Decision Trees with Minimum Average Depth for Sorting Eight Elements

Hassan AbouEisha, Igor Chikalov, Mikhail Moshkov*

Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia

Abstract

We prove that the minimum average depth of a decision tree for sorting 8 pairwise different elements is equal to 620160/8!. We show also that each decision tree for sorting 8 elements, which has minimum average depth (the number of such trees is approximately equal to 8.548×10^{326365}), has also minimum depth. Both problems were considered by Knuth in [6]. To obtain these results, we use tools based on extensions of dynamic programming which allow us to make sequential optimization of decision trees relative to depth and average depth, and to count the number of decision trees with minimum average depth.

Keywords: sorting, decision tree, average depth

1. Introduction

The problem of sorting n pairwise different elements from a linearly ordered set is one of the model problems in algorithm theory [6]. For solving this problem, we use binary decision trees [4, 8] where each step is a comparison of two elements. The minimum number of nodes in such a tree is equal to $2(n!) - 1$. For $n = 2, \ldots, 11$, the minimum depth of a decision tree for sorting n elements is equal to the well known lower bound $\lceil \log_2(n!) \rceil$ [10]. For $n = 2, 3, 4, 5, 6, 9, 10$, the minimum average depth of a decision tree for sorting n elements is equal to the known lower bound $\varphi(n)/n!$, where $\varphi(n) = (\lceil \log_2 n! \rceil + 1) \cdot n! - 2^{\lceil \log_2 n! \rceil}$ is the minimum external path length in

*Corresponding author. Tel.: +966 12 808 0334; fax: +966 12 802 1291.
Email address: mikhail.moshkov@kaust.edu.sa (Mikhail Moshkov)

an extended binary tree with $n!$ terminal nodes [6]. Césary [3] proved that, for $n = 7$ and $n = 8$, there are no decision trees for sorting n elements whose average depth is equal to $\varphi(n)/n!$. Kollár [7] found that the minimum average depth of a decision tree for sorting 7 elements is equal to $62416/7!$. We find that the minimum average depth of a decision tree for sorting 8 elements is equal to $620160/8!$.

Another open problem considered by Knuth [6] is the existence of decision trees for sorting n elements which have simultaneously minimum average depth and minimum depth. As it was mentioned by Knuth in [6], if a decision tree for sorting n elements has average depth equal to $\varphi(n)/n!$ then this tree has depth equal to $\lceil \log_2 n! \rceil$. Therefore, for $n = 2, 3, 4, 5, 6, 9, 10$, each decision tree for sorting n elements, which has minimum average depth, has also minimum depth. We extended this result to the cases $n = 7$ (Kollár in [7] did not consider this question) and $n = 8$. For $n = 2, \ldots, 8$, we counted also the number of decision trees for sorting n elements which have minimum average depth. In particular, for $n = 8$, the number of such trees is approximately equal to 8.548×10^{326365}. We recalculate known values of the minimum depth for $n = 2, \ldots, 8$ and minimum average depth for $n = 2, \ldots, 7$ to make sure that the new results are valid.

To obtain these results, we use tools based on extensions of dynamic programming [2, 5, 9] which allow us to make sequential optimization of decision trees relative to depth and average depth, and to count the number of decision trees with minimum average depth. The considered algorithms are not, of-course, brute-force algorithms (it is impossible to work directly with 8.548×10^{326365} optimal trees for $n = 8$). However, they require the work with big number of subproblems (for $n = 8$, the number of subproblems is equal to $431,723,379$). We describe the notion of subproblems in Section 2. The construction of the directed acyclic graph (DAG) representing the ordering of subproblems is discussed in Section 3.1.

The functions provided by the system DAGGER depend on the number of nodes and edges of the DAG corresponding to the input problem. From empirical study, our system can work with DAGs of size around fifty million nodes.

The paper consists of four sections. Section 2 contains main results, Section 3 – description of tools, and Section 4 – short conclusions.

2. Main Results

Let $x_1, \ldots, x_n$ be pairwise different elements from a linearly ordered set. We should find a permutation $(p_1, \ldots, p_n)$ from the set P_n of all permutations of the set $\{1, \ldots, n\}$ such that $x_{p_1} < \ldots < x_{p_n}$. Each nonempty subset Q of the set P_n can be considered as a subproblem of the initial sorting problem P_n with inputs $x_1, \ldots, x_n$ for each of which there exists a permutation $(p_1, \ldots, p_n) \in Q$ such that $x_{p_1} < \ldots < x_{p_n}$. We give all required definitions not for P_n but for an arbitrary subset (subproblem) Q of P_n.

We denote by $I(n)$ the set of all inequalities of the form $x_i < x_j$ such that $(i, j) \in \pi(n) = \{(i, j) : 1 \leq i, j \leq n,\ i \neq j\}$. We say that the permutation $p = (p_1, \ldots, p_n)$ is compatible with the inequality $x_i < x_j$ if and only if i precedes j in p. For $s_1, \ldots, s_m \in I(n)$, we denote by $Q(s_1) \ldots (s_m)$ the set of all permutations from Q which are compatible with all inequalities $s_1, \ldots, s_m$.

For solving the subproblem Q, we use binary decision trees in which terminal nodes are labeled with permutations from Q. Each nonterminal node is labeled with a comparison $x_i : x_j$ of two elements where $(i, j) \in \pi(n)$. Two edges start in this node which are labeled with results of the comparison $x_i < x_j$ and $x_j < x_i$, respectively.

We denote by $E(Q)$ the set of comparisons $x_i : x_j$ such that $(i, j) \in \pi(n)$, $Q(x_i < x_j) \neq \emptyset$ and $Q(x_j < x_i) \neq \emptyset$.

Let Γ be a decision tree and v be a node of Γ. We denote $Q(v) = Q(s_1) \ldots (s_m)$ where $s_1, \ldots, s_m$ are all inequalities attached to the edges in the path from the root of Γ to v (if v is the root of Γ then $Q(v) = Q$).

We will say that Γ solves the subproblem Q if each node v of Γ satisfies the following conditions:

1. If $|Q(v)| = 1$ and $Q(v) = \{p\}$ then v is a terminal node labeled with the permutation p;
2. If $|Q(v)| > 1$ then the node v is a nonterminal node which is labeled with a comparison $x_i : x_j$ from the set $E(Q(v))$.

We consider three cost functions for decision trees. Let Γ be a decision tree for solving the subproblem Q. We denote by $h(\Gamma)$ the depth of Γ which is the maximum length of a path from the root to a terminal node, by $l(\Gamma)$ – the external path length in Γ (the sum of lengths of all paths from the root to terminal nodes of Γ), and by $h_{avg}(\Gamma)$ – the average depth of Γ which is equal to $l(\Gamma)/|Q|$ (one can show that each decision tree for solving Q has $|Q|$ terminal nodes).

We denote by $h(Q)$ the minimum depth, by $l(Q)$ – the minimum external path length, and by $h_{avg}(Q)$ – the minimum average depth of decision trees for solving the subproblem Q. Note that $h_{avg}(Q) = l(Q)/\,|Q|$.

For $n = 2, \ldots, 8$, the values $h(P_n)$ and $h_{avg}(P_n) = l(P_n)/n!$ can be found in Tables 1 and 2. For the considered values of n, the parameter $h(P_n)$ is equal to its lower bound $\lceil \log_2(n!) \rceil$, and the parameter $h_{avg}(P_n)$ is equal to its lower bound $\varphi(n)/n!$ for $n = 2, \ldots, 6$.

n	2	3	4	5
$h(P_n)$	1	3	5	7
$h_{avg}(P_n)$	2/2!	16/3!	112/4!	832/5!
$\varphi(n)/n!$	2/2!	16/3!	112/4!	832/5!
$\lvert Opt_{h_{avg}}(P_n)\rvert$	1	12	27,744	2,418,647,040
$\lvert\mathcal{P}(n)\rvert$	3	19	219	4231

Table 1: Results for sorting $n = 2, 3, 4, 5$ elements.

Let Γ be a decision tree for solving the subproblem Q and ψ be one of the cost functions h, l, h_{avg}. We will say that Γ is optimal relative to ψ if $\psi(\Gamma) = \psi(Q)$. We denote by $Opt_\psi(Q)$ the set of decision trees for the subproblem Q which are optimal relative to ψ.

It is clear that $Opt_{h_{avg}}(P_n) = Opt_l(P_n)$. Based on results of computer experiments we obtain that $Opt_{h_{avg}}(P_n) \subseteq Opt_h(P_n)$ for $n = 2, \ldots, 8$. For $n = 2, \ldots, 8$, we count also the cardinality of the set $Opt_{h_{avg}}(P_n)$.

n	6	7	8
$h(P_n)$	10	13	16
$h_{avg}(P_n)$	6896/6!	62416/7!	620160/8!
$\varphi(n)/n!$	6896/6!	62 368/7!	619 904/8!
$\lvert Opt_{h_{avg}}(P_n)\rvert$	1.968×10^{263}	4.341×10^{6681}	8.548×10^{326365}
$\lvert\mathcal{P}(n)\rvert$	130,023	6,129,859	431,723,379

Table 2: Results for sorting $n = 6, 7, 8$ elements.

A nonempty subproblem $Q \subseteq P_n$ is called a separable subproblem of P_n if there exists a subset I of the set of inequalities $I(n)$ such that Q is the set of all permutations from P_n which are compatible with each inequality from I. In particular, $Q = P_n$ if $I = \emptyset$. We denote by $\mathcal{P}(n)$ the set of all separable

subproblems of P_n. The cardinality of the set $\mathcal{P}(n)$ for $n = 1, \ldots, 8$ can be found in Tables 1 and 2.

All computations were done using our software system Dagger [1] for optimization of decision trees and rules. This system is based on extensions of dynamic programming that allow us to describe the set of all decision trees for the initial problem, to make sequential optimization relative to different cost functions and to count the number of optimal decision trees for some cost functions. The work of Dagger involves the construction and transformations of a directed acyclic graph whose nodes are subproblems of the initial problem. In the case of sorting n elements, the set of nodes coincides with the set $\mathcal{P}(n)$ of separable subproblems of P_n.

3. Tools

3.1. Graph Δ_n and its Proper Subgraphs

We now consider an algorithm for the construction of a directed acyclic graph Δ_n, which can represent the set of all decision trees for sorting n elements. Nodes of this graph are separable subproblems of P_n. During each step we process one node and mark it with symbol *. We start with the graph that consists of one node P_n and finish when all nodes of the graph are processed.

Let the algorithm have already performed t steps. We now describe the step number $(t+1)$. If all nodes are processed then the algorithm terminates, and the resulted graph is Δ_n. Otherwise, choose a node (subproblem) Q that has not been processed yet.

If $|Q| = 1$, mark the considered node with symbol * and proceed to the step number $(t + 2)$. Let $|Q| > 1$. For each $x_i : x_j \in E(Q)$, draw a pair of edges from the node Q (this pair of edges will be called $(x_i : x_j)$-pair) and label these edges with the inequalities $x_i < x_j$ and $x_j < x_i$. These edges enter the nodes $Q(x_i < x_j)$ and $Q(x_j < x_i)$, respectively. If any of the nodes $Q(x_i < x_j)$, $Q(x_j < x_i)$ are not present in the graph then add these nodes to the graph. Mark the node Q with symbol * and proceed to the step number $(t + 2)$.

It is clear that Δ_n is a directed acyclic graph. A node of this graph is called terminal if it has no outgoing edges. A node Q is terminal if and only if $|Q| = 1$. We define the size of Δ_n to be the number of its nodes and edges. The complexity of a given problem mainly depends on the size of its corresponding DAG.

We now introduce the notion of proper subgraph of the graph Δ_n. For each node of the graph Δ_n, which is not terminal, we can remove any but not all pairs of edges that leave the node. The obtained subgraph will be called a proper subgraph of the graph Δ_n. It is clear that all terminal nodes of this subgraph are terminal nodes of the graph Δ_n. We consider Δ_n as a proper subgraph of the graph Δ_n. Proper subgraphs of the graph Δ_n can be obtained as results of procedures of decision tree optimization relative to the depth or average depth (external path length).

Let G be a proper subgraph of the graph Δ_n. Now, for each node Q of the graph G, we describe the set of decision trees corresponding to it. We will move from terminal nodes to the node P_n. Let Q be a terminal node and $Q = \{p\}$. Then the only trivial decision tree depicted in Figure 1 corresponds to the considered node.

Figure 1: Trivial decision tree

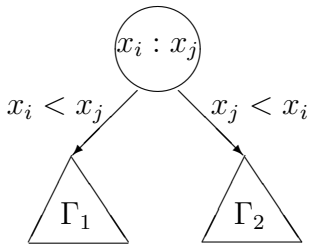

Figure 2: Aggregated decision tree

Let Q be a nonterminal node. Then there is a number of pairs of edges starting in Q. We consider an arbitrary pair and describe the set of decision trees corresponding to this pair. Let the considered pair be an $(x_i : x_j)$-pair where $x_i : x_j \in E(Q)$, and Γ_1 and Γ_2 be decision trees from the sets corresponding to the nodes $Q(x_i < x_j)$ and $Q(x_j < x_i)$, respectively. Then the decision tree depicted in Figure 2 belongs to the set of decision trees, which correspond to this pair. All such decision trees belong to the considered set, and this set does not contain any other decision trees. Then the set of decision trees corresponding to the node Q coincides with the union of sets of decision trees corresponding to pairs starting in Q. We denote by $D_G(Q)$ the set of decision trees corresponding to the node Q in the graph G.

Using the graph G it is easy to find the number of decision trees in the set $D_G(Q)$: $|D_G(Q)| = 1$ if Q is a terminal node. Let us consider a node Q, which is not terminal, and an $(x_i : x_j)$-pair of edges, which start in this node and enter the nodes $Q_1 = Q(x_i < x_j)$, $Q_2 = Q(x_j < x_i)$. We assign to this

pair the number $|D_G(Q_1)| \times |D_G(Q_2)|$. Then $|D_G(Q)|$ is equal to the sum of numbers corresponding to pairs starting in Q.

The next statement follows from the results obtained in [2].

Proposition 1. *Let Q be a node in the graph Δ_n (a separable subproblem of P_n). Then the set $D_{\Delta_n}(Q)$ coincides with the set of all decision trees for solving the subproblem Q. In particular, the set $D_{\Delta_n}(P_n)$ coincides with the set of all decision trees for sorting n elements.*

3.2. Procedures of Optimization

Let G be a proper subgraph of the graph Δ_n, and ψ be either depth h or external path length l (it is more convenient for us to work with l than with h_{avg}). Below we describe a procedure of decision tree optimization relative to ψ, which transforms the graph G into a proper subgraph G^ψ of G. We begin from terminal nodes and move to the node P_n. We attach a number $\psi(Q, G)$ to each node Q which is equal to the minimum value of ψ for a decision tree from $D_G(Q)$, and we may remove some pairs of edges, which start in the considered node. Let Q be a terminal node. We attach the number 0 to the node Q. Let us consider a node Q, which is not terminal, and an $(x_i : x_j)$-pair of edges, which start in this node. Then the edges are labeled with inequalities $x_i < x_j$, $x_j < x_i$, and enter nodes $Q(x_i < x_j)$, $Q(x_j < x_i)$, respectively. Let the numbers a_1, a_2 be attached already to these nodes. Then we attach to the considered pair of edges the number $1 + \max\{a_1, a_2\}$ if $\psi = h$, and the number $|Q| + a_1 + a_2$ if $\psi = l$.

Among numbers attached to pairs starting in Q we choose the minimum number a and attach it to the node Q as the number $\psi(Q, G)$. We remove all pairs starting in Q to which numbers are attached that are greater than a. When all nodes are treated we obtain a proper subgraph G^ψ of the graph G.

The next statement follows from the results obtained in [2].

Proposition 2. *Let $\psi \in \{h, l\}$, G be a proper subgraph of the graph Δ_n and Q be a node of G. Then $\psi(Q, G) = \min\{\psi(\Gamma) : \Gamma \in D_G(Q)\}$, $D_{G^\psi}(Q) = \{\Gamma : \Gamma \in D_G(Q), \psi(\Gamma) = \psi(Q, G)\}$ if $\psi = l$, and $D_{G^\psi}(Q) \subseteq \{\Gamma : \Gamma \in D_G(Q), \psi(\Gamma) = \psi(Q, G)\}$ if $\psi = h$.*

3.3. Tool Use

For $n = 2, \ldots, 8$, we construct the graph Δ_n and find the number of its nodes which is equal to $|\mathcal{P}(n)|$. Then we apply to the graph Δ_n the procedures of decision tree optimization relative to l and h separately. We obtain proper subgraphs Δ_n^l and Δ_n^h of the graph Δ_n in which (by Propositions 1 and 2) nodes P_n are labeled with numbers $l(P_n, \Delta_n) = l(P_n)$ and $h(P_n, \Delta_n) = h(P_n)$, respectively. As a result, we obtain also the value $h_{avg}(P_n) = l(P_n)/n!$. We count the cardinality of the set $D_{\Delta_n^l}(P_n)$ of decision trees corresponding to the node P_n in the graph Δ_n^l. By Propositions 1 and 2, $\left|D_{\Delta_n^l}(P_n)\right| = \left|Opt_{h_{avg}}(P_n)\right|$.

After that, we apply to the graph Δ_n^l the procedure of decision tree optimization relative to h. As a result, we obtain proper subgraph $\left(\Delta_n^l\right)^h$ of the graph Δ_n^l with the number $h(P_n, \Delta_n^l)$ attached to the node P_n. We find that $h(P_n, \Delta_n^l) = h(P_n)$ and $\left(\Delta_n^l\right)^h = \Delta_n^l$. Using Propositions 1 and 2, we obtain that all decision trees for sorting n elements with minimum average depth have the same depth equal to $h(P_n)$. Therefore $Opt_{h_{avg}}(P_n) \subseteq Opt_h(P_n)$.

For $n = 2, \ldots, 7$, all computations were done on a Mac Pro desktop with two 2.40GHz 6-core Intel Xeon processors and 64GB of RAM. For $n = 8$, we used Amazon cr1.8xlarge instance with two 2.60GHz 8-core Intel Xeon processors and 244GB of RAM (see http://aws.amazon.com/ec2/instance-types/ for details).

For $n = 2, \ldots, 7$, all computations were done directly as it was described above. For $n = 8$, we used a notion of equivalent subproblems and studied only nonequivalent separable subproblems defined by one inequality (the number of such subproblems is equal to one) and defined by two inequalities (the number of such subproblems is equal to three).

For a subproblem Q of the problem P_n, we denote by $T(Q)$ the set of decision trees for the subproblem Q solving. We say that two subproblems Q_1 and Q_2 of the problem P_n are equivalent if $|Q_1| = |Q_2|$ and there exists a bijection (a one-to-one correspondence) $\varphi : T(Q_1) \to T(Q_2)$ which preserves depth and external path length of decision trees.

First, we show that, for any $i, j \in \{1, \ldots, n\}$, $i \neq j$, the subproblem $P_n(x_i < x_j)$ is equivalent to the subproblem $P_n(x_1 < x_2)$. Evidently, $|P_n(x_1 < x_2)| = |P_n(x_i < x_j)|$. As a bijection

$$\varphi : T(P_n(x_1 < x_2)) \to T(P_n(x_i < x_j))$$

we can use a mapping which transforms a decision tree for the subproblem $P_n(x_1 < x_2)$ solving into a decision tree for the subproblem $P_n(x_i < x_j)$ solving by changing indexes in comparisons, inequalities and permutations attached to nonterminal nodes, edges and terminal nodes as follows: $1 \to i$, $2 \to j$, $i \to 1$, and $j \to 2$.

Next, we show in a similar way that, for any $i, j, k, l \in \{1, \ldots, n\}$, $i \neq j$, $k \neq l$, $\{i, j\} \neq \{k, l\}$, the subproblem $P_n(x_i < x_j)(x_k < x_l)$ is equivalent to one of the four subproblems: $S_1^n = P_n(x_1 < x_2)(x_3 < x_4)$, $S_2^n = P_n(x_1 < x_2)(x_2 < x_3)$, $S_3^n = P_n(x_1 < x_2)(x_1 < x_3)$, and $S_4^n = P_n(x_2 < x_1)(x_3 < x_1)$.

After that, we show that subproblems S_3^n and S_4^n are equivalent. It is clear that $S_3^n = \{(p_n, \ldots, p_1) : (p_1, \ldots, p_n) \in S_4^n\}$. As a bijection $\varphi : T(S_3^n) \to T(S_4^n)$ we can use a mapping which transforms a decision tree Γ for solving the subproblem S_3^n into a decision tree $\varphi(\Gamma)$ for solving the subproblem S_4^n in the following way. If an edge in Γ is labeled with an inequality $x_i < x_j$ then the corresponding edge in $\varphi(\Gamma)$ is labeled with the inequality $x_j < x_i$. If a terminal node in Γ is labeled with a permutation $(p_1, \ldots, p_n)$ then the corresponding terminal node in $\varphi(\Gamma)$ is labeled with the permutation $(p_n, \ldots, p_1)$.

We studied subproblems S_1^8, S_2^8, and S_3^8 in a way similar to the described above for P_n, and combined the obtained results. The numbers of separable subproblems for S_1^8, S_2^8, and S_3^8 are equal to 29668833, 12650470, and 50444492, respectively. The time of computation is 1848 seconds for S_1^8, 326 seconds for S_2^8, and 3516 seconds for S_3^8.

The study of separable subproblems of P_8 described by at most two equations allows us to understand the structure of the first two levels of decision trees solving P_8 and having minimum average depth. The root (the only node in the first level) can be labeled with an arbitrary comparison $x_i : x_j$ such that $i, j \in \{1, \ldots, n\}$, $i \neq j$. Any node in the second level can be labeled with an arbitrary comparison $x_k : x_l$ such that $k, l \in \{1, \ldots, n\}$, $k \neq l$ and $\{k, l\} \cap \{i, j\} = \emptyset$.

4. Conclusions

We find the minimum average depth 620160/8! of a decision tree for sorting 8 elements. We show that, for $n = 7, 8$, each decision tree for sorting n elements which has minimum average depth, has also minimum depth.

Acknowledgment

Research reported in this publication was supported by the King Abdullah University of Science and Technology (KAUST).

References

[1] A. Alkhalid, T. Amin, I. Chikalov, S. Hussain, M. Moshkov, B. Zielosko, Dagger: a tool for analysis and optimization of decision trees and rules, in: F.V.C. Ficarra (Ed.) Computational Informatics, Social Factors and New Information Technologies: Hypermedia Perspectives and Avant-Garde Experiences in the Era of Communicability Expansion, Blue Herons Editions, Bergamo, 2011, pp. 29–39.

[2] A. Alkhalid, I. Chikalov, S. Hussain, M. Moshkov, Extensions of dynamic programming as a new tool for decision tree optimization, in: S. Ramanna, R.J. Howlett, L.C. Jain (Eds.), Emerging Paradigms in Machine Learning, Smart Innovation, Systems and Technologies, vol. 13, Springer, Heidelberg, 2013, pp. 11–29.

[3] Y. Césary, Questionnaire, codage et tris, PhD Thesis, Institut Blaise Pascal, Centre National de la Recherche, Paris, 1968.

[4] I. Chikalov, Average Time Complexity of Decision Trees, Intelligent Systems Reference Library, vol. 21, Springer, Heidelberg, 2011.

[5] M.R. Garey, Optimal binary identification procedures, SIAM Journal on Applied Mathematics, 23 (1972) 173–186.

[6] D.E. Knuth, Sorting and Searching, second ed., The Art of Computer Programming, vol. 3, Addison–Wesley, Reading, MA, 1998.

[7] Ľ. Kollár, Optimal sorting of seven element sets, in: J. Gruska, B. Rovan, J. Wiedermann (Eds.), Proceedings Mathematical Foundations of Computer Science, Lecture Notes in Computer Science 233, Springer, Heidelberg, 1986, pp. 449–457.

[8] M. Moshkov, Time complexity of decision trees, in: J.F. Peters, A. Skowron (Eds.), Transactions on Rough Sets III, Lecture Notes in Computer Science 3400, Springer, Heidelberg, 2005, pp. 244–459.

[9] M. Moshkov, I. Chikalov, On algorithm for constructing of decision trees with minimal depth, Fundamenta Informaticae, 41 (2000) 295–299.

[10] M. Peczarski, The new results in minimum-comparison sorting, Algorithmica, 40(2) (2004) 133–145.