Then compute L, A, R from (14), and the radius ρ_k supplied by Theorem 4.1. Since $\eta < 1/32$, the same radius is sufficient for the quantile calculations. Let

$$b_k = 2(d+1)^{\rho_k+1}$$

be the corresponding neighborhood-size bound.

On a completed run with $q \le 2T$, (15) gives

$$N_o \le \frac{3T}{k}, \qquad N_s \le \frac{T}{k\sigma}, \qquad P \le \frac{4T}{k}, \qquad \Gamma \le \frac{T}{k}.$$

Thus, as the selected stage tends to infinity, $\eta = o(1)$, $P = o(T)$, and $\Gamma = o(T)$. The coding bound (5) then gives $q \ge (1 - o(1))T$. Runs with $q > 2T$ already exceed the target.

8.2 Make one algorithm, with one polynomial exponent

Allowing the running-time exponent to depend on a desired accuracy would not establish Theorem 1.1. We instead let the stage grow extremely slowly with n.

During initialization, spend at most n bit operations computing successive stages. Keep the largest fully computed stage satisfying

$$b_k \le \lfloor \log(n+1) \rfloor .$$

If none fits, answer using a fixed total order of the labels. Every fixed stage is a finite computation and eventually meets both the initialization and neighborhood budgets. Therefore the stage $k(n)$ selected in this way tends to infinity.

Every local calculation then uses $b \le \log(n+1)$ coordinates. Appendix C uses $O(b^3 2^b)$ arithmetic operations, which is a polynomial with a fixed exponent. Constructing a neighborhood, propagating intervals, and finding paths use ordinary polynomial-time graph operations.

8.3 Binary precision is also polynomial

All endpoints and anchors are dyadic rationals, meaning integers divided by powers of two. Suppose the current denominators divide 2^{Λ}. A quantile search adds only $O(\log(b+1))$ binary digits; the grid rounding adds a constant number. A central-window refinement adds only a constant number. Propagation and cuts use existing bounds and add no digits.

Before $q = O(n \log n)$ queries, (16) therefore gives

$$\Lambda = O\big(n \log(n+1) \log\log(n+4)\big).$$

All coordinates lie in $[0, n]$. The exact local table uses integers with

$$O\big(b(\Lambda + \log(n+1) + \log(b+1))\big)$$

bits. Thus the arithmetic bit cost is polynomial as well. There are at most $P+1$ nodes and at most q edge introductions. Stage descriptions have at most n bits, by the initialization budget.

After $2n\lceil\log(n+1)\rceil$ queries, the adversary may freeze a total order compatible with its saved public comparison graph, found by topological sorting, and answer from it thereafter. This threshold is already beyond the lower bound. It prevents repeated gratuitous queries from enlarging the state indefinitely.

The representation proves legality, the rarity estimates and coding theorem prove the lower bound, and the local computation and precision bounds prove the fixed polynomial running time. This establishes Theorem 1.1.

9 Computational checks and related formalization

The exact local-volume recurrence of Appendix C was compared with an independent rational implementation of the segment-assignment method on 630 seeded acyclic interval systems with one to seven vertices. The two implementations agree in every case, including infeasible systems and mixed dyadic precisions. A further 27 analytic checks cover independent boxes, chains, empty and cyclic cases, an unequal-interval comparison probability of $3/4$, triangle densities, and the refinement identity $jf_x(c)$. The accompanying source includes the script and its results. These finite checks verify the numerical implementation and selected identities; the locality and asymptotic guarantees follow from the proofs in this paper.

A related Lean 4 development is available in the sorting-adversary-proofs repository [4]. That development follows the earlier recursive integer-colour construction. The continuous-position formulation, prefix-code accounting, and exact-volume recurrence presented here are given as mathematical arguments; citing the repository does not assert a line-by-line formalization of this presentation.

A Density windows from elementary geometry

A.1 Sections of a convex region

A region is *convex* if it contains the segment joining any two of its points. Our regions are convex because every defining condition is a linear inequality.

Lemma A.1 (The two-cone bound)**.** *Let a point be uniform in a positive-volume bounded convex region in m dimensions. Suppose a nonconstant linear coordinate Y ranges over (L, U), of width*

$W = U - L$. Its probability density g satisfies

$$g(t) \le \frac{m}{W}. \tag{17}$$

For every $0 < \theta \le 1/2$, it also satisfies

$$g(t) \ge \frac{\theta^{m-1}}{W} \qquad (L + \theta W \le t \le U - \theta W). \tag{18}$$

Proof. Use a linear change of variables making Y the first coordinate. Its constant volume factor cancels in densities. Write $A(t)$ for section volume and $V = \int_L^U A(t)\,dt$ for total volume.

Join the section at t to feasible points with first coordinates approaching L and U. Convexity puts both cones inside the region. A section of a cone shrinks in its $m-1$ free directions by a common factor, so the two cone volumes tend to

$$\frac{t-L}{m}A(t) + \frac{U-t}{m}A(t) = \frac{W}{m}A(t).$$

Thus $V \ge WA(t)/m$, proving (17). For $m = 1$ the distribution is uniform and both assertions are immediate.

There is an interior t_0 with $A(t_0) \ge V/W$, or a sequence approaching this bound. For $t \ge t_0$, join this section toward the upper endpoint. At t, the scale factor tends to $(U-t)/(U-t_0) \ge \theta$. Hence $A(t) \ge \theta^{m-1}A(t_0)$. Use the lower endpoint when $t \le t_0$ and take limits if needed. This proves (18). □

This proof uses only convexity and the integral $\int_0^1 u^{m-1}\,du = 1/m$, not a general theorem about log-concave measures.

A.2 Why the relevant dimension is small

Lemma A.2 (Condition on the incomparable labels)**.** *Let x have c other comparable labels in the active graph and effective interval (ℓ_x, r_x) of width W_x. Then*

$$f_x(t) \le \frac{c+1}{W_x}, \tag{19}$$

$$f_x(t) \ge \frac{4^{-c}}{W_x} \quad (\ell_x + W_x/4 \le t \le r_x - W_x/4). \tag{20}$$

For another coordinate X_y, or a fixed number Y, and any fixed interval I of length a,

$$\mathbb{P}(X_x - Y \in I) \le \frac{(c+1)a}{W_x}. \tag{21}$$

In the coordinate case $Y = X_y$, independence is not assumed.

Proof. Fix all coordinates incomparable with x. For almost every feasible fixing, the remaining region is a positive-volume convex region on x and its c comparable labels, and the conditional law is uniform there.

Crucially, the available interval of x is still (ℓ_x, r_x). A new lower bound from a fixed outside label could reach x only along a directed path from that label to x; a new upper bound would require the reverse path. Both contradict incomparability. This is also seen directly by propagating the new fixed bounds. Apply theorem A.1 in dimension $c+1$, then average over the fixed outside coordinates. This gives (19)–(20).

If y is incomparable with x, it was fixed in this conditioning, so the projection $X_x - X_y$ has width W_x. The same is true for a fixed number Y.

If y is a successor of x, its upper endpoint r_y is unchanged by fixing the outside labels: an outside upper bound propagating to y would also make that outside label a successor of x. Fix y at a feasible value within δ of r_y. By theorem 3.1, x can then range over

$$(\ell_x, \min(r_x, X_y)),$$

of width at least $W_x - \delta$, since $r_y \geq r_x$. Thus the projection $X_x - X_y$ of the conditional region has width at least W_x, on letting $\delta \downarrow 0$. For a predecessor, fix y within δ of its unchanged lower endpoint and argue symmetrically. The upper part of theorem A.1, now for the linear coordinate $X_x - X_y$, bounds its conditional density by $(c+1)/W_x$. Integrate over I and then over the outside coordinates to obtain (21). □

Taking $I = [-z, z]$ gives $\mathbb{P}(|X_x - X_y| \leq z) \leq 2(c+1)z/W_x$. The claim is for a difference of coordinates or a fixed threshold. It does not assert the same bound for an arbitrary interval chosen as an arbitrary function of all other coordinates.

A.3 A useful interval between middle quantiles

We need one shape fact. A positive function f is *log-concave* if

$$f((1-\lambda)a + \lambda b) \geq f(a)^{1-\lambda} f(b)^{\lambda} \quad (0 \leq \lambda \leq 1).$$

It has no downward valley between two values. The following specialized proof keeps the argument self-contained.

Lemma A.3 (Shape of a coordinate density)**.** *A coordinate density in our interval-and-arrow region is positive and continuous on its effective interval and is log-concave there.*

Proof. Let $S(t)$ be its unnormalized section volume at $X_x = t$. Positivity follows from theorem 3.1. Take feasible assignments p, q with $p_x = t - \delta$, $q_x = t + \delta$, and put $h = q - \delta$ in every coordinate. Both p, h satisfy the strict arrows, although h need not satisfy the original intervals.

Join the endpoints of an arrow $u \to v$ in an undirected graph when

$$\max(p_u, h_u) \geq \min(p_v, h_v).$$

On any such edge, $h - p$ is nonzero and has the same sign at both ends. Otherwise strict increase of p or h would give the reverse inequality. At x the sign is positive, so $h > p$ throughout its connected component. Swap p, h on this component, and then add δ back to the second assignment. Both resulting assignments have coordinate t at x.

Within the swapped component, $a_u < p_u < h_u = q_u - \delta < b_u - \delta$, so both outputs obey the original intervals. Across its boundary, either choice at either endpoint obeys the arrow by the definition of the component. Internal arrows are preserved as well.

The map is injective: subtract δ from the second output. The unordered coordinate pairs are unchanged, so they reconstruct the same graph and component and tell us which swap to reverse. On each piece of its domain the map permutes coordinates and adds constants, preserving volume in the two sections. Hence

$$S(t-\delta)S(t+\delta) \leq S(t)^2.$$

Section volume is continuous in the interior: moving t changes feasibility only in finitely many slabs whose widths tend to zero, inside a bounded box. Iterating the midpoint inequality and using continuity gives log-concavity. Normalization by the fixed total volume preserves it. □

Lemma A.4 (Middle mass gives a reliable interval)**.** *Suppose a local cumulative probability $\widehat{F}$ is within $1/32$ of the true cumulative probability F at every threshold. Choose a, b with*

$$\widehat{F}(a) \in [7/32, 9/32], \qquad \widehat{F}(b) \in [23/32, 25/32].$$

Then $a < b$ are strictly interior, and, with $m = b - a$,

$$f_x(t) \geq \frac{1}{16m} \qquad (a \leq t \leq b).$$

Proof. The true cumulative values lie respectively in $[3/16, 5/16]$ and $[11/16, 13/16]$. The local and full supports at x are the same: the local region retains the effective interval and contains every restriction of a full feasible assignment. Thus a, b are interior. The mass between them is at least $3/8$, so some $t \in [a, b]$ has $f_x(t) \geq 3/(8m)$.

Suppose $f_x(a) < 1/(16m)$. Then $t > a$, and the secant slope of $\ln f_x$ from a to t exceeds $\ln 6/m$. Concavity implies, for $u < a$ in the support,

$$f_x(u) \leq f_x(a) \exp(-(a - u) \ln 6/m).$$

Integration gives $F(a) \leq m f_x(a)/\ln 6 < 1/(16 \ln 6) < 3/16$, a contradiction. Reflecting the coordinate proves the same lower bound at b. Log-concavity gives at least the smaller endpoint density throughout $[a, b]$. □

This argument needs no separate log-concavity theorem for cumulative probabilities or tails.

For the stated bisection bound in Lemma 4.2, apply theorem A.1 directly to the b_0-dimensional local region. Its marked density is at most b_0/W_x. Bisect the original effective interval, stopping when the reported cumulative value is within $1/32$ of the target $1/4$ or $3/4$. Until success, bracket endpoints have cumulative difference greater than $1/16$. After r failures, their distance is $W_x/2^r$, so the difference is at most $b_0/2^r$. This is impossible for $r = \lceil \log(32 b_0) \rceil + 1$. Each bisection adds at most one binary fractional digit. This proves Lemma 4.2 in full.

B Why distant constraints have a small effect

We prove Theorem 4.1 independently of the recursive adversary. The proof uses random variables to compare two distributions; the algorithm itself executes only the finite table of Appendix C.

There are two ingredients. First, repeatedly refreshing a coordinate from its permitted interval causes two copies of the system to draw closer. Second, the density bound shows that small coordinate errors rarely change a comparison sign.

B.1 Two stationary experiments with shared random numbers

Start one experiment uniformly in the full feasible region and another uniformly in the retained neighborhood. These are mathematical distributions, not samples that the algorithm has to produce. The two starting points may have any joint coupling.

Visit vertices in a topological order. With all other coordinates fixed, the permitted values of X_x form an interval (a, b), where

$$a = \max\{\ell_x,\ X_y : y \to x\}, \qquad b = \min\{r_x,\ X_z : x \to z\}.$$

Replace X_x by $(1 - U_x)a + U_x b$, where U_x is a fresh uniform number in $(0, 1)$. This is precisely the coordinate's conditional uniform distribution. Resampling from a conditional distribution

preserves the joint uniform law, so each experiment remains in its own original uniform distribution after every update.

Use the same fresh U_x in both experiments at a shared vertex. Restrict the full topological order to the neighborhood. Write Δ_x for the absolute discrepancy between the two copies. If every incident edge is retained, then

$$\Delta_x^{\text{new}} \le (1 - U_x) \max_{y \to x} \Delta_y^{\text{new}} + U_x \max_{x \to z} \Delta_z^{\text{old}}. \tag{22}$$

An empty maximum is zero. Taking a maximum with a common lower bound, or a minimum with a common upper bound, cannot increase a discrepancy.

The predecessor terms refer to new values because the sweep is topological. Expand them backward within that sweep. A predecessor chain contains at most h vertices and eventually reaches a common fixed lower bound, whose discrepancy is zero.

Let A_x be the minimum of the fresh $1 - U$ values at x and all its ancestors. The coefficient reaching the zero-discrepancy boundary is at least A_x^h. The coefficients of all remaining old discrepancies sum to at most $1 - A_x^h$, and those old discrepancies occur within distance h. Thus

$$\boxed{\Delta_x^{\text{new}} \le (1 - A_x^h) \max_{y \in B_h(x)} \Delta_y^{\text{old}}.} \tag{23}$$

Here $B_h(x)$ is the undirected distance-h ball. This expansion can follow whichever predecessor maximizes the discrepancy; A_x bounds all possible choices.

B.2 One moment contracts

Put

$$M = (d+1)^{h+1}, \qquad a = \frac{1}{4M^2}, \qquad p = (4M^2)^h \left\lceil \log(4M) \right\rceil.$$

Both the ball and the set of clocks defining A_x contain at most M vertices. A union bound gives

$$\mathbb{P}(A_x < a) \le Ma = \frac{1}{4M}.$$

On the complementary event,

$$(1 - A_x^h)^p \le (1 - a^h)^p \le e^{-pa^h} \le \frac{1}{4M}.$$

The last step uses $pa^h = \lceil \log_2(4M) \rceil \ge \ln(4M)$. It follows that

$$\mathbb{E}\big[(1 - A_x^h)^p\big] \le \frac{1}{2M}.$$

Crucially, A_x uses only fresh clocks from this sweep; it is independent of the whole vector of old discrepancies. Raise (23) to the integer power p, bound a maximum by a sum, and use that independence:

$$\begin{aligned} \mathbb{E}\big[(\Delta_x^{\text{new}})^p\big] &\le \frac{1}{2M} \sum_{y \in B_h(x)} \mathbb{E}\big[(\Delta_y^{\text{old}})^p\big] \\ &\le \frac{1}{2} \max_{y \in B_h(x)} \mathbb{E}\big[(\Delta_y^{\text{old}})^p\big]. \end{aligned} \tag{24}$$

This is the whole contraction argument. The large power p makes near-failures of contraction sufficiently rare to compensate for the possible neighbors.

B.3 Choose the radius so the boundary cannot interfere

Use the explicit constants

$$\begin{aligned} C = M + 1, \qquad & u = \frac{\varepsilon}{16C}, \\ \lambda = \lceil \log(R/u) \rceil, \qquad & \mu = \lceil \log(8/\varepsilon) \rceil, \\ t = p\lambda + \mu, \qquad & \rho = ht. \end{aligned}$$

These are computable using integer comparisons with powers of two.

For a marked vertex x_*, apply (24) at sweep j only to vertices within distance $h(t-j)$ of x_*. Their required h-balls lie in the preceding interior. The predecessor expansion uses at most $h-1$ predecessor edges followed by one incident edge, so all needed edges are retained. This explains the radius ht.

Both copies of an initially retained coordinate lie in its original effective interval. In the marked vertex's original component, initial discrepancies are therefore at most RW_{x_*}. Induction using (24) gives

$$\mathbb{E}\big[\Delta_{x_*}(t)^p\big] \le 2^{-t}(RW_{x_*})^p.$$

For a nonnegative random variable Y, the inequality $\mathbb{P}(Y > a) \le \mathbb{E}[Y^p]/a^p$ follows by integrating $Y^p \ge a^p$ on that event. Apply it here:

$$\mathbb{P}(\Delta_{x_*}(t) > uW_{x_*}) \le 2^{-t}(R/u)^p \le 2^{-\mu} \le \varepsilon/8. \tag{25}$$

For two marked vertices, use this estimate for each one's component and width. The components may differ; independence of the two error events is unnecessary. An isolated coordinate can simply start at the same uniform value in both experiments.

B.4 From coordinate error to comparison error

Every mark has at most M comparable vertices. The dependent-difference estimate (21) therefore applies with the constant $C = M + 1$.

Outside the two exceptional events in (25), different signs of $X_x - X_y$ in the two experiments require

$$|X_x - X_y| \le u(W_x + W_y) \le 2u \max(W_x, W_y).$$

Use (21) at the endpoint of larger width. The interval for the difference has length at most $4u \max(W_x, W_y)$, so its probability is at most $4Cu = \varepsilon/4$. The two exceptional events have total probability at most $\varepsilon/4$. Thus sign disagreement has probability at most $\varepsilon/2$, which is more than sufficient. A threshold comparison uses one exceptional event and a shorter interval.

Because both experiments preserve their original laws, a bound on disagreement is a bound on the difference of their probabilities. This proves locality.

The local region has positive volume: restricting any full feasible assignment gives a local one. Its marked effective support is unchanged, because the old effective interval is retained explicitly and every value in it occurs in some full assignment. A degree-d radius-ρ neighborhood of two marks contains at most $2(d+1)^{\rho+1}$ vertices. This completes the geometric part of Theorem 4.1.

C Exact local calculation: which coordinate comes last?

The local problem has b coordinates. We give a finite exact calculation that uses just subsets, polynomial integration, and integer arithmetic. It does not first count linear extensions.

C.1 A recurrence for every subset

For each subset S of the local vertices, let $F_S(t)$ be the volume of assignments to S that respect their own intervals, the arrows with both endpoints in S, and $X_x < t$ for all $x \in S$. Set $F_\varnothing(t) = 1$.

For nonempty S, almost every assignment has a unique largest coordinate. If it belongs to v, there can be no arrow $v \to w$ with $w \in S \setminus \{v\}$. Call such a v a *sink of the induced graph on S*. Once $X_v = u$ is the largest value, every incoming arrow to v holds automatically. The remaining assignments have volume $F_{S\setminus\{v\}}(u)$. Therefore

$$F_S(t) = \sum_{\substack{v \in S \\ v \text{ a sink in } S}} \int_{a_v}^{\min(t,b_v)} F_{S\setminus\{v\}}(u)\,du, \tag{26}$$

where a term is zero when $t \le a_v$.

Sort all interval endpoints. On an open segment between consecutive endpoints, the set of intervals containing that segment is fixed. Differentiating (26) on this segment gives

$$\boxed{F_S'(t) = \sum_{\substack{v \text{ a sink in } S \\ (a_v,b_v) \text{ contains the segment}}} F_{S\setminus\{v\}}(t).} \tag{27}$$

This is the whole recurrence. At the smallest endpoint, every nonempty subset has volume zero. On each segment, compute the polynomials in increasing subset size and carry their values to the next segment. Their degrees are at most $|S|$. At the largest endpoint, F_V is the required full volume.

Computing all subsets avoids extra terminology or data structures: no downward-closed-set test and no transitive closure are required. Each subset is simply its own induced-graph problem.

C.2 An all-integer implementation

Scale all dyadic endpoints by 2^Λ, making them integers. At a segment's left endpoint a, store

$$Q_S = |S|!\,F_S(a).$$

These values start as integers. For $r = |S|$, compute coefficients

$$\begin{aligned} C_{S,0} &= Q_S, \\ C_{S,k} &= \sum_{\substack{v \text{ a sink in } S \\ v \text{ accepts this segment}}} C_{S\setminus\{v\},\,k-1}, \qquad 1 \le k \le r. \end{aligned} \tag{28}$$

The invariant is

$$C_{S,k} = (r-k)!\,F_S^{(k)}(a+).$$

It follows inductively from (27). If the segment length is the integer Δ, the update to its right endpoint is

$$\boxed{Q_S^{\text{new}} = \sum_{k=0}^{r} \binom{r}{k} C_{S,k}\Delta^k.} \tag{29}$$

Thus all operations are integer additions and multiplications. At the end, the original unscaled volume is

$$\frac{Q_V}{b!\,2^{\Lambda b}}.$$

There are at most 2^b subsets, b coefficients per subset, b summands per coefficient, and $O(b)$ endpoint segments. Precompute outgoing-neighbor masks to test sinks, binomial coefficients, and the powers of each segment length. The total is $O(b^3 2^b)$ arithmetic operations.

If original endpoints lie in $[0, K]$, put $N = \max\{1, 2^\Lambda K\}$. Expanding the deletion recurrence bounds

$$0 \le C_{S,k} \le r!\, N^{r-k}, \qquad 0 \le Q_S \le r!\, N^r.$$

The entries therefore have

$$O\big(b(\Lambda + \log(K+1) + \log(b+1))\big)$$

bits. This justifies the bit bound used in Section 8.

For a comparison numerator, add its arrow and repeat the table; a directed cycle gives zero. For a threshold numerator, shorten the marked coordinate's upper interval bound and repeat. Use a common coordinate scale for numerator and denominator; it cancels in their ratio. For all thresholds use the same original local region as denominator. The estimates then form one actual cumulative distribution, as required for quantile bisection.

C.3 A small example

With two independent coordinates in $(0, 1)$,

$$F_{\{A\}}(t) = F_{\{B\}}(t) = t, \qquad F'_{\{A,B\}}(t) = 2t,$$

so the area is $F_{\{A,B\}}(1) = 1$. With the arrow $A \to B$, only B can be last:

$$F'_{\{A,B\}}(t) = F_{\{A\}}(t) = t, \qquad F_{\{A,B\}}(1) = \tfrac{1}{2}.$$

The recurrence is the elementary triangle calculation repeated over subsets and interval segments.

References

[1] Gerth Stølting Brodal, Shiva Chaudhuri, and Jaikumar Radhakrishnan. *The randomized complexity of maintaining the minimum.* Nordic Journal of Computing 3(4), 337–351, 1996. The binary adversary and its interval interpretation are reviewed in Section 2 and the introduction of [2].

[2] Haim Kaplan, Or Zamir, and Uri Zwick. *A sort of an adversary.* Proceedings of the Thirtieth Annual ACM–SIAM Symposium on Discrete Algorithms (SODA), 1291–1310, 2019. doi:10.1137/1.9781611975482.79. The discussion of continuous intervals and richer dependencies is in Section 6.

[3] Richard P. Stanley. *Two poset polytopes.* Discrete & Computational Geometry 1, 9–23, 1986. doi:10.1007/BF02187680. Author's copy.

[4] Nicholas Nash. *Sorting adversaries in Lean.* Lean 4 source repository, 2026. `https://github.com/nicknash/sorting-adversary-proofs`. Source snapshot `15d93dd`, accessed 7 November 2026.