Electronic Colloquium on Computational Complexity, Report No. 96 (2020)

On the asymptotic complexity of sorting

Igor S. Sergeev*

Abstract

We investigate the number of pairwise comparisons sufficient to sort n elements chosen from a linearly ordered set. This number is shown to be $\log_2(n!) + o(n)$ thus improving over the previously known upper bounds of the form $\log_2(n!) + \Theta(n)$. The new bound is achieved by the proposed group insertion sorting algorithm.

1 Introduction

To start, we remind that the problem of sorting is thoroughly discussed e.g. in [1, Ch. 3], [9, §5.3], [11, Ch. 2].

The sorting algorithm may be represented as a binary rooted tree with the edges oriented from the root. Such a tree is usually called a comparison tree, as well as a decision tree or a binary decision diagram. An inner vertex of the tree corresponds to the operation of comparing some two elements. Depending on the comparison result, the algorithm moves along one of the edges to the next vertex. A terminal vertex (leaf) of the tree corresponds to the final linear ordering of the input set as a result of comparisons. The *complexity* of the algorithm is the depth of the tree, that is, the maximal distance in edges between the root and a leaf.

We restrict ourselves with algorithms not containing redundant comparisons (which do not add anything about the order of elements). In trees corresponding to such algorithms, any possible ordering of inputs is associated with exactly one leaf. In particular, trees for sorting n-element set have exactly $n!$ leaves.

Let $S(n)$ denote the minimal complexity of an algorithm sorting n inputs. Since the number of all possible permutations of n elements is $n!$, and the depth of a binary tree with m leaves is at least $\log m$, then

$$S(n) \geq \log(n!) \sim n \log n. \tag{1}$$

*Research Institute "Kvant" (Moscow); e-mail: isserg@gmail.com

ISSN 1433-8092

(Here and below, we skip the base of binary logarithms.) Moreover, this bound holds for the average complexity of sorting over all permutations of inputs as well, see also [1, 9, 11]. Such lower bounds are called information-theoretic.

The well-known Ford–Johnson algorithm [5] (the method of binary insertions) proposed 60 years ago provides a rather tight upper bound for the complexity of sorting

$$S(n) \le \log(n!) + cn + O(\log n), \tag{2}$$

where the constant c varies from $\log(3e/8) \approx 0.028$ (in the favourable case $n \sim 2^k/3$) to $\log(3/(4\ln 2)) \approx 0.114$ (in the unfavourable case $n \sim \ln 2 \cdot 2^k/3$) depending on the value of n. The method of binary insertions is provably optimal for $n \le 15$ and some larger values of n, see [12].

One can deduce a slightly improved constant for the unfavourable case from [14]. The authors of [10] made a considerable attempt to reduce a disbalance in values of c. As a result of their work, the value of c for the unfavourable case may be specified as $c \approx 0.07$ (it's hard to be more precise since the strongest form of the result in [10] is expressed graphically). The same constant for average sorting complexity was recently reduced to $c \approx 0.032$, see [4, 8]. Apparently, in the favourable case (involving infinite series of comparatively short intervals around numbers $2^k/3$) the initial method [5] was never beaten even in average.

Essentially, the Ford–Johnson algorithm is a sequence of insertions of an element into a linearly ordered (sub)set. Any insertion is performed via the binary method: first, compare an inserted element with the median element of the target set, then compare it with the median element of the subset determined as a result of the first comparison, etc. Efficiency of the algorithm is secured by the choice of target sets with optimal values of cardinality $2^k - 1$.

It is known, however, that a joint insertion of several elements may be implemented faster than independent insertions. For instance, if the cardinality m of a target set satisfies $2^k < m < \frac{17}{14} \cdot 2^k - 1$, then we can save one comparison against the binary method via processing ordered pairs [6, 7], see also [9]. Pair insertion also helps in average sorting, see [8]. As shown in [14], even more profitable is to split the elements into groups of size 4 or 5 and to sort them prior to insertion.

As a further development of the mentioned idea, we propose an algorithm for the group insertion of a large number of elements organized in a sense as a mass service system. This approach resembles the concept of mass production (of partially ordered sets), which is followed by the best known algorithms for selecting an element of a given order [13, 3].

The algorithm processes ordered pairs of elements in turn. At some point after a series of comparisons, a pair can be broken if, for example, it turns

out that its elements fall into disjoint subsets (intervals) of the target set. Then the elements of the pair are processed separately. When it is beneficial, a single element is inserted into the target interval via the binary method. Otherwise, it is placed into a temporary storage, a *container*. As soon as two elements appear in the container, the algorithm groups them into the ordered pair and processes it further. The algorithm exploits a collection of containers attributed to different intervals. At the very end, the single elements remaining in the containers are inserted into the target set via the binary method. This is the general outline of the method.

Following the strategy above, we can insert p elements into a target set of size $m \gg p$ with the complexity almost matching the information-theoretic lower bound, that is, $p(\log_2 m + o(1))$ comparisons. As a result, a sorting algorithm by group insertions witnesses the complexity bound $S(n) \leq \log_2(n!) + o(n)$. Indeed, the same bound holds for the complexity of sorting in average.

The paper is organized as follows. First, we remind the basic method of binary insertions in §2. Then we mention some simple initial considerations behind the proposed method in §3. The general concept of the method and the tools to bound its complexity are developed in §4. In §5, we provide a rather simple though not optimal version of a sorting algorithm. The core of the proposed method, the strategy of comparisons, is given in §6. Finally, in §7, we present the group insertion method together with its complexity analysis.

2 The method of binary insertions

Let $e_1?e_2$ denote the operation of comparison of elements e_1 and e_2, and $e_1 < e_2$ denote the relation of order. W.l.o.g. we can assume that all inputs are different. For brevity, we refer to a linearly ordered set of elements as to a *chain*. By the *length* of a chain we mean the number of elements in it. The *rank* of an element in a chain is its number according to the numeration from 1 in ascending order.

We recall the essence of the binary insertion method [5], see also [9]. All elements are splitted into pairs, comparisons are made inside pairs, larger elements of pairs are sorted. As a result, we have a partial order shown[1] in Fig. 1. Elements in pairs are denoted by α_i, β_i, where $\beta_i > \alpha_i$, and indexing follows the ascending order of β_i. In the odd case $n = 2k + 1$, we denote by α_{k+1} an element not receiving a pair.

[1]The order is shown with the use of Hasse diagram, where edges connect elements with a known relation, see more details e.g. in [9].

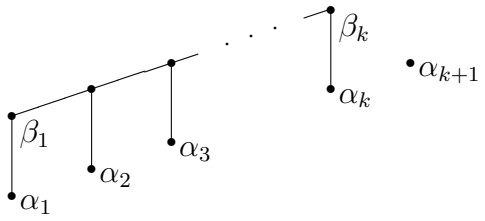

Figure 1: A partial order in the binary insertion method

By construction, the elements $\alpha_1, \beta_1, \beta_2, \ldots, \beta_k$ form a chain. The algorithm inserts the remaining elements α_i into this chain in turn. First, we use two comparisons to insert α_3, then also two comparisons for the element α_2. Next, all elements are inserted for which three comparisons are sufficient (these are α_5 and α_4 in this order), etc. Note that each element is inserted into a subchain of length $2^j - 1$, where $j = 2, 3, \ldots$ Only in the final series of insertions a subchain of incomplete length is used as a target.

It can be verified that j comparisons are sufficient to insert any element α_i satisfying $u_{j-1} < i \le u_j$, where $u_j = \frac{2^{j+1}+(-1)^j}{3}$. Hence, for $\lfloor 2^{k+1}/3 \rfloor \le n < \lfloor 2^{k+2}/3 \rfloor$ the complexity $f(n)$ of the sorting of an n-input set by the Ford–Johnson method is

$$f(n) = kn - \left\lfloor 2^{k+2}/3 \right\rfloor + \lfloor k/2 \rfloor + 1.$$

This expression may be rewritten in the asymptotic form. Let $n = \tau \cdot 2^{k+1}/3$, where $\tau \in [1,\, 2)$. Then

$$f(n) = n \log n - (2/\tau + \log(2\tau/3))\, n + \log n/2 + O(1). \tag{3}$$

The coefficient in the linear term takes the maximum value $\log(8/3) \approx 1.415$ when $\tau = 1$, and the minimum value $\log(4e \ln 2/3) \approx 1.329$ when $\tau = 2 \ln 2$. A more rigorous analysis of the method see in [5, 9].

Let us introduce a standard notation $M(m, n)$ for the complexity of merging of two chains of length m and n.

In [14], the bound $M(5, n) \le 5k - 8$ was proved for $n \le \frac{319}{448} \cdot 2^k - O(1)$. It provides the opportunity to insert five elements into a chain of length n by $5k - 8 + S(5) = 5k - 1$ comparisons, i.e. with the average complexity $k - 1/5$ per element. Thus, in the final stage of the Ford–Johnson method, it would be advantageous to insert elements with indices over u_{k-1} into a shorter subchain with the average complexity $k - 1/5$ instead of k. This observation leads to a refinement of the bound (3) for all n, except for those very close to the points of the sequence $\{2^i/3\}$. So, the maximal value of the constant in (2) decreases

to $c \approx 0.105$. A similar result was obtained in [10] by a very complicated way. In [10], merges of long chains via the method [2] are used instead of insertions for the final stage of the binary insertions algorithm. However, the bound for the infavourable case is improved significantly.

3 Preliminary observations

In the context of merging or insertion we will refer to a longer chain as to the *principal* chain. Let us call *interval* (α, β) the set of elements of the principal chain located between some elements α and β, that is, $(\alpha, \beta) = \{z \in Z \mid \alpha < z < \beta\}$, where Z is the principal chain. We also allow imaginary elements $\pm\infty$ to designate the ends of an interval. By definition, $-\infty < \alpha < +\infty$ holds for all α. We denote the *length* of an interval as $|(\alpha, \beta)|$ and define it as the number of possibilities to insert a new element in it, that is the incremented number of elements in the interval.

Let $Q(a)$ denote the maximal number n such that the insertion of an element into an interval of length n has complexity at most a, in other words, $M(1, n-1) \le a$. Trivially,

$$Q(a) = 2^{\lfloor \log a \rfloor}. \tag{4}$$

By analogy, let $P(a)$ denote the maximal n such that $M(2, n-1) \le 2a$. It is known [6, 7] that

$$P(a) = \begin{cases} \left\lfloor \frac{17}{14} \cdot 2^a \right\rfloor, & a \in \mathbb{N} \\ \left\lfloor \frac{12}{7} \cdot 2^{a-1/2} \right\rfloor, & (a - 1/2) \in \mathbb{N} \end{cases}. \tag{5}$$

Let $M(m \times k, n)$ stand for the complexity of merging of m length-k chains with a chain of length n. We define $Q_m(a)$ and $P_m(a)$ as the maximal number n such that $M(m \times 1, n-1) \le am$ and respectively $M(m \times 2, n-1) \le 2am$. By definition, $Q_1(a) = Q(a)$ and $P_1(a) = P(a)$. The bounds $Q_m(a) \ge Q(a) - (m-1)$ and $P_m(a) \ge P(a) - 2(m-1)$ are quite straightforward.

A less obvious bound follows from (5). Since

$$Q_{2m}(a) \ge P_m(a - 1/2), \tag{6}$$

then

$$Q_2(a) \ge \begin{cases} 2^a - 1, & a \in \mathbb{N} \\ \left\lfloor \frac{17}{14} \cdot 2^{a-1/2} \right\rfloor, & (a - 1/2) \in \mathbb{N} \end{cases}. \tag{7}$$

Therefore, $Q_2(a)$ may be sufficiently larger than $Q(a)$.

For $m = 1$, only half-integer arguments of functions $P_m(a)$ and $Q_{2m}(a)$ make sense. Say, $P_1(r - 1/4) = P_1(r - 1/2) \approx \frac{6}{7} \cdot 2^r$, $r \in \mathbb{N}$. However, matters change when m grows.

Simple example. For warm-up let us prove the bound

$$P_m\left(r - \frac{1}{4} + \frac{1}{4m}\right) \geq \frac{51}{56} \cdot 2^r - 4m \tag{8}$$

illustrating the idea behind the general method.

Let α be the element of rank $\left\lceil \frac{17}{56} \cdot 2^r \right\rceil - 2m + 1$ in a principal chain of length $\left\lceil \frac{51}{56} \cdot 2^r \right\rceil - 4m$. Then $|(-\infty, \alpha)| \leq Q_2(r - 3/2) - 2(m - 1)$ by (7), and $|(\alpha, +\infty)| \leq P(r - 1) - 2(m - 1)$ by (5).

The algorithm processes any pair $\beta_0 < \beta_1$ in the following way. Compare β_0 with α. If $\beta_0 > \alpha$, then insert the pair into the interval $(\alpha, +\infty)$ by $2(r-1)$ comparisons via the method [6, 7]. Otherwise, split the pair. Insert β_1 into the principal chain by r comparisons via the binary method, and put β_0 into a container. If the container is empty, turn to processing of the next pair. Else, if the container got two elements, insert them into the interval $(-\infty, \alpha)$ by $2r - 3$ comparisons (the complexity bound is provided by (5)).

While the algorithm proceeds, the principal chain and its subintervals become longer. Thus, we provide a reserve of $2(m-1)$ in both intervals to ensure the same complexity bounds for all m pairs: $2(r - 3/2)$ comparisons to insert two elements into $(-\infty, \alpha)$, and $2(r - 1)$ comparisons to insert a pair into $(\alpha, +\infty)$.

After the main process is finished, the container may still keep a single element. We can trivially insert it into the interval $(-\infty, \alpha)$ by $r - 1$ comparisons. So, the total number of comparisons performed by the algorithm does not exceed $m + (m - 1)(r + r - 3/2) + (r + r - 1) = 2m(r - 1/4) + 1/2$, and the required bound (8) follows.

4 General method

In any method of merging or insertion, we can consider a comparison operation as a step of dividing of the principal chain into shorter intervals where the inserted elements should be located. So, the algorithm may be built recursively: after a sequence of comparisons, insertion into a long chain reduces to insertions into shorter chains.

As in the simple example above, the general algorithm processes pairs and calls procedures providing bounds (4), (5), (7) to insert elements in some intervals. Certain intervals receive containers having the storage space of 2

elements each. As soon as the container is full, its two elements are extracted and processed further.

Let us define *conditional complexity* (of a pair's insertion algorithm) as the number of comparisons applied to the elements of a pair in the worst case assuming that none of the two remained in a container (for the purpose of calculating, we may assume that all containers on the way were non-empty). Let us stress that, while processing two elements jointly, we account any comparison as 1/2 comparisons attributed to each of the two.

For example, the conditional complexity of the insertion algorithm in the simple example above is $2(r - 1/4)$.

Now we can upper bound the complexity of the pair insertion algorithm by

$$\rho m + C \log_2(n + 2m), \tag{9}$$

where ρ is the conditional complexity, m is the number of pairs to insert, n is the initial length of the principal chain, and C is the number of containers. We bounded the additional number of comparisons required to insert an element remaining in a container very roughly as $\log_2(n + 2m)$ (a more accurate argument leads to the bound $O(1)$).

We now introduce the concept of *comparison strategy*. Let numbers $y_1, \ldots, y_d \in \mathbb{R}$ satisfy $0 \le y_1 < y_2 < \ldots < y_d < 1$. Let $p_{r,i}$ characterize the length of some interval (further denoted as $J_{r,i}$) in a sense that the conditional complexity of the pair insertion into this interval is at most $2(r + y_i)$. In addition, let $x_{r,j} = p_{r,j} \cdot 2^{-r}$.

For any $i = 1, \ldots, d$ and $r \ge r_0$, we introduce a reduced comparison tree $T_{r,i}$ for the insertion of a pair into the interval $J_{r,i}$. Under a *reduced tree* we understand a subtree of a comparison tree. Leaves of $T_{r,i}$ correspond to (recursive) calls of procedures of insertion into shorter intervals named *terminal*. Pair splitting along the reduced tree is not allowed, except at terminal vertices.

We assign to any leaf of $T_{r,i}$ some restrictions on the lengths of terminal intervals ensuring the bound on the conditional complexity of insertion into $J_{r,i}$ to hold. If a leaf on the distance s from the root corresponds to a terminal interval (α', α''), then the restriction has the form $|(\alpha', \alpha'')| \le p_{l,j}$, where $l + y_j + s/2 \le r + y_i$. For a single element β to insert into (α', α''), the restriction looks like $|(\alpha', \alpha'')| \le p_{l,j}$ or $|(\alpha', \alpha'')| \le 2^l$. In the first case, β adds $l + y_j + s/2 + 1/2$ to the conditional complexity, while in the second case it adds $l + s/2$. The sum for two elements of a pair should not exceed $2(r + y_i)$.

Assume that the ranks of α', α'' are expressed as linear combinations of $p_{l,j}$. Then, some restrictions may hold trivially (say, in the case of identical left and right sides of some inequality). Others constitute a *system of restrictions* of the strategy.

Now we define *comparison strategy* as a family of reduced trees (that is, algorithms) $T_{r,i}$ and a system of restrictions providing conditional complexity bounds $2(r+y_i)$ to insert a pair into an interval $J_{r,i}$. The *depth* of the strategy is the maximal depth of $T_{r,i}$.

If the minimal conditional complexity of insertion of a pair into a terminal interval corresponding to a leaf of $T_{r,i}$ is $2(r'+y_j)$, then define $w_{r,i} = r - r' + 1$.

From now on, we consider only *uniform* strategies, where ranks of target elements in $J_{r,i}$ do not depend on r (up to a rounding). Then all trees $T_{r,i}$ are similar to some tree T_i. We may assume that any inner vertex of T_i is associated with a pair (y, Δ), $y \in \{0,1\}$, $\Delta \in \mathbb{R}$, encoding the comparison $\beta_y ? \alpha$, where (β_0, β_1) is the pair to insert, and α is the element of rank $\Delta \cdot |J_{r,i}|$ in the interval $J_{r,i}$. In particular, $w_{r,i} = w_i$ for any r. We call $\max_i w_i$ the *width* of a uniform strategy. The width of a strategy characterizes the maximum difference in the complexity of insertion between the initial interval and its terminal subintervals.

Below, we allow non-integer values for lengths of intervals and ranks of elements assuming that rounding to the nearest integer in some direction is required. In the case of a statement about the complexity of insertion into an interval of non-integer length x, we will require the fulfillment of this statement for an interval of length $\lceil x \rceil$.

Our next step is to describe a transition from a strategy to an actual algorithm. Assuming the existence of limits $z_i = \lim_{r\to\infty} x_{r,i}$, we rewrite strategy restrictions in terms of z_i and therefore obtain a linear programming problem.

Consider a system σ of normalized linear inequalities

$$\sum_{i=1}^{d}\sum_{j=0}^{w-1} a_{i,j,k} \cdot x_{t+j,i} \le b_k, \quad \text{where} \quad \sum_{i=1}^{d}\sum_{j=0}^{w-1} |a_{i,j,k}| = 1, \qquad k = 1, \ldots, K \quad (10)$$

defined on the sequence of d-dimensional real vectors $x_0, x_1, \ldots \in \mathbb{R}^d$, $x_j = (x_{j,1}, x_{j,2}, \ldots, x_{j,d})$ for all $t \ge 0$. Here w is the width of the system, and $a_{i,j,k}, b_k$ are real constants.

By the formal substitution $x_{j,i} = z_i$ for all i, j, we transform (10) to a *reduced system*

$$\sum_{i=1}^{d} a_{i,k} \cdot z_i \le b_k, \qquad a_{i,k} = \sum_{j=0}^{w-1} a_{i,j,k}, \qquad k = 1, \ldots, K, \quad (11)$$

over variables z_i. The system (11) determines a simplex $T(\sigma) \subset \mathbb{R}^d$. For any $\varepsilon \ge 0$ define an embedded simplex $T^\varepsilon(\sigma) \subset T(\sigma)$ by

$$\sum_{i=1}^{d} a_{i,k} \cdot z_i \le b_k - \varepsilon, \qquad k = 1, \ldots, K. \quad (12)$$

We say that the subsequence $x_l, x_{l+1}, \ldots, x_{l'}$ satisfies (10), if restrictions (10) hold for all $t = l, \ldots, l' + w - 1$, and for some extension of the subsequence by vectors $x_{l'+1}, \ldots, x_{l'+w-1}$.

Let $||\cdot||$ denote the l_∞-norm in a vector space (maximal absolute value of the vector coordinate), and $\langle \cdot, \cdot \rangle$ denote the Euclidean inner product of vectors. We rely upon the following simple fact about the rate of convergence to a given solution of (10).

Lemma 1. *Let $u \in T^\varepsilon(\sigma)$, $v \in T(\sigma)$, and $0 < \varepsilon < ||v - u||$. Then there exists a sequence $\{x_i\} \subset [u, v]$ satisfying the system σ (10) of width w, which begins at $x_0 = x_1 = \ldots = x_{w-1} = u$ and converges to v with the rate*

$$||v - x_i|| \le (1 - \varepsilon\Delta)^{i/w-1} \cdot ||v - u||, \qquad \Delta = ||v - u||^{-1}. \tag{13}$$

Proof. Define $\varepsilon_i = \varepsilon(1 - \varepsilon\Delta)^i$, and

$$x_{iw+j} = x_{(i-1)w+j} + \varepsilon_{i-1}\Delta \cdot (v - u). \tag{14}$$

for all $i \ge 1$, and $j = 0, \ldots, w - 1$. By definition,

$$v - x_{iw+j} = (1 - (\varepsilon_0 + \ldots + \varepsilon_{i-1})\Delta)\,(v - u) = (1 - \varepsilon\Delta)^i \cdot (v - u). \tag{15}$$

Therefore, the bound (13) holds. We are left to check that $\{x_i\}$ satisfies σ.

Since $x_{lw} = x_{lw+1} = \ldots = x_{lw+w-1} \in [u, v] \subset T(\sigma)$, the inequalities (10) hold for any $t = lw$, where $l \in \mathbb{N}$.

Let us show that $x_{lw} \in T^{\varepsilon_l}(\sigma)$. When $l = 0$, this is provided by the condition $x_0 = u \in T^\varepsilon(\sigma)$. Denote $a_k = (a_{1,k}, \ldots, a_{d,k})$. Then it follows from (15) that for $l > 1$ and any $k = 1, \ldots, K$,

$$\langle a_k, x_{lw} \rangle = \langle a_k, (1 - \varepsilon_l/\varepsilon)v + (\varepsilon_l/\varepsilon)u \rangle \le (1 - \varepsilon_l/\varepsilon)b_k + (\varepsilon_l/\varepsilon)(b_k - \varepsilon) = b_k - \varepsilon_l$$

due to linearity of inner product.

Now consider $lw < t < (l + 1)w$. The left sides of inequalities (10) involve only coordinates of vectors x_{lw}, $x_{(l+1)w}$. With the use of (14), and taking into account $x_{lw} \in T^{\varepsilon_l}(\sigma)$, we obtain for any $k = 1, \ldots, K$ that

$$\sum_{i=1}^{d}\sum_{j=0}^{w-1} a_{i,j,k} \cdot x_{t+j,i} \le \sum_{i=1}^{d}\sum_{j=0}^{w-1} a_{i,j,k} \cdot x_{lw,i} + \varepsilon_l \cdot \sum_{i=1}^{d}\sum_{j=0}^{w-1} |a_{i,j,k}| \le (b_k - \varepsilon_l) + \varepsilon_l.$$

Hence, the sequence $\{x_i\}$ satisfies σ. □

The next lemma is our main technical ingredient for proving upper bounds. Denote

$$\pi(a) = \lim_{r \to \infty} P(r + a) \cdot 2^{-r}.$$

It is easy to see from (5) that

$$\pi(a) = \begin{cases} \frac{17}{14}, & 0 \le a < \frac{1}{2} \\ \frac{12}{7}, & \frac{1}{2} \le a < 1 \end{cases}. \tag{16}$$

Lemma 2. *Let $y_1, \ldots, y_d \in \mathbb{R}$, where $0 \le y_1 < y_2 < \ldots < y_d < 1$. Suppose we have a comparison strategy of depth h and width w specifying a system of restrictions (10) on $x_{r,i}$, where $x_{r,i} \cdot 2^r$ is a lower bound on the length of an interval for the insertion of an ordered pair with conditional complexity $2(r + y_i)$. Let $v = (v_1, \ldots, v_d) \in T(\sigma)$, $u = (u_1, \ldots, u_d) \in T^{\varepsilon}(\sigma)$, where $0 < \varepsilon < ||v - u||$. Suppose that $u_i < v_i$, $u_i \le \pi(y_i)$ holds for all $i = 1, \ldots, d$. Denote $u_{min} = \min\{u_i\}$. Then for any $r \in \mathbb{N}$ and $m \in \mathbb{N}$, $\varphi > 0$ satisfying*

$$m \le \min\{u_{min}, \varepsilon/2\} \cdot 2^{r-\varphi-1}, \tag{17}$$

the following bound holds:

$$P_m\left(r + y_i + \frac{(r+2)\varphi \cdot 2^{\varphi}}{m}\right) \ge \\ \ge v_i \cdot 2^r - \frac{R}{\varepsilon} \cdot m \cdot 2^{\varphi+2} - R \cdot 2^r \cdot e^{-\frac{\varepsilon}{2R}\left(\frac{\varphi}{w(d+1)(h+1)} - 2\right)}, \tag{18}$$

where $i = 1, \ldots, d$, and $R = ||v - u||$.

Proof. Take some r and appropriate parameters m and φ. Set $L = \left\lfloor \frac{\varphi}{(h+1)(d+1)} \right\rfloor$ and $q = r - L$. It follows from (17) and (16) that $m \le \frac{6}{7} \cdot 2^{r-\varphi}$, hence $r > \varphi \ge L$. Therefore, $q > 0$.

Let $\delta \in (0, \varepsilon)$ be a parameter to be chosen later. Consider the point $v' = (1 - \delta/\varepsilon)v + (\delta/\varepsilon)u \in [u, v]$. Applying the argument from Lemma 1 we deduce that $v' \in T^{\delta}(\sigma)$. Indeed, for any $k = 1, \ldots, K$,

$$\langle a_k,\, v' \rangle = \langle a_k,\, (1 - \delta/\varepsilon)v + (\delta/\varepsilon)u \rangle \le (1 - \delta/\varepsilon)b_k + (\delta/\varepsilon)(b_k - \varepsilon) = b_k - \delta.$$

By definition, the width of the system of restrictions does not exceed the width of its corresponding strategy. We may assume that both widths are equal to w. Consider σ', a system obtained from σ by subtracting δ from the right-hand sides of the inequalities (10). Then, $v' \in T(\sigma')$ and $u \in T^{\varepsilon-\delta}(\sigma')$. Applying Lemma 1 to the system σ', we construct a sequence $\{x_l = (x_{l,1}, \ldots, x_{l,d})\}$ converging to v', and starting from $x_q = \ldots = x_{q+w-1} = u$.

Let $X_{l,i} = x_{q+l,i} \cdot 2^{q+l}$. By the condition of the lemma, the given comparison strategy provides the conditional complexity $2(q + l + y_i)$ for a pair insertion

into an interval of length $X_{l,i}$. We show by induction on $dl + i$ that to insert m pairs into an interval of length

$$X'_{l,i} = X_{l,i} - 2^{(dl+i)h} \cdot 2m, \tag{19}$$

there is an algorithm of the same conditional complexity $2(q + l + y_i)$ with $c_{dl+i} \le (dl + i) \cdot 2^{(dl+i)(h+1)}$ containers. (Recall that an interval of length $X'_{l,i}$ means an interval of length $\lceil X'_{l,i} \rceil$.)

First, check that $X'_{l,i} > 0$. Indeed, by (17),

$$X_{l,i} \ge u_i \cdot 2^{q+l} \ge m \cdot 2^{\varphi+1-r+q+l} \ge 2m \cdot 2^{(h+1)(d+1)L+l-L} > 2m \cdot 2^{(d+1)Lh}.$$

Now we show that the reserve of length and the number of containers are sufficient for insertion, then we will ensure that the restrictions (10) for the subsequence of vectors x'_{q+l}, $0 \le l \le L$, with components $x'_{q+l,i} = X'_{l,i} \cdot 2^{-(q+l)}$ are satisfied.

For $0 \le l \le w - 1$ (the base of induction), the required properties are guaranteed by the conditions of the lemma on u_i: the lengths of the intervals allow to apply bounds (5). There is no need in containers. To insert m pairs a reserve of length $2m - 2$ is required with possible additional correction by 1 due to the length rounding, and by 1 to compensate a potential rounding error in (16) with respect to (5). Thus, a reserve of length at least $2m$ provided by (19) is sufficient.

For $l \ge w$, starting from the partition of an interval of length $X_{l,i}$ by subintervals in accordance to the strategy, we will build the partition of an interval of length $X'_{l,i}$. Essentially, the task is to distribute the reserve of length $2^{(dl+i)h} \cdot 2m$ between subintervals.

We move along the comparison tree from the root to a leaf. Assume that at some vertex a comparison with the element[2] α of the principal chain should be done. Let (α', α'') be the minimal interval including α, and with the ends being either elements of preceding comparisons or the ends of the initial interval. Then we divide the reserve of length for the interval (α', α'') equally between the intervals (α', α) and (α, α''). Since the depth of the comparison tree is h, any subinterval produced by the points of comparisons receives the reserve at least $2^{(dl+i-1)h} \cdot 2m$. By the induction hypothesis, this reserve is sufficient to insert a pair with conditional complexity $2(q + l' + y_j)$, or to insert a single element with conditional complexity[3] $q + l' + y_j + 1/2$, where $l' < l$ or $j < i$. Of

[2]Since the strategy operates with real values of lengths, by an element here we mean some conventional point to partition the interval.

[3]By the conditional complexity of insertion of a single element we mean its contribution to the pair's insertion conditional complexity.

course, the chosen reserve is limited from above by the length of the interval to keep it positive.

Also notice, that the conditional complexity of insertion into a terminal interval of the strategy is at least $2(q + l - w) \geq 2q$, hence the recursive call of the algorithm (for insertion into the terminal interval) is legal. In the (not improbable) case when the strategy assumes insertion of a single element into an inner interval with the conditional complexity over $q + l + y_i + 1/2$, we just call an algorithm of insertion into the initial interval of length $X'_{l,i}$.

To construct the partition of an interval of length $\lceil X'_{l,i} \rceil$, we take the partition of the interval of length $X'_{l,i}$ and round ranks of all partition elements to the nearest integers from above. So, the length of any inner interval (including terminal) changes at most by 1, that is, turns into its rounding in some direction. Therefore, our estimate of the conditional complexity of the algorithm does not change after taking roundings into account.

Let us estimate the sufficient number of containers. The comparison tree has at most 2^h leaves. Each leaf corresponds to at most two terminal intervals. By the induction hypothesis, insertion into any terminal interval is implemented by an algorithm involving at most $2c_{dl+i-1}$ containers. Additionally, $2^{h+1} + 1$ containers may be required to associate with terminal intervals and the initial interval. Thus, we obtain

$$c_{dl+i} \leq 2^{h+1} \left((dl + i - 1) \cdot 2^{(dl+i-1)(h+1)} \right) + 2^{h+1} + 1 \leq (dl + i) \cdot 2^{(dl+i)(h+1)}.$$

Next, check the fulfillment of restrictions σ for $x'_{l,i}$. Put formally $x'_{j,i} = x_{j,i}$ when $j > r$. Choose $\delta = 2^{\varphi - r + 1} \cdot m$. By the condition (17), the inequality $\delta < \varepsilon/2$ holds. Furthermore, $x'_{j,i} - x_{j,i} \in (-\delta, 0]$ holds for all $j \geq q$ due to (19) and the choice of parameters φ and δ. Hence, for all $t \geq q$ and any $k = 1, \ldots, K$, we obtain

$$\sum_{i=1}^{d} \sum_{j=0}^{w-1} a_{i,j,k} \cdot x'_{t+j,i} \leq \sum_{i=1}^{d} \sum_{j=0}^{w-1} a_{i,j,k} \cdot x_{t+j,i} + \delta \cdot \sum_{i=1}^{d} \sum_{j=0}^{w-1} |a_{i,j,k}| \leq (b_k - \delta) + \delta.$$

So, the sequence $\{x'_l = (x'_{l,1}, \ldots, x'_{l,d})\}$ satisfies the system of restrictions σ for $t \geq q$. This completes the induction step. We proved that the pair's insertion into an interval of length $X'_{L,i}$ is performed with the conditional complexity $2(r + y_i)$.

We are left to verify the bound (18). The choice of L provides

$$c_{dl+i} \leq (d + 1)L \cdot 2^{(d+1)(h+1)L} \leq \varphi \cdot 2^{\varphi}. \tag{20}$$

The Lemma 1 implies that

$$\begin{aligned}|x'_{r,i} - v_i| \le |x'_{r,i} - x_{r,i}| + |x_{r,i} - v'_i| + |v'_i - v_i| \le \\ \le \delta + (1 - (\varepsilon - \delta)/R)^{L/w-1} \cdot R + (\delta/\varepsilon) \cdot |v_i - u_i| \le \\ \le \delta + e^{-(\varepsilon-\delta)(L/w-1)/R} \cdot R + (\delta/\varepsilon) \cdot R \le \\ \le \delta + e^{-\frac{\varepsilon}{2R}\left(\frac{\varphi}{w(d+1)(h+1)}-2\right)} \cdot R + (\delta/\varepsilon) \cdot R < \\ < \frac{2R}{\varepsilon} \cdot m \cdot 2^{\varphi-r+1} + e^{-\frac{\varepsilon}{2R}\left(\frac{\varphi}{w(d+1)(h+1)}-2\right)} \cdot R, \quad (21)\end{aligned}$$

where we also used the well-known inequality $(1-x)^{1/x} \le e^{-1}$ for $x \in (0, 1)$. Now the bound (18) is obtained via the substitution of (20) and (21) into (9), where we may assume $n + 2m \le X_{L,i} = x_{r,i} \cdot 2^r < 2^{r+2}$, since $x_{r,i} \le 2^{y_i+1/2} < 4$ by the simple information-theoretic reasoning. □

Note, the bounds in both lemmas are very rough. We sacrifice precision in favour of simplicity and generality. In particular, the strategies in the examples below need substantially smaller number of containers.

5 Advanced example

Before we turn to the asymptotically optimal insertion algorithm, consider a simpler example extending bounds (5) and improving over (8).

Lemma 3. *Let*[4] $m \asymp 2^{r/2}$, $v_0 = \frac{80}{63}$, $v_1 = \frac{286}{189}$, $v_2 = \frac{338}{189}$, $v_3 = \frac{58}{27}$. *Then*

$$P_m(r + i/4 + 2^{-\gamma r}) \ge v_i \cdot 2^r - O\left(2^{(1-\gamma)r}\right) \quad (22)$$

for $i = 0, \dots, 3$ *and* γ *small enough, say, for* $\gamma = \frac{1}{100000}$.

Proof. Let $p_{k,i}$ be a bound on the length of an interval with pair's insertion conditional complexity $2(k + i/4)$, $i = 0, \dots, 3$. We are going to describe an appropriate strategy. For any i, the sequence of comparisons is fixed: pair's elements are compared with elements $\alpha_1, \alpha_2, \dots$ of the principal chain in turn as shown in Fig. 2. After the comparison with α_j is performed, we move either to a terminal vertex or to the comparison with α_{j+1}.

We summarize the sequence of comparisons performed depending on their results in Table 1. Its rows correspond to all possible terminal vertices. In the left column, the path to a leaf is shown. In the right column, we write intervals where elements of the pair fall, and the conditional complexity of insertion.

[4]The symbol $\asymp$ denotes equal orders of growing.

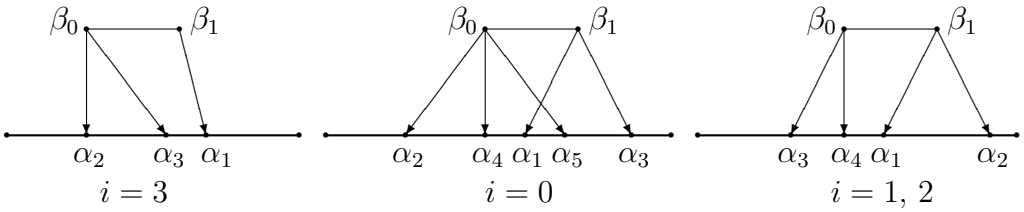

Figure 2: The sequence of comparisons

Let us list restrictions of the strategy securing the claimed complexity bounds. In the simplest case $i = 3$, we choose elements of ranks $p_{k,1}$, 2^k, $3 \cdot 2^{k-1}$ as $\alpha_1, \alpha_2, \alpha_3$, respectively. Then, we require the following conditions:

$$|(\alpha_1, +\infty)| = p_{k,3} - p_{k,1} \le p_{k-1,0},$$
$$|(\alpha_3, +\infty)| = p_{k,3} - 3 \cdot 2^{k-1} \le p_{k-1,1}.$$

In the case $i = 0$, we choose elements of ranks $p_{k-1,2}$, $p_{k-2,3}$, $p_{k,0} - p_{k-4,3}$, $p_{k-2,3} + 2^{k-2}$, $p_{k-2,3} + 3 \cdot 2^{k-3}$ as $\alpha_1, \ldots, \alpha_5$. Also, the following restrictions should hold:

$$|(\alpha_1, +\infty)| = p_{k,0} - p_{k-1,2} \le p_{k-2,1},$$
$$|(\alpha_2, +\infty)| = p_{k,0} - p_{k-2,3} \le p_{k-1,1},$$
$$|(\alpha_1, \alpha_3)| = p_{k,0} - p_{k-1,2} - p_{k-4,3} \le 2^{k-2},$$
$$|(\alpha_5, \alpha_3)| = p_{k,0} - p_{k-2,3} - p_{k-4,3} - 3 \cdot 2^{k-3} \le p_{k-3,2}.$$

In the case $i = 1$, the elements of ranks $p_{k-1,3}$, $p_{k,1} - p_{k-3,1}$, $p_{k-1,0}$, $p_{k-1,0} + p_{k-2,0}$ are chosen as $\alpha_1, \ldots, \alpha_4$. The restrictions are

$$|(\alpha_1, \alpha_2)| = p_{k,1} - p_{k-1,3} - p_{k-3,1} \le 2^{k-2},$$
$$|(\alpha_4, \alpha_2)| = p_{k,1} - p_{k-1,0} - p_{k-2,0} - p_{k-3,1} \le p_{k-2,1}.$$

In the last case $i = 2$, we choose elements of ranks $p_{k,0}$, $p_{k,2} - 2^{k-2}$, $p_{k-1,1}$, $p_{k-1,1} + p_{k-2,1}$ as $\alpha_1, \ldots, \alpha_4$. The additional restrictions:

$$|(\alpha_1, \alpha_2)| = p_{k,2} - p_{k,0} - 2^{k-2} \le p_{k-3,3},$$
$$|(\alpha_4, \alpha_2)| = p_{k,2} - p_{k-1,1} - p_{k-2,1} - 2^{k-2} \le p_{k-2,2}.$$

After rewriting in terms of $x_{k,i} = p_{k,i} \cdot 2^{-k}$ and normalization, we obtain

conditional complexity $2(k+3/4)$	
$\beta_1 < \alpha_1$	$\beta_0, \beta_1 \in (-\infty, \alpha_1)$: $2(k+1/4)$
$\ldots, \beta_0 < \alpha_2$	$\beta_0 \in (-\infty, \alpha_2)$: k, $\beta_1 \in (\alpha_1, +\infty)$: $k-1/2$
$\ldots, \beta_0 < \alpha_3$	$\beta_0 \in (\alpha_2, \alpha_3)$: $k-1$, $\beta_1 \in (\alpha_1, +\infty)$: $k-1/2$
$\ldots, \beta_0 > \alpha_3$	$\beta_0, \beta_1 \in (\alpha_3, +\infty)$: $2(k-3/4)$
conditional complexity $2k$	
$\beta_1 < \alpha_1$	$\beta_0, \beta_1 \in (-\infty, \alpha_1)$: $2(k-1/2)$
$\ldots, \beta_0 < \alpha_2$	$\beta_0 \in (-\infty, \alpha_2)$: $k-3/4$, $\beta_1 \in (\alpha_1, +\infty)$: $k-5/4$
$\ldots, \beta_1 > \alpha_3$	$\beta_0 \in (\alpha_2, +\infty)$: $k-1/4$, $\beta_1 \in (\alpha_3, +\infty)$: $k-11/4$
$\ldots, \beta_0 < \alpha_4$	$\beta_0 \in (\alpha_2, \alpha_4)$: $k-2$, $\beta_1 \in (\alpha_1, \alpha_3)$: $k-2$
$\ldots, \beta_0 < \alpha_5$	$\beta_0 \in (\alpha_4, \alpha_5)$: $k-3$, $\beta_1 \in (\alpha_1, \alpha_3)$: $k-2$
$\ldots, \beta_0 > \alpha_5$	$\beta_0, \beta_1 \in (\alpha_5, \alpha_3)$: $2(k-5/2)$
conditional complexity $2(k+i/4)$, $i = 1, 2$	
$\beta_1 < \alpha_1$	$\beta_0, \beta_1 \in (-\infty, \alpha_1)$: $2(k+(i-2)/4)$
$\ldots, \beta_1 > \alpha_2$	$\beta_0 \in (-\infty, +\infty)$: $k+(i+2)/4$, $\beta_1 \in (\alpha_2, +\infty)$: $k+(i-10)/4$
$\ldots, \beta_0 < \alpha_3$	$\beta_0 \in (-\infty, \alpha_3)$: $k+(i-3)/4$, $\beta_1 \in (\alpha_1, \alpha_2)$: $k+(i-9)/4$
$\ldots, \beta_0 < \alpha_4$	$\beta_0 \in (\alpha_3, \alpha_4)$: $k+(i-7)/4$, $\beta_1 \in (\alpha_1, \alpha_2)$: $k+(i-9)/4$
$\ldots, \beta_0 > \alpha_4$	$\beta_0, \beta_1 \in (\alpha_4, \alpha_2)$: $2(k+i/4-2)$

Table 1: The strategy description

the following system σ of width 5:

$$\begin{aligned}
(2x_{k,3} - 2x_{k,1} - x_{k-1,0})/5 &\le 0,\\
(2x_{k,3} - x_{k-1,1})/3 &\le 1,\\
(4x_{k,0} - 2x_{k-1,2} - x_{k-2,1})/7 &\le 0,\\
(4x_{k,0} - 2x_{k-1,1} - x_{k-2,3})/7 &\le 0,\\
(16x_{k,0} - 8x_{k-1,2} - x_{k-4,3})/25 &\le 4/25,\\
(16x_{k,0} - 4x_{k-2,3} - 2x_{k-3,2} - x_{k-4,3})/23 &\le 6/23,\\
(8x_{k,1} - 4x_{k-1,3} - x_{k-3,1})/13 &\le 2/13,\\
(8x_{k,1} - 4x_{k-1,0} - 2x_{k-2,1} - 2x_{k-2,0} - x_{k-3,1})/17 &\le 0,\\
(8x_{k,2} - 8x_{k,0} - x_{k-3,3})/17 &\le 2/17,\\
(4x_{k,2} - 2x_{k-1,1} - x_{k-2,2} - x_{k-2,1})/8 &\le 1/8.
\end{aligned}$$

The reduced system then looks like

$$\begin{aligned}
(2z_3 - 2z_1 - z_0)/5 &\le 0,\\
(2z_3 - z_1)/3 &\le 1,\\
(4z_0 - 2z_2 - z_1)/7 &\le 0,\\
(4z_0 - 2z_1 - z_3)/7 &\le 0,\\
(16z_0 - 8z_2 - z_3)/25 &\le 4/25,\\
(16z_0 - 5z_3 - 2z_2)/23 &\le 6/23,\\
(7z_1 - 4z_3)/13 &\le 2/13,\\
(5z_1 - 6z_0)/17 &\le 0,\\
(8z_2 - 8z_0 - z_3)/17 &\le 2/17,\\
(3z_2 - 3z_1)/8 &\le 1/8.
\end{aligned} \tag{23}$$

One can check that the vector $v = (v_0, \dots, v_3)$ from the statement is a solution of the linear programming problem for the system (23) maximizing all variables z_i. In particular, $v \in T(\sigma)$. It can be also verified by direct calculation that $u = (1.02, 17/14, 1.47, 12/7) \in T^{\varepsilon}(\sigma)$ for $\varepsilon = 1/350$.

We are left to apply Lemma 2 to our strategy with parameters $d = 4$, $\{y_i\} = \{0, 1/4, 1/2, 3/4\}$, $w = 5$, $h = 5$. Recall that Lemma 2 requires $u \le (17/14, 17/14, 12/7, 12/7)$, see (16), and this indeed holds true. Note that $R = ||v - u|| = 58/27 - 12/7 \approx 0.434$. Set $\varphi = \log m - \gamma r - 2\log(r+2)$, where γ is to be chosen soon. Then the condition (17) is satisfied for growing r. Finally, by Lemma 2, we obtain

$$\begin{aligned}
P_m\left(r + \frac{i}{4} + 2^{-\gamma r}\right) \ge P_m\left(r + \frac{i}{4} + \frac{(r+2)\varphi \cdot 2^{\varphi}}{m}\right) \ge\\
\ge v_i \cdot 2^r - O\left(m^2 \cdot 2^{-\gamma r}\right) - O\left(2^r \cdot e^{-\frac{\varepsilon(1/2-\gamma)r - O(\log r)}{2Rw(d+1)(h+1)}}\right).
\end{aligned}$$

The last two terms fall into $O\left(2^{(1-\gamma)r}\right)$ for any small enough γ, say, for $\gamma = \frac{1}{100000}$. □

We can try to apply already proven bounds to constructing of the theoretically fast sorting algorithm.

Corollary 1. *For $r \in \mathbb{N}$, $m \asymp 2^{r/2}$, and $\gamma = \frac{1}{100000}$, the following holds:*

$$Q_m\left(r - 1/4 + 2^{-\gamma r}\right) \ge \frac{143}{189} \cdot 2^r - O\left(2^{(1-\gamma)r}\right).$$

Proof. Easily follows from Lemma 3 by (6). □

Theorem 1. $S(n) \le \log(n!) + 0.09n + o(n)$.

Proof. We will follow the analysis from §2. Let $n = \tau 2^{k+1}/3$, where $\tau \in [1,\, 2)$. If $\tau \notin \left[\frac{143}{126},\, \frac{235}{126}\right]$, then we apply the estimate (3) and use direct calculation to verify that the stated bound is valid. Otherwise, we modify the procedure of the final stage of the Ford–Johnson method.

Take $m \asymp \sqrt{n}$. First, we insert elements α_i (shown in Fig. 1) with indices i satisfying

$$u_{k-1} < i \le Q_m\left(k - 1/4 + 2^{-\gamma k}\right) - u_{k-1}$$

by grouping them into portions of size m in the order opposite to the numeration via the method of Lemma 3. In such a way, $\frac{17}{189} \cdot 2^k - O(n^{1-\gamma})$ elements may be inserted with the average complexity $k - 1/4 + O(n^{-\gamma})$ per element by Corollary 1. After this insertion is performed, the length of the principal chain is still below 2^k. So, each of the remaining elements may be inserted by the standard binary algorithm via k comparisons.

Thus, we save $\left(\frac{1}{4} - O(n^{-\gamma})\right) \cdot \left(\frac{17}{189} \cdot 2^k - O(n^{1-\gamma})\right)$ comparisons against the bound (3). As a consequence,

$$S(n) \le n \log n - (\theta/\tau + \log(2\tau/3))\, n + O(n^{1-\gamma}),$$

where $\theta = 2 + \frac{17}{504}$. The coefficient in the linear term takes the maximum value $\log(2\theta e \ln 2/3) > \log e - 0.09$ in the case $\tau = \theta \ln 2$. □

The presented sorting algorithm does not improve over [10]. The problem lies in the large step of discretization of values for the conditional complexity in Lemma 3. Instead, the step should be chosen as $o(1)$, and the dimension d should be growing function of n. But then it would be hard to describe separate optimal strategies for different target values of conditional complexity. So we need a universal strategy which is good enough for any value.

6 Universal strategy

Here, we construct a comparison strategy to prove a more precise asymptotic complexity bound for sorting. It is rather obvious, that to sort a partially ordered set optimally any performed comparison should divide the set of possible total orderings approximately equally. This observation leads to the gradient method. First, perform a sequence of comparisons each dividing the set of possible orderings as equal as possible. Then process the obtained partial order via some simple algorithm.

The information-theoretic lower bound for insertion of a pair into an interval of length $n - 1$ is $\log(n(n-1)/2) < 2(\log n - 1/2)$. Let us imagine an idealistic situation. Assume that a pair may be inserted into an interval of length $Y(a) = (1-\lambda)\cdot 2^{a+1/2}$ with conditional complexity $2a$ for any sufficiently large a, where $\lambda \ge 0$ is independent of a parameter to be chosen later.

We are going to describe some gradient strategy to insert a pair $\beta_0 < \beta_1$ into an interval of length $Y(a)$. The strategy is parameterized by $s \in \mathbb{N}$. Note once more that we operate with real lengths of intervals. Rounding issues are covered by Lemma 2 converting strategy into algorithm.

First, consider the problem of the optimal choice for the next element to compare with. Suppose that in the current partial order, the element β_1 falls into the interval I_1, and the element β_0 falls into the interval $I_0 \cup I_1$, where $I_0 \cap I_1 = \emptyset$ and $|I_0| = b \cdot |I_1|$. Easy to check that the optimal choice to compare with β_0 is the element[5] α of rank $(b/2 + 1/4) \cdot |I_1|$ in $I_0 \cup I_1$. (If $b > 1/2$, then $\alpha \in I_0$.) The optimal choice to compare with β_1 is the element α' of rank $\sqrt{b^2 + b + 1/2} \cdot |I_1|$, see Fig. 3. Generally, if we want to divide the set of outcomes as $x : (1 - x)$, then to compare with β_0 and β_1 we should choose elements of ranks $x(b + 1/2) \cdot |I_1|$ and $\sqrt{b^2 + x(2b + 1)} \cdot |I_1|$, respectively.

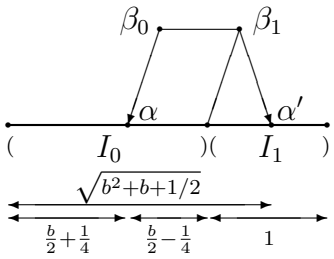

Figure 3: Optimal choices for the next comparison

6.1 Initial partition

W.l.o.g. assume that the first comparison is $\beta_1 ? \alpha_1$. Choose an element of rank $Y(a - 1/2) = (1 - \lambda) \cdot 2^a$ as α_1. If $\beta_1 < \alpha_1$, call the algorithm to insert the pair into the interval $(-\infty, \alpha_1)$ of length $Y(a - 1/2)$. Otherwise, perform the comparison $\beta_0 ? \alpha_2$, where α_2 is the element with optimal value of rank $(1 - \lambda) \cdot \frac{\sqrt{2}+1}{4} \cdot 2^a$. The reason to perform the second comparison with β_0 will be explained later.

If $\beta_0 < \alpha_2$, then the element β_0 is inserted into the interval $(-\infty, \alpha_2)$ with the conditional complexity $a - 2 + \log(\sqrt{2} + 1)$, and β_1 is inserted into the interval $(\alpha_1, +\infty)$ of length $(1 - \lambda)(\sqrt{2} - 1) \cdot 2^a$ with the conditional complexity $a + \log(\sqrt{2} - 1)$. Hence, in this case, we perform $2a$ conditional comparisons to insert the pair.

If $\beta_0 > \alpha_2$, then $\beta_1 \in I_1$ and $\beta_0 \in I_0 \cup I_1$, where $I_1 = (\alpha_1, +\infty)$ and $I_0 = (\alpha_2, \alpha_1)$. Denote $b = |I_0|/|I_1| = \frac{2\sqrt{2}+1}{4}$. With the use of s equally-dividing

[5]As before, here and below, by an element we mean some point to partition an interval.

comparisons we can locate β_1 in one of 2^s subintervals of the interval I_1. By the aforementioned formulas, the ends of these subintervals are the elements α'_k of ranks $b_k \cdot |I_1|$, $b_k = \sqrt{b^2 + k(2b+1) \cdot 2^{-s}}$, $k = 0, \ldots, 2^s - 1$, and the point $\alpha'_{2^s} = +\infty$. Denote $J_k = (\alpha'_{k-1}, \alpha'_k)$. The general partition scheme is shown in Fig. 4.

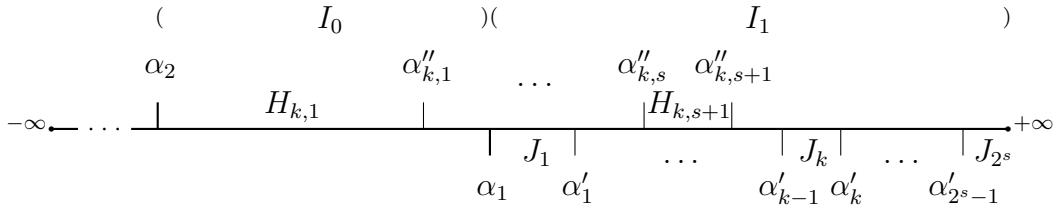

Figure 4: The universal partition

Assume that β_1 falls into J_k. The next stage is the insertion of β_0. We sequentially compare β_0 with elements $\alpha''_{k,1}, \ldots, \alpha''_{k,s+1}$, which are defined as follows. Set $\alpha''_{k,0} = \alpha_2$. Denote by $H_{k,j}$ the interval $(\alpha''_{k,j-1}, \alpha''_{k,j})$. The length of $H_{k,j}$ is chosen as

$$|H_{k,j}| = (1-\lambda) \cdot 2^{2a-s-2-j-\log(|J_k|/(1-\lambda))}. \tag{24}$$

If $\beta_0 > \alpha''_{k,1}$, perform the comparison $\beta_0 ? \alpha''_{k,2}$; then, if $\beta_0 > \alpha''_{k,2}$, compare β_0 and $\alpha''_{k,3}$, etc. As soon as $\beta_0 < \alpha''_{k,j}$, the elements β_0 and β_1 are inserted separately into the intervals $H_{k,j}$ and J_k with the cumulative conditional complexity $2a - s - 2 - j$. Therefore, overall conditional complexity of the pair's insertion is $2a$.

Finally, if $\beta_0 > \alpha''_{k,s+1}$, then both elements β_0, β_1 fall into $(\alpha''_{k,s+1}, \alpha'_k)$. We would like to insert the pair into this interval by $2a - 2s - 3$ comparisons. This would be possible, for example, in the case $\alpha''_{k,s+1} = \alpha_{k-1}$. Unfortunately, the latter equality does not generally hold, and the resulting interval appears to be longer than required. Let us estimate the excess of length. The length of the interval J_k is

$$|J_k| = (b_k - b_{k-1}) \cdot |I_1| = \frac{2b+1}{2^s(b_k + b_{k-1})} \cdot |I_1| = (1-\lambda) \cdot \frac{\sqrt{2}+1}{2^{s+1}(b_k + b_{k-1})} \cdot 2^a. \tag{25}$$

Then, (24) implies that

$$|H_{k,j}| = (1-\lambda) \cdot 2^{a-1-j-\log(\sqrt{2}+1)+\log(b_k+b_{k-1})} = \frac{b_k + b_{k-1}}{2^{j+1}} \cdot |I_1|. \tag{26}$$

Hence, the sum of lengths of intervals $H_{k,j}$ for a fixed k is

$$\sum_{j=1}^{s+1} |H_{k,j}| = (1 - 2^{-s-1}) \cdot \frac{b_k + b_{k-1}}{2} \cdot |I_1|. \tag{27}$$

Therefore,

$$|(\alpha''_{k,s+1}, \alpha'_k)| = b_k \cdot |I_1| - \sum_{j=1}^{s+1} |H_{k,j}| = \left(b_k - (1 - 2^{-s-1})(b_k + b_{k-1})/2\right) |I_1| =$$
$$= \left(\frac{b_k - b_{k-1}}{2} + \frac{b_k + b_{k-1}}{2^{s+2}}\right) |I_1| < \left(\frac{2b+1}{2^{s+1} \cdot 2b} + \frac{b+1}{2^{s+1}}\right) |I_1| < \frac{7}{2^{s+2}} \cdot |I_1|.$$

Consequently,

$$|(\alpha''_{k,s+1}, \alpha'_k)| - Y(a - s - 3/2) < \frac{7}{2^{s+2}} \cdot |I_1| - (1 - \lambda) \cdot 2^{a-s-1} =$$
$$= \frac{7 - 2(\sqrt{2}+1)}{2^{s+2}} \cdot |I_1| < \frac{\ln 2}{2^s} \cdot |I_1|. \tag{28}$$

We will make use of the obtained bound later. Now we are going to modify the above strategy by allowing a controlled aberration of the ideal, that is, applying a slightly non-equal partitioning. Namely, we will violate the distribution of lengths of subintervals J_k inside I_1 to decrease the estimated complexity of insertions into them, and as a consequence, to increase the allowable complexity for insertions into $H_{k,j}$, which makes the latter longer. To do this, we have a reserve, a simple bound (4) on the complexity $Q(a)$ of insertion of single elements, which is better than $Y(a - 1/2)$, when a is close to an integer from below[6]. The length of the interval I_1 will remain intact.

Observe, that the length of J_k varies from $\frac{2b+1}{2b} \cdot 2^{-s} \cdot |I_1|$ to $\frac{2b+1}{2b+2} \cdot 2^{-s} \cdot |I_1|$ when k grows, see (25). It follows from $b < 1$ that $\left[\frac{2}{3}, \frac{4}{3}\right] \subset \left[1 - \frac{1}{2b+1}, 1 + \frac{1}{2b+1}\right]$. Thus, for any (sufficiently large) a some intervals J_k have lengths close to the powers of 2. Now it is time to explain that the reason behind the second comparison $\beta_0 ? \alpha_2$ is precisely to reduce b below 1.

6.2 Correction of the partition

At this stage, we correct the strategy by violating lengths of J_k and $H_{k,j}$, and at the same time discretisize the problem. Let us denote the modified intervals

[6] One can hardly build a fast insertion algorithm not relying upon efficient binary insertions into intervals of lengths close to 2^k. We can see it in the known algorithms [6, 7] and in both examples from §3 and §5.

by J_k^* and $H_{k,j}^*$. Let $d \in 2\mathbb{N}$ be a discretization parameter, and set $y_i = i/d$, $i = 0, \ldots, d-1$. We still use the expression $Y(r+y_i) = (1-\lambda)\cdot 2^{r+y_i+1/2}$ to bound the length of an interval sufficient for insertion of a pair with conditional complexity $2(r+y_i)$.

After taking some $a = r + y_i$, $r \in \mathbb{N}$, we are ready to describe the modified partition. The parity of d allows us to keep the same choice of α_1 as in the idealistic strategy above. However, now we have to estimate the complexity of insertion of an element into the interval $I_1 = (\alpha_1, +\infty)$ as

$$a + \lceil d \cdot \log(\sqrt{2}-1)\rceil/d < a + \log(\sqrt{2}-1) + 1/d.$$

As a result, instead of α_2 we have to choose the element α_2^* of rank at most $(1-\lambda)\cdot 2^{a-2-\lceil d\cdot\log(\sqrt{2}-1)\rceil/d}$. Actually, to provide conditions for the future application of Lemma 2, we additionally save $1/d$ conditional comparisons, and choose as α_2^* the element of a slightly lesser rank

$$(1-\lambda)\cdot 2^{a-2-\lceil d\cdot\log(\sqrt{2}-1)\rceil/d-1/d} > (1-\lambda)\cdot\frac{\sqrt{2}+1}{4}\cdot 2^{a-2/d}. \tag{29}$$

Hence, with the use of the inequality $e^x \geq 1+x$, we obtain

$$|I_0^*| = |(\alpha_2^*, \alpha_1)| \leq (1-\lambda)\cdot 2^a\left(1 - \frac{\sqrt{2}+1}{4}\cdot 2^{-2/d}\right) \leq$$
$$\leq (1-\lambda)\cdot 2^a\left(\frac{3-\sqrt{2}}{4} + \frac{\sqrt{2}+1}{2d}\cdot\ln 2\right).$$

As a consequence,

$$0 \leq |I_0^*| - |I_0| \leq \frac{(3+2\sqrt{2})}{2d}\cdot\ln 2\cdot|I_1|. \tag{30}$$

For d large enough, $|I_0^*| < |I_1|$.

Let us redistribute intervals J_k inside I_1. Introduce a contraction parameter $\mu > 0$. We require the conditional complexity of insertion of an element into the interval J_k^* be smaller than that of insertion into J_k in the idealistic strategy at least by $\left(2 - \frac{k}{2^s}\right)\mu$ comparisons, $k = 1, \ldots, 2^s$. Recall that the complexity of insertion in the idealistic strategy is estimated as $\log(|J_k|/(1-\lambda))$. We resort to the non-uniform contraction because of the effect of migration of the right ends of intervals J_k. Since $|I_0| < |I_1|$, for the uniform contraction by $1+\mu$ times, it would be hard to compensate this for small numbers k. The total length of intervals $H_{k,j}$ would increase by approximately $\mu\cdot|I_0|$, and at the same time the rank of α'_{k+1} (the right end of the interval J_k) would increase by roughly $\mu\cdot|I_1|$. To overcome this obstacle, we choose smaller contraction coefficients for intervals with larger indices.

6.3 Contraction

We are going to shorten almost all intervals J_k, and to compensate the contraction by extending a small number of intervals with lengths close to the powers of 2. First, observe that b_k satisfies simple inequalities (which can be verified by squaring)

$$b + \frac{k}{2^s} \cdot b \le b_k \le b + \frac{k}{2^s} \cdot \frac{2b+1}{2b^2}. \tag{31}$$

Taking the discretization into account, the length of the contracted interval J_k^* may be chosen within the limits

$$2^{-\left(2-\frac{k}{2^s}\right)\mu-\frac{1}{d}} \cdot |J_k| \le |J_k^*| \le 2^{-\left(2-\frac{k}{2^s}\right)\mu} \cdot |J_k|. \tag{32}$$

Next, we estimate the overall shortening θ_p of intervals with indices from p to 2^s. Under the pessimistic assumption that all intervals are contracted, again exploiting the inequality $e^x \ge 1 + x$ together with (25) and (31), we obtain

$$\begin{aligned}
\theta_p &\le \sum_{k=p}^{2^s} (|J_k| - |J_k^*|) \le \sum_{k=p}^{2^s} |J_k| \left(1 - 2^{-\left(2-\frac{k}{2^s}\right)\mu-\frac{1}{d}}\right) \le \\
&\le \sum_{k=p}^{2^s} |J_k| \left(\left(2 - k2^{-s}\right)\mu + 1/d\right) \ln 2 = \\
&= \left(2\mu + \frac{1}{d}\right)(b + 1 - b_{p-1}) \ln 2 \cdot |I_1| - \mu 2^{-s} \ln 2 \cdot \sum_{k=p}^{2^s} k|J_k| = \\
&= \left(2\mu + \frac{1}{d}\right)(b + 1 - b_{p-1}) \ln 2 \cdot |I_1| - \mu 2^{-2s}(2b+1) \ln 2 \cdot \sum_{k=p}^{2^s} \frac{k}{b_k + b_{k-1}} |I_1| \le \\
&\le \left(2\mu\left(1 - \frac{p-1}{2^s} \cdot b\right) - \mu 2^{-2s} \cdot \frac{2b+1}{2(b+1)} \cdot \frac{2^{2s} - p^2}{2} + \frac{1}{d}\right) \ln 2 \cdot |I_1| = \\
&= \left(\left(2 - 2b \cdot \frac{p-1}{2^s} - \frac{2b+1}{4(b+1)} \cdot \left(1 - \frac{p^2}{2^{2s}}\right)\right)\mu + \frac{1}{d}\right) \ln 2 \cdot |I_1|.
\end{aligned}$$

In particular, the overall shortening of all intervals J_k may be bounded as

$$\theta_1 \le (2\mu + 1/d) \ln 2 \cdot |I_1|, \tag{33}$$

and for $p \ge 2$, the bound

$$\theta_p \le \left(\left(2 - \frac{2b+1}{4(b+1)}\right)\mu + \frac{1}{d}\right) \ln 2 \cdot |I_1|, \tag{34}$$

holds, since $p^2/2^s \le p \le 2(p-1)$ and $\frac{2b+1}{4(b+1)} < b = \frac{2\sqrt{2}+1}{4}$.

6.4 Extension

Now, we bound from below the size of possible compensation. If

$$(1-\lambda)\cdot 2^{l+2\mu} \le |J_k| < 2^l, \quad l \in \mathbb{N},$$

then the idealistic strategy above estimates the complexity of insertion of the element β_1 into the interval J_k at least as $l+2\mu$. Nevertheless, the length of the interval may be increased to 2^l, and at the same time, the bound on the complexity decreases at least by 2μ, as desirable.

First, observe that for sufficiently small λ and μ, it holds for some $l \in \mathbb{N}$ that

$$|J_{2^s}| \le (1-\lambda)\cdot 2^{l+2\mu} < 2^l \le |J_1|. \tag{35}$$

In the case $\mu \ll \lambda$ the middle inequality indeed holds true, then (35) follows from $|J_1|/|J_{2^s}| \ge \frac{2}{1-\lambda}$. Applying (25) and (31) we deduce

$$\frac{|J_1|}{|J_{2^s}|} = \frac{b_{2^s}+b_{2^s-1}}{b_1+b_0} \ge \frac{2b+2-\frac{b}{2^s}}{2b+\frac{2b+1}{2b^2\cdot 2^s}} \ge$$
$$\ge \frac{b+1}{b}\left(1-\frac{b}{2(b+1)\cdot 2^s}\right)\left(1-\frac{2b+1}{4b^3\cdot 2^s}\right) > \frac{b+1}{b}\left(1-2^{1-s}\right),$$

where are also used simple inequalities $\frac{1}{1+x} \ge 1-x$ and $(1-x)(1-y) \ge 1-x-y$ valid for $x, y \ge 0$. It is easy to check by direct numeric calculation, that this bound prevails over $\frac{2}{1-\lambda}$, say, for any $s \ge 10$ and $\lambda \le 1/50$.

So, we have that some segment $g = \left[(1-\lambda)\cdot 2^{l+2\mu},\, 2^l\right]$ lies inside $[|J_{2^s}|, |J_1|]$. Let us bound the size of extension only for those intervals J_k, whose lengths fall into g. Recall that lengths of intervals monotonically decrease while k grows, see (25). The length of the segment g may be bounded from below as

$$\left(1-(1-\lambda)2^{2\mu}\right)\cdot 2^l \ge (1-(1-\lambda)(1+2\mu))\cdot |J_{2^s}| \ge$$
$$\ge L = (\lambda-2\mu)\,\frac{2b+1}{2(b+1)\cdot 2^s}\cdot |I_1| \tag{36}$$

due to (25) and the inequality $2^x \le 1+x$ valid for $0 \le x \le 1$.

Now, we may bound the difference of lengths of adjacent intervals as

$$\frac{2^s}{(2b+1)\cdot |I_1|}\cdot(|J_k|-|J_{k+1}|) = \frac{1}{b_k+b_{k-1}} - \frac{1}{b_{k+1}+b_k} \le$$
$$\le \frac{b_{k+1}-b_{k-1}}{4b^2} = \frac{2(2b+1)}{4b^2(b_{k+1}+b_{k-1})\cdot 2^s} \le \frac{2b+1}{4b^3\cdot 2^s}.$$

Consequently, for any k,

$$|J_k| - |J_{k+1}| \le \Delta = \frac{(2b+1)^2}{4b^3 \cdot 2^{2s}} \cdot |I_1|. \tag{37}$$

It is easy to verify, that if a set of points partitions a segment of length L by subsegments with lengths at most Δ, then the sum of distances from these points to the end of the segment is at least $\frac{L(L-\Delta)}{2\Delta}$. Applying this formula with actual values of L and Δ, we deduce that

$$\begin{aligned}\frac{L(L-\Delta)}{2\Delta} &\ge \left((\lambda-2\mu)^2 \cdot \frac{b^3}{2(b+1)^2} - (\lambda-2\mu)\cdot\frac{2b+1}{4(b+1)\cdot 2^s}\right)|I_1| = \\ &= (\lambda-2\mu)\left(\lambda-2\mu-\frac{(2b+1)(b+1)}{2b^3\cdot 2^s}\right)\frac{b^3}{2(b+1)^2}\cdot|I_1| > \\ &> (\lambda-2\mu)\left(\lambda-2\mu-2^{2-s}\right)\frac{b^3}{2(b+1)^2}\cdot|I_1|.\end{aligned}$$

So we lower bounded the maximal possible extension of intervals J_k with lengths from g (to the length 2^l).

The compensation for the contraction of the rest intervals J_k is possible when $\frac{L(L-\Delta)}{2\Delta} \ge \theta_1$. By (33), we can state the sufficient condition for this as

$$(\lambda-2\mu)\left(\lambda-2\mu-2^{2-s}\right)\frac{b^3}{2(b+1)^2} \ge \left(2+\frac{1}{d}\right)\ln 2\cdot\mu. \tag{38}$$

For a moment, we are quite satisfied by the following observation: for s large enough, one can choose parameters as $d \asymp 2^s$, $\lambda \asymp 2^{-s/2}$, $\mu \asymp 2^{-s}$ so that the condition (38) is fulfilled.

6.5 Putting things together

Now it is the time to examine whether the undertaken measures eliminate the deficit of length of intervals $H_{k,j}$ expressed in (28). Note that the conditions behind the bound (28) have changed. Though we can still use the old expression for $Y(a-s-3/2)$, the interval I_0 has been extended, see (30), and the right end of the interval J_k may have migrated further to the right by a distance bounded from above as θ_{k+1}, see (34).

Thus, for any k, we require the overall growth of intervals $H_{k,j}$ to be at least

$$|I_0^*| - |I_0| + \theta_{k+1} + \frac{\ln 2}{2^s}\cdot|I_1| + \delta\cdot|I_1|, \tag{39}$$

where $\delta\cdot|I_1|$ is the extra reserve of length provided for the future application of Lemma 2.

By construction, $|H^*_{k,j}| \geq |H_{k,j}| \cdot 2^{\left(2-\frac{k}{2^s}\right)\mu-\frac{1}{d}}$, if we remember about discretization. Therefore, by (27) and (31), we obtain

$$\sum_{j=1}^{s+1}(|H^*_{k,j}| - |H_{k,j}|) \geq \left(2^{\left(2-\frac{k}{2^s}\right)\mu-\frac{1}{d}} - 1\right) \cdot \sum_{j=1}^{s+1} |H_{k,j}| \geq$$

$$\geq \left(\left(2-\frac{k}{2^s}\right)\mu - \frac{1}{d}\right) \ln 2 \cdot \left(1 - \frac{1}{2^{s+1}}\right) \cdot \frac{b_k + b_{k-1}}{2} \cdot |I_1| \geq$$

$$\geq \left(2-\frac{k}{2^s}\right)\left(1-\frac{1}{2^{s+1}}\right)\left(1+\frac{k-\frac{1}{2}}{2^s}\right)\mu b \cdot \ln 2 \cdot |I_1| - \frac{b+1}{d} \cdot \ln 2 \cdot |I_1| \geq$$

$$\geq \left(2+\frac{k-2}{2^s}-\frac{k^2}{2^{2s}}\right)\mu b \cdot \ln 2 \cdot |I_1| - \frac{b+1}{d} \cdot \ln 2 \cdot |I_1| \geq$$

$$\geq \left(\left(2-2^{1-s}\right)\mu b - \frac{b+1}{d}\right) \cdot \ln 2 \cdot |I_1|.$$

This bound together with (30) and (34) allows to state the sufficient for (39) condition as

$$\left(2-2^{1-s}\right)\mu b - \frac{b+1}{d} \geq \frac{(3+2\sqrt{2})}{2d} + \left(2 - \frac{2b+1}{4(b+1)}\right)\mu + \frac{1}{d} + \frac{1}{2^s} + \delta \cdot \log e. \quad (40)$$

Let $d = 2^s$ and $\delta = 2^{-s}$. Next, to satisfy (40), we choose $\mu = 30 \cdot 2^{-s}$ (assuming that $s \geq 10$). Then, the condition (38) is fulfilled if, for example, $\lambda = 20 \cdot 2^{-s/2} + 64 \cdot 2^{-s}$. The particular condition $\lambda \leq 1/50$ securing (35) is satisfied by any $s \geq 20$.

6.6 Resulting strategy

By fixing the chosen values of parameters d, δ, μ, λ, we obtain the comparison strategy with a system of restrictions denoted by $\sigma[s]$. Comparisons are performed as described above. For insertions into subintervals we require the complexity provided by the above calculations. Let us describe the system of restrictions of the given strategy.

Let $p(a)$ be an estimate of the length of an interval with the pair's insertion conditional complexity $2a$, and $p_c(J)$ be an estimate of the length of an interval with the element's insertion conditional complexity larger by c than for the insertion into some interval J of the proposed partition. To estimate the complexity of insertion into intervals J we choose either bounds of the form $Q(a)$, or bounds of the form $p(a)$, depending on a.

Due to the choice of α_1, the first comparison does not generate restrictions. To support the second comparison, we introduce a restriction

$$p(a) \leq p(a - 1/2) + p_{1/d}(I_1). \quad (41)$$

We rely here on the reserve of the complexity provided by the choice of α_2^*, see (29). It allows insertion into the interval I_1 to utilize a bit more complexity. This is merely an artificial trick to formally satisfy the requirements of Lemma 2.

The following s comparisons locating β_1 inside some of the intervals J_k^* correspond to inner vertices of the comparison tree. To provide the required complexity for insertion of the element β_1 into any of such intervals, we introduce a restriction

$$p(a) \le p(a - 1/2) + \sum_{k=1}^{2^s} p_0(J_k^*). \tag{42}$$

The restriction is single, since we can choose the lengths of the intervals J_k^*, $k > 1$, freely by appropriate selection of the boundary elements $\alpha_k'^*$. Then, only the length of the interval J_1^* is expressed through the lengths of other intervals.

Also, we can freely choose the lengths of the intervals $H_{k,j}^*$ (just take an appropriate element for comparison). Only the final insertions after $2s + 3$ comparisons generate restrictions

$$p(a) \le p_0(I_{-1}) + \sum_{j=1}^{s+1} p_0(H_{k,j}^*) + p(a - s - 3/2) + \sum_{i=k+1}^{2^s} p_0(J_i^*), \tag{43}$$

where $I_{-1} = (-\infty, \alpha_2^*)$.

Finally, (41), (42) and (43) for various values $a = r + i/d$ constitute the system of restrictions $\sigma[s]$. To rewrite the system in a canonic form from the definition of strategy, we should replace $p(x)$ and $p_c(J)$ in the said inequalities by $p_{r,i}$ or by numeric expressions, when the bound $Q(a)$ is implied.

The depth of the described strategy is $2s + 3$. Now we will check that the width (of the strategy, and of the system $\sigma[s]$ as well) does not exceed $s + 4$. Indeed,

$$\frac{|I_1|}{p(a)} = \sqrt{2} - 1 > \frac{1}{2^2}, \qquad \frac{|I_{-1}|}{p(a)} \ge \frac{\sqrt{2} + 1}{4 \cdot 2^{2/d}} > \frac{1}{2}$$

by (29). (Recall that $p(a) = Y(a)$.) Further, from (32) and (25) we derive

$$\frac{|J_k^*|}{p(a)} \ge \frac{|J_k|}{p(a) \cdot 2^{2\mu+1/d}} \ge \frac{\sqrt{2} + 1}{(b + 1) \cdot 2^{s+2+2\mu+1/d}} > \frac{1}{2^{s+2}}.$$

At last, it follows from (26) that

$$\frac{|H_{k,s+1}^*|}{p(a)} \ge \frac{|H_{k,s+1}|}{p(a)} \ge \frac{b}{2^{s+1}} \cdot \frac{|I_1|}{p(a)} > \frac{1}{2^{s+3}}.$$

Therefore, if we determine r' from $p(a) = p_{r',i'}$, then after representing (41)–(43) in terms of $p_{r,i}$, the obtained restrictions contain only those $p_{r,i}$ with $r \geq r' - s - 3$.

Lemma 4. *Suppose $s \geq 20$, $d = 2^s$, $\mu = 30 \cdot 2^{-s}$, $\lambda = 20 \cdot 2^{-s/2} + 64 \cdot 2^{-s}$ and $\varepsilon = \frac{1}{25 \cdot 2^s}$. Also let $y_i = i/d$ and $v_i = (1-\lambda) \cdot 2^{y_i + 1/2}$. Then*
(i) $v = (v_0, \ldots, v_{d-1}) \in T(\sigma[s])$;
(ii) $u = v/2 \in T^{\varepsilon}(\sigma[s])$.

Proof. The statement (i) is already proven, since the strategy itself is built to satisfy it. We are left to prove (ii).

Recall that to obtain the canonic form (10) of the system of restrictions, we rewrite all inequalities in terms of $x_{r,i} = p_{r,i} \cdot 2^{-r}$ and normalize them. To pass on to the reduced system (11), we perform the substitution $x_{r,i} = z_i$.

Let $p(a) = p_{r',i'}$. First, we preliminarily normalize restrictions (41)–(43) via multiplication by $2^{-r'}$, and rewrite them in terms of $x_{r,i}$. We refer to the obtained inequalities as (41')–(43'). Now we estimate the sum of the absolute values of the coefficients for $x_{r,i}$ in each of the restrictions to finally determine normalizing coefficients.

The contribution from the summand $p(a)$ to every sum of coefficients is 1. Further, each of the summands $p(a - 1/2)$, $p_{1/d}(I_1)$, $p_0(I_{-1})$ and $p_0(H_{k,1})$ contributes at most 1, the contribution from $p(a - s - 3/2)$ does not exceed 2^{-s-1}, and for any j, the contribution from $p_0(H_{k,j+1})$ is the half of the contribution from $p_0(H_{k,j})$. Since $|J_k^*| \leq |J_1| \leq 2^{a-s}$ by (25), the contribution from any $p_0(J_k^*)$ is at most 2^{-s}. As a result, the sums of coefficients for $x_{r,i}$ in the inequalities (41'), (42') and (43') are bounded from above by 3, 3 and 5, respectively.

Now we estimate a reserve for the vector u to satisfy the reduced inequalities (41')–(43'). By construction, for the main solution v the reduced inequality (41') would hold even after replacing $p_{1/d}(I_1)$ by $p_0(I_1)$ in (41). Therefore, in the case $p_{r,i} = v_i \cdot 2^r$, the right-hand side of (41) is larger than the left-hand side at least by

$$p_{1/d}(I_1) - p_0(I_1) \geq |I_1|(2^{1/d} - 1) \geq \frac{\ln 2}{d} \cdot |I_1| = \\ = (1-\lambda)(\sqrt{2} - 1) \ln 2 \cdot 2^{a-s} > 2^{a-s-2}.$$

After the normalization (that is, division by at most $3 \cdot 2^a$) and replacing v by u, the gap between the sides of the reduced inequality (41') decreases by at most $6 \cdot 2^a$ times. So, the resulting gap is at least $\frac{1}{24 \cdot 2^s} > \varepsilon$.

Let us estimate the constant term appearing in (42') due to the fact that we use the bound $Q(a)$ for the complexity of insertions into some intervals J_k^*. The

number of such intervals is at least $\lfloor L/\Delta \rfloor$, see above, and the applied bound lies within the limits $2^l \geq |J_{2^s}| \geq 2^{a-s-2}$ by (35) and (25). Thus, applying actual values of L and Δ from (36), (37), we deduce that the constant in the right-hand side of (42') is at least

$$\frac{L-\Delta}{\Delta \cdot 2^{s+2}} = (\lambda - 2\mu) \cdot \frac{b^3}{2(b+1)(2b+1)} - \frac{1}{2^{s+2}} > \frac{1}{2^{s/2}}.$$

After replacing v by u in the reduced inequality (42'), all non-constant terms are halved. Then, we can safely halve the constant term as well. So, if we take the normalizing into account, the term can be safely reduced to $\frac{1}{6 \cdot 2^{s/2}} > \varepsilon$.

For $p_{r,i} = v_i \cdot 2^r$, in each of the inequalities (43), the right-hand side prevails over the left-hand side by at least

$$\delta \cdot |I_1| = (1-\lambda)\delta(\sqrt{2}-1) \cdot 2^a > 2^{a-s+1}/5$$

due to the reserve provided for the sum of lengths of intervals $H^*_{k,j}$, see (39). After normalizing (that is, division by at most $5 \cdot 2^a$) and replacing v by u, the gap decreases by at most $10 \cdot 2^a$ times, and in the reduced inequality (43') it appears to be not smaller than $\varepsilon = \frac{1}{25 \cdot 2^s}$.

Hence, after decreasing of constant terms in the reduced inequalities (41')–(43') by ε, the vector u delivers the solution of the system. □

7 Fast sorting

Theorem 2. *For any $a \geq 2$ and $2^{a/4} \leq m \leq 2^{a/2}$,*

$$P_m(a) \geq 2^{a+1/2} - O\left(a^{-\gamma} \cdot 2^a\right), \tag{44}$$

where γ is small enough, say, $\gamma = \frac{1}{5}$.

Proof. We will apply Lemma 2 to the result of Lemma 4. The values of all parameters are borrowed from Lemma 4. We have $v \in T(\sigma[s])$ and $u \in T^\varepsilon(\sigma[s])$ for the system of restrictions $\sigma[s]$ of depth $h = 2s+3$ and width $w \leq s+4$. Note that $u_i < 2^{y_i - 1/2} < \pi(y_i)$, see (16). We also have $\epsilon < 1-\lambda < R = ||v-u|| < \sqrt{2}$.

For $a < 100$, the inequality (44) holds with sufficiently large values of the constant hidden behind the symbol "O". Thus, we may assume $a \geq 100$. Set $\varphi = \log\left(\frac{m\varepsilon}{4a^2}\right)$ and $s = \lceil 2\gamma \log a \rceil$. If $\gamma \leq 1/4$, then $m\varepsilon \geq 2^{a/4}/(50 \cdot a^{2\gamma}) \geq 4a^2$, hence, $\varphi > 0$. The condition (17) in Lemma 2 is automatically satisfied.

Let $r + y_i \le a - \frac{(a+2)\varphi \cdot 2^\varphi}{m} \le r + y_i + \frac{1}{d}$, where $r \in \mathbb{N}$. By Lemma 2, we derive the bound

$$P_m(a) \ge P_m\left(r + y_i + \frac{(r+2)\varphi \cdot 2^\varphi}{m}\right) \ge$$

$$\ge v_i \cdot 2^r - Rm \cdot 2^{\varphi+2}/\varepsilon - R \cdot 2^r \cdot e^{-\frac{\varepsilon}{2R}\left(\frac{\varphi}{w(d+1)(h+1)}-2\right)} \ge$$

$$\ge \left(1 - O\left(2^{-s/2}\right)\right) \cdot 2^{a+\frac{1}{2}-\frac{(a+2)\varphi\cdot 2^\varphi}{m}-\frac{1}{d}} - O\left(m^2/a^2\right) - O\left(2^a \cdot e^{-\Theta\left(a\cdot s^{-2}\cdot 2^{-2s}\right)}\right) =$$

$$= \left(1 - O\left(2^{-s/2}\right)\right) \cdot 2^{a+1/2-O(2^{-s})} - O(2^a/a^2) - 2^{a-O\left(a\cdot s^{-2}\cdot 2^{-2s}\right)} =$$

$$= 2^{a+1/2} \cdot \left(1 - O\left(a^{-\gamma}\right)\right),$$

when, for example, $\gamma \le 1/5$. □

Corollary 2. *For any $a \ge 2$ and $2^{a/4} \le m \le 2^{a/2}$,*

$$Q_{2m}(a) \ge 2^a - O\left(a^{-1/5} \cdot 2^a\right).$$

We note that already a simplified version of the strategy from §6 with the choice of $s = 1$ and depth 6 would allow us to improve the upper bound of the method of binary insertions [5] for all sufficiently large n. However, to achieve this, we would have to sacrifice universality, and carefully select optimal partitions for any i instead. To get a better bound, we have to choose s growing.

Theorem 3. *For any n,*

$$S(n) = \log(n!) + O\left(n \log^{-1/5} n\right).$$

Proof. We start as in the method of binary insertions. Divide n inputs into pairs, order them, and sort the larger elements. The latter constitute the principal chain. To do this, we require $n/2 + S(n/2)$ comparisons. We keep notation α_i for the smaller elements in the pairs from Fig. 1.

Next, we proceed with the Ford–Johnson method to insert first $n_0 = n/\log n$ of the elements α_i into the principal chain by $n + O(n_0)$ comparisons. We split the remaining elements into groups of size $m \approx \sqrt{n}/\log n$. The group number k contains elements $\alpha_{n_0+(k-1)m+1}, \ldots, \alpha_{n_0+km}$. Groups are inserted into the principal chain in turn in ascending order of numbers by the method of Corollary 2. The last group may contain less than m elements, so we use trivial binary method to insert them.

The group number t should be inserted into an interval of length $L_t = 2(n_0 + tm) - m$. By Corollary 2, it holds for some $\rho_t = \log L_t + O(\log^{-1/5} n)$

that $Q_m(\rho_t) \geq L_t$. So, the complexity of insertion of the t-th group does not exceed $\rho_t \cdot m$.

Imagine idealistic situation, when any element α_k may be inserted into the principal chain through $\log(2k-1)$ comparisons. In this case, to insert all elements α_k we would use

$$\log \prod_{k=1}^{\lceil n/2 \rceil} (2k-1) = \log(n!) - \log((n/2)!) - n/2 + O(\log n) \tag{45}$$

comparisons.

In the group insertion method, in order to insert an element α_k, we perform at most $\log(2k+m) + O(\log^{-1/5} n) = \log(2k-1) + O(\log^{-1/5} n)$ comparisons for $n_0 < k < n/2 - m$, and at most $\log(2k-1) + O(1)$ comparisons for $k \leq n_0$ and $k \geq n/2 - m$. So, the excess number of comparisons with respect to (45) may be estimated as

$$O(n_0 + m) + O\left((n/2 - n_0) \log^{-1/5} n\right).$$

Finally, we obtain a recurrent relation

$$S(n) \leq n/2 + S(n/2) + \log(n!) - \log((n/2)!) - n/2 + O\left(n \log^{-1/5} n\right),$$

or, if rewritten in terms of $s(n) = S(n) - \log(n!)$,

$$s(n) \leq s(n/2) + O\left(n \log^{-1/5} n\right).$$

Thus, it trivially follows that $s(n) = O\left(n \log^{-1/5} n\right)$. □

Acknowledgements

The author would like to thank Stasys Jukna for a number of helpful comments.

The research is supported by RFBR grant, project no. 19-01-00294a.

References

[1] *Aho A.V., Hopcroft J.E., Ullman J.D.* The design and analysis of computer algorithms. Reading, Mass.: Addison-Wesley, 1976.

[2] *Christen C.* Improving the bounds for optimal merging // in: Proc. 19th IEEE Conf. on Found. of Comput. Sci. (Ann Arbor, USA, 16–18 October 1978). NY: IEEE, 1978, 259–266.

[3] *Dor D., Zwick U.* Selecting the median // SIAM J. Comput. 1999. **28**(5), 1722–1758.

[4] *Edelkamp S., Weiß A., Wild S.* QuickXsort: a fast sorting scheme in theory and practice // Algorithmica. 2020. **82**, 509–588.

[5] *Ford L.R., Johnson S.M.* A tournament problem // Amer. Math. Monthly. 1959. **66**(5), 387–389.

[6] *Graham R.L.* On sorting by comparisons // in: Computers in Number Theory. London: Academic Press, 1971, 263–269.

[7] *Hwang F.K., Lin S.* Optimal merging of 2 elements with n elements // Acta Inf. 1971. **1**, 145–158.

[8] *Iwama K., Teruyama J.* Improved average complexity for comparison-based sorting // Theor. Comput. Sci. 2020. **807**, 201–219.

[9] *Knuth D.E.* The art of computer programming. Vol. 3. Sorting and searching. Reading, Mass.: Addison-Wesley, 1998.

[10] *Manacher G.K., Bui T.D., Mai T.* Optimum combinations of sorting and merging // J. ACM. 1989. **36**(2), 290–334.

[11] *Mehlhorn K.* Data structures and algorithms. Vol. 1. Sorting and searching. Berlin, NY: Springer, 1984.

[12] *Peczarski M.* The Ford-Johnson algorithm still unbeaten for less than 47 elements // Inf. Process. Lett. 2007. **101**(3), 126–128.

[13] *Schönhage A., Paterson M., Pippenger N.* Finding the median // J. Comp. Sys. Sci. 1976. **13**, 184–199.

[14] *Schulte Mönting J.* Merging of 4 or 5 elements with n elements // Theor. Comput. Sci. 1981. **14**, 19–37.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il