ON THE COMPLEXITY OF PARTIAL ORDER PRODUCTIONS

Andrew Chi-Chih Yao

CS-TR-136-88

February 1988

On The Complexity of Partial Order Productions

Andrew Chi-Chih Yao

Department of Computer Science
Princeton University
Princeton, New Jersey 08544

Abstract

Let P=($<_P$, Y) be a partial order on a set $Y = \{y_1, y_2, \ldots, y_n\}$ of n elements. The problem of *P-production* is, given an input of n distinct numbers $x_1, x_2, \ldots, x_n$, find a permutation σ of (1,2,...,n) such that $y_i <_P y_j$ implies $x_{\sigma(i)} < x_{\sigma(j)}$. Let $C(P)$, $\bar{C}(P)$ be, respectively, the minimum number and the minimum average number of binary comparisons $x_i : x_j$ needed by any decision-tree algorithm to produce P. We prove that $C(P) = \Theta(\bar{C}(P))$. As an intermediate result, we show that $C(P) = O(\log_2(n!/\mu(P)) + n)$, where $\mu(P)$ is the number of permutations consistent with P, proving a conjecture of Saks.

1 Introduction

Sorting and median-finding of a set of n numbers are two of the classical problems in combinatorial computation. It is well known (see Knuth [Kn, Section 5.3]) that sorting n numbers takes asymptotically $\Theta(n \log n)$ binary comparisons of the form $x_i : x_j$, both in the worst case and in the average case. For median-finding, it was first proved that the average-case complexity is $\Theta(n)$ (Floyd and Rivest [FR]), and later it was discovered that the worst-case complexity is also $\Theta(n)$ (Blum et al [BFPRT]). Thus, in both problems, the worst-case complexity and the average-case complexity are of the same order of magnitude. Are they special cases of a general class of problems for which this phenomenon is true? In this paper we will show that this is indeed so.

Let $P = (<_P, Y)$ be a partial order on a set $Y = \{y_1, y_2, \ldots, y_n\}$. The *P-production problem* is the following: Given n distinct numbers $x_1, x_2, \ldots, x_n$, find a permutation σ of (1,2,...,n) such that $y_i <_P y_j$ implies $x_{\sigma(i)} < x_{\sigma(j)}$. We are interested in the intrinsic complexity of this problem in the decision tree model. Clearly, sorting and median-finding are both special cases of the P-production problem.

A *decision tree* T is a binary tree, each of whose internal nodes u contains a comparison of the form $x_i : x_j$, and has two outgoing edges labeled by "<" and ">"; associated with each leaf ℓ is a permutation σ_ℓ of (1,2,...,n). Given any input $\tilde{x} = (x_1, x_2, \ldots, x_n)$ of distinct numbers, we travese a path $\xi(T, \tilde{x})$ in T from the root down, making comparisons and branching according to the outcomes, until a leaf $\ell_{\tilde{x}}$ is reached. We call T an *algorithm* for P-production if, for every $\tilde{x}$, $y_i <_P y_j$ implies $x_{\rho(i)} < x_{\rho(j)}$ where $\rho = \sigma_\ell$. Let *cost*(T, $\tilde{x}$) denote the number of comparisons made by T along the path $\xi(T, \tilde{x})$, and let *cost*(T) be $\max_{\tilde{x}}$ cost(T, $\tilde{x}$). Denote by $\mathcal{A}_P$ the family of all algorithms for P-productions. The *minimax complexity* $C(P)$ of P-production is defined as $\min\{ \text{cost}(T) \mid T \in \mathcal{A}_P\}$.

Let Γ_n be the set of all permutations of (1,2,...,n). A permutation ρ is said to be *consistent* with P, if $y_i <_P y_j$ implies $\rho(i) < \rho(j)$ for all i, j. Let $\Delta(P) \subseteq \Gamma_n$ be the set of all permutations consistent with P, and define $\mu(P) = |\Delta(P)|$.

The complexity problem of P-production was formulated and investigated by Schönhage [Sch], who showed by an information-theoretic argument that $C(P) \geq \log_2(n!/\mu(P))$. Further results on this problem were derived in Aigner [Ai]. It was conjectured in Saks [Sa] that Schönhage's lower bound can be achieved asymptotically, in the sense that $C(P) = O(\log_2(n!/\mu(P)) + n)$.

For any $T \in \mathcal{A}_P$, the *average cost* of T is defined as $\text{cost}'(T) = \frac{1}{n!} \sum_{\rho \in \Gamma_n} \text{cost}(T, \tilde{x}_\rho)$, where $\tilde{x}_\rho = (\rho(1), \rho(2), \ldots, \rho(n))$. The *minimean complexity* of P-production is defined as $\bar{C}(P) = \min\{\text{cost}'(T) \mid T \in \mathcal{A}_P\}$.

A partial order $P = (<_P, Y)$ is said to be *connectd*, if for every two distinct elements y and

y' in Y, there exists a sequence $y = y_1, y_2, \ldots, y_m = y'$ such that $y_i <_P y_{i+1}$ or $y_i >_P y_{i+1}$ for all i. Every partial order P can be uniquely decomposed into the disjoint union of connected partial orders $P_i = (<_{P_i}, Y_i)$, where the sets Y_i form a partition of Y. Let $\beta(P)$ denote the number of connected components in this decomposition.

In this paper we will prove the following results:

Theorem 1 For all P, $\bar{C}(P) = \Omega\Big(n - \beta(P) + \log_2\Big(\frac{n!}{\mu(P)}\Big)\Big)$.

Theorem 2 For all P, $C(P) = O\Big(n - \beta(P) + \log_2\Big(\frac{n!}{\mu(P)}\Big)\Big)$.

Theorem 3 For all P, $C(P) = \Theta(\bar{C}(P))$.

Theorem 2 proves the conjecture of Saks [Sa] mentioned earlier. Since $C(P) \geq \bar{C}(P)$ by definition, Theorem 3 is an immediate consequence of Theorems 1 and 2. The rest of this paper is devoted to a proof of Theorem 1 and Theorem 2.

2 Proof of Theorem 1

We will prove two lemmas. The first one is an extension of Schönhage's lower bound on the minimax complexity $C(P)$ to the minimean complexity.

Lemma 1 $\bar{C}(P) \geq \log_2\Big(\frac{n!}{\mu(P)}\Big)$.

Proof Let $T \in \mathcal{A}_P$. We will prove

$$\mathrm{cost}'(T) \geq \log_2\Big(\frac{n!}{\mu(P)}\Big). \tag{1}$$

For each leaf ℓ of T, let Q_ℓ be the partial order on X generated by the constraints $x_i > x_j$ along the path from the root to ℓ. As $T \in \mathcal{A}_P$, each Q_ℓ contains an isomorphic copy of P as a sub-partial order. This implies that $\mu(Q_\ell) \leq \mu(P)$. Let $q_\ell = \mu(Q_\ell)/n!$. Then

$$q_\ell \leq \frac{\mu(P)}{n!}. \tag{2}$$

If we consider a random input $\tilde{x}_\rho = (\rho(1), \rho(2), \ldots, \rho(n))$, where ρ is uniformly chosen from Γ_n, then q_ℓ is the probability that the traversed path $\xi(T, \tilde{x}_\rho)$ in T will end in the leaf ℓ. Let d_ℓ be the distance from the root to ℓ. Then,

$$\sum_\ell q_\ell = 1, \tag{3}$$

$$\mathrm{cost}'(T) = \sum_\ell q_\ell d_\ell. \tag{4}$$

It follows from (3) and (4) that $\text{cost}'(T)$ is the expected length of a uniquely decipherable code for an alphabet with symbol frequencies q_ℓ. It is a well-known fact (see e.g. [Ab, Section 4.1]) in Information Theory that a lower bound is given by the entropy, that is,

$$\text{cost}'(T) \geq \sum_{\ell} q_\ell \log_2 \frac{1}{q_\ell}.$$

Inequality (1) now follows from (2) and (3).□

Lemma 2 $\bar{C}(P) \geq n - \beta(P)$.

Proof Suppose the lemma is false. Then there exists $T \in \mathcal{A}_P$ and an input $\tilde{x} = (x_1, x_2, \ldots, x_n)$, such that $\text{cost}(T, \tilde{x}) < n - \beta(P)$. Let σ be the output permutation for $\tilde{x}$. We will derive a contradiction.

Let W be the sequence of inequalities $x_i < x_j$ generated along the path $\xi(T, \tilde{x})$; then $|W| < n - \beta(P)$. Denote by Q the partial order on X imposed by W. Then Q has more than $n - (n - \beta(P)) = \beta(P)$ connected components. Thus, there are integers r, s such that y_r, y_s are in the same component in P, while $x_{\sigma(r)}, x_{\sigma(s)}$ are in different components in Q.

Let $y_r = y_{i_1}, y_{i_2}, \ldots, y_{i_m} = y_s$ be such that, for each j, either $y_{i_j} <_P y_{i_{j+1}}$ or $y_{i_j} >_P y_{i_{j+1}}$. By the definition of $T \in \mathcal{A}_P$, every adjacent pairs in the sequence $x_{\sigma(r)} = x_{\sigma(i_1)}, x_{\sigma(i_2)}, \ldots, x_{\sigma(i_m)} = x_{\sigma(s)}$ must also be related in Q. This contradicts the assumption that $x_{\sigma(r)}$ and $x_{\sigma(s)}$ are in different components in Q.□

Theorem 1 follows immediately from Lemma 1 and Lemma 2.

3 Reduction

In this section, we show that to prove Theorem 2, it suffices to prove the following result:

Theorem 4 There exists a constant $\lambda > 0$ such that

$$C(P) \leq \lambda\Big(n - 1 + \log_2\big(\frac{n!}{\mu(P)}\big)\Big).$$

Assume that Theorem 4 is true. We will prove Theorem 2. Let $c = \beta(P)$. By definition, P can be written as the disjoint union of partial orders $P_i = (<_{P_i}, Y_i)$, $1 \leq i \leq c$, where Y_i's form a partition of Y. Let $|Y_i| = n_i$. Then $n_i > 0$ for all i, and

$$\sum_i n_i = n, \tag{5}$$

$$\mu(P) = \Big(\frac{n!}{n_1! n_2! \cdots n_c!}\Big)\mu(P_1)\mu(P_2)\cdots\mu(P_c). \tag{6}$$

We now describe an algorithm T for P-production. Let $N_0 = 0$. Define $N_i = \sum_{1 \le k \le i} n_i$, and $I_i = \{j \mid N_{i-1} < j \le N_i\}$ for $1 \le i \le c$. Given any input set $X = \{x_i \mid 1 \le i \le n\}$, consider for each $1 \le i \le c$, the set X_i as input to P_i-production, where $X_i = \{x_j \mid j \in I_i\}$. Apply Theorem 4 to each P_i, and let $\sigma_i : I_i \to I_i$ be the permutation found for P_i-production. Then define the output permutation $\sigma \in \Gamma_n$ by $\sigma(j) = \sigma_i(j)$ if $j \in I_i$. It is clear that $T \in \mathcal{A}_P$, since the output permutation σ satisfies the constraint that $y_j <_P y_k$ implies $x_{\sigma(j)} < x_{\sigma(k)}$.

Using (5) and (6), we obtain

$$\begin{aligned}
\text{cost}(T) &\le \lambda \sum_{1 \le i \le c} \Big(n_i - 1 + \log_2\Big(\frac{n_i!}{\mu(P_i)}\Big)\Big) \\
&= \lambda\Big(n - c + \log_2\Big(\frac{n_1! n_2! \cdots n_c!}{\mu(P_1)\mu(P_2)\cdots\mu(P_c)}\Big)\Big) \\
&= \lambda\Big(n - c + \log_2\Big(\frac{n!}{\mu(P)}\Big)\Big).
\end{aligned}$$

We have proved Theorem 2, assuming that Theorem 4 is true. In the next two sections we will prove Theorem 4.

4 An Algorithm

4.1 Preliminaries

Let $P = (<_P, Y)$ be a partial order. A subset $A \subseteq Y$ is an *independent set* of P, if $y \not<_P y'$ is true for all distinct $y, y' \in A$. The *width* of P, denoted by *width*(P), is the maximum size of any independent set of P. We associate with each P an independent set Y_P of maximum size. (Pick any one if there are several choices.)

Let $k \ge 2$ be any integer. A *k-partition of P* is a k-tuple $(A_1, A_2, \ldots, A_k)$, where the A_i's are disjoint subsets of Y whose union equals Y, such that $y <_P y'$, $y \in A_i$ and $y' \in A_j$ imply $i \le j$. We are interested in two special k-partitions. Let $\mathcal{B}_P = (B_{P,1}, B_{P,2}, B_{P,3})$, where $B_{P,1} = \{y \mid y <_P y' \text{ for some } y' \in Y_P\}$, $B_{P,2} = Y_P$, and $B_{P,3} = Y - Y_P - B_{P1}$. Clearly, $\mathcal{B}_P$ is a 3-partition. Note that some $B_{P,i}$ may be empty. To describe the second partition, let $\mathcal{M}_P$ be the set of all 2-partitions (A_1, A_2) of P such that $|A_1| = \lceil n/2 \rceil$ and $|A_2| = \lfloor n/2 \rfloor$. Let $\mathcal{D}_P = (D_{P,1}, D_{P,2})$ be a member of $\mathcal{M}_P$ such that $\mu(P_1)\mu(P_2)$ is maximum over all possible $(A_1, A_2) \in \mathcal{M}_P$, where P_i is the partial order P restricted to A_i.

Notations For the rest of the paper, $P = (<_P, Y)$ will denote a partial order on $Y = \{y_1, y_2, \ldots, y_n\}$, and X will denote the input set $\{x_1, x_2, \ldots, x_n\}$. For any $J \subseteq \{1, 2, \ldots, n\}$, we will use Y_J to denote the set $\{y_j \mid j \in J\}$, and P_J to denote the partial order induced by P on Y_J; we agree that $\mu(P_J) = 1$ when $J = \emptyset$. Similarly, for any $I \subseteq \{1, 2, \ldots, n\}$, we use X_I

to denote the set $\{x_i \mid i \in I\}$. For any two sets of numbers A, B, we write $A < B$ if $y < z$ for all $y \in A$ and $z \in B$. We adopt the convention that $0! = 1$, and we will employ two constants $c_2 = 40$ and $c_3 = 80$.

Lemma 3 Let $k \in \{2,3\}$. Let I be a nonempty subset of $\{1, 2, \ldots, n\}$, and $n_1, n_2, \ldots, n_k$ be nonnegative integers satisfying $\sum_{1 \le i \le k} n_i = |I|$. Then there is a decision tree T of height $c_k|I|$ such that, given any input of n distinct numbers $X = \{x_1, x_2, \ldots, x_n\}$, T determines disjoint $I_1, I_2, \ldots, I_k$ satisfying (a) $\cup_i I_i = I$, (b) $|I_i| = n_i$ for all i, and (c) $X_{I_1} < X_{I_2} < \cdots < X_{I_k}$.

Proof If $k = 2$, the decision tree first finds the $(n_1 + 1)$-st smallest element x_j in X_I, and then determines $I_1 = \{i \mid x_i < x_j\}$ and $I_2 = \{i \mid x_i \ge x_j\}$. This can be done in less than $40n$ comparisons using the selection algorithm in [BFPRT]. Similarly, if $k = 3$, we can find the $(n_1 + 1)$-st smallest and the $(n_1 + n_2 + 1)$-st smallest elements in X_I, by applying the selection algorithm in [BFPRT] twice, and then find I_1, I_2, I_3.□

4.2 Procedure POPROD

The algorithm can be described as a recursive procedure. Depending on the width of P, we will either use comparisons to divide X into three parts X_{I_i} satisfying $X_{I_1} < X_{I_2} < X_{I_3}$, or use comparisons to divide X into two parts X_{I_i} satisfying $X_{I_1} < X_{I_2}$. In the first case, we can match the elements $y_j \in B_{P,2}$ with elements in X_{I_2} in any fashion, and then recursively solve two subproblems: X_{I_1} as input to the production problem of P restricted to $B_{P,1}$, and X_{I_3} as input to the production problem of P restricted to $B_{P,3}$. This gives a valid final output, because $B_{P,2}$ is an independent set in P and $(B_{P,1}, B_{P,2}, B_{P,3})$ is a 3-partition. In the other case, we will simply solve recursively two subproblems: X_{I_1} as input to the production problem of P restricted to $D_{P,1}$, and X_{I_2} as input to the production problem of P restricted to $D_{P,2}$. Of course, the cardinality of the sets I_i need to be chosen to match those of the 3-partitions and 2-partitions.

The criterion for deciding which case to use is whether the width of P is greater than a fraction of n. Intuitively, the first case is more like the median-finding problem and the second case is more like the sorting problem. In the first case, we would like to get immediately a large indepenent subset of the elements y_j in Y assigned, while in the second case, we rely on the technique of divide-and-conquer, and try to divide the problems into two subproblems of nearly equal size.

As an example, consider the partial order P shown in Figure 1. The width of P is relatively large, and we have the first case. For this partial order, $B_{P,1} = \{y_1, y_2, y_3, y_4\}$, $B_{P,2} = \{y_5, y_8, y_{10}, y_{11}\}$, $B_{P,3} = \{y_6, y_7, y_9, y_{12}, y_{13}\}$. We thus use comparisons to divide X into three parts X_{I_i} satisfying $X_{I_1} < X_{I_2} < X_{I_3}$, where $|I_1| = 4$, $|I_2| = 4$, and $|I_3| = 5$. The elements in $B_{P,2}$ can be assigned in a 1-1 way to the x's in X_{I_2} without further comparisons. Of the two subproblems to be solved recursively, we will examine just the first one. To match the elements in $B_{P,1}$ with X_{I_1}, we observe that Q, the partial order P restricted to $B_{P,1}$, is $y_1 <_Q y_2 <_Q y_3 <_Q y_4$,

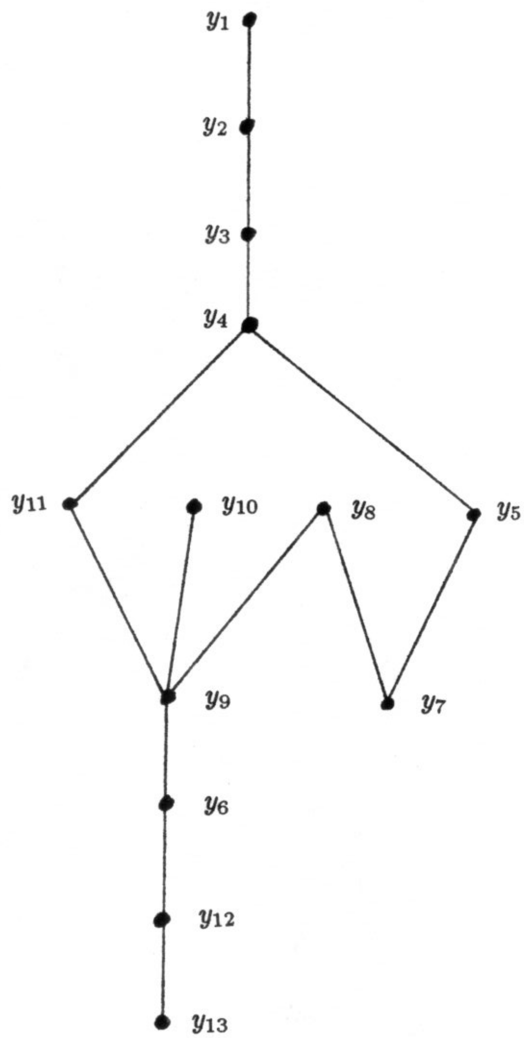

Figure 1 A partial order P ; smaller elements on top, e.g. $y_2 <_P y_4$.

which has width equal to 1. This means we have the second case for this subproblem. Clearly, $D_{Q,1} = \{y_1, y_2\}$ and $D_{Q,2} = \{y_3, y_4\}$. Therefore, we use comparisons to divide X_{I_1} into two parts A and A' with $A < A'$. Now we need to solve two subproblems: matching $D_{Q,1}$ to A, and $D_{Q,2}$ to A'.

We now specify the algorithm formally. Given an input set $X = \{x_1, x_2, \ldots, x_n\}$, the algorithm will output a permutation σ in the form of a set $\{(i, \sigma(i)) \mid 1 \le i \le n\}$; the correctness requirement is that $y_i <_P y_j$ implies $x_{\sigma(i)} < x_{\sigma(j)}$. We will give a recursive algorithm that takes as additional input arguments two sets $J \subseteq \{1, 2, \ldots, n\}$ and $I \subseteq \{1, 2, \ldots, n\}$ of equal size, and returns a *matching* between J and I, i.e. a set $V \subseteq J \times I$ such that each $j \in J$ appears exactly in one element $(j, k) \in V$, and each $i \in I$ appears exactly in one element $(m, i) \in V$. We will later prove that the matching produced satisfies the condition that, for $(j, m) \in V$ and $(j', m') \in V$, $y_j <_P y_{j'}$ implies $x_m < x_{m'}$. Thus, if we let $J = I = \{1, 2, \ldots, n\}$, we obtain the required permutation σ in the output.

Procedure POPROD(X, J, I);

CASE 1. $|J| \le 1$:

  if $J = I = \emptyset$, then return $\emptyset$;

  if $J = \{j\}$, $I = \{i\}$, then return $\{(j, i)\}$;

CASE 2. $(|J| > 1) \wedge (\text{width}(P_J) > \lceil |J|/100 \rceil)$:

  (a) Use $c_3|I|$ or less comparisons to divide I into disjoint I_1, I_2, I_3 such that $X_{I_1} < X_{I_2} < X_{I_3}$ and $|I_i| = |B_{P_J,i}|$ for $1 \le i \le 3$;
  [Comments: This can be done, by Lemma 3.]

  (b) Suppose $B_{P_J,i} = \{y_j \mid j \in J_i\}$, for $1 \le i \le 3$;
  let $V_2 \leftarrow \{(k_s, i_s) \mid 1 \le s \le |J_2|\}$, where k_s and i_s are the s-th smallest elements in J_2 and I_2.
  [Comments: No comparisons are used here.]

  (c) $V_1 \leftarrow \text{POPROD}(X, J_1, I_1)$;
  $V_3 \leftarrow \text{POPROD}(X, J_3, I_3)$;

  (d) return $V \leftarrow V_1 \cup V_2 \cup V_3$;

CASE 3. $(|J| > 1) \wedge (\text{width}(P_J) \le \lceil |J|/100 \rceil)$:

  (a) Use $c_2|I|$ or less comparisons to divide I into disjoint I_1, I_2 such that $X_{I_1} < X_{I_2}$, $|I_1| = \lceil n/2 \rceil$, and $|I_2| = \lfloor n/2 \rfloor$;
  [Comments: This can be done, by Lemma 3.]

  (b) Suppose $D_{P_J,i} = \{y_j \mid j \in J_i\}$, for $1 \le i \le 2$;

  (c) $V_1 \leftarrow \text{POPROD}(X, J_1, I_1)$;
  $V_2 \leftarrow \text{POPROD}(X, J_2, I_2)$;

  (d) return $V \leftarrow V_1 \cup V_2$;

4.3 Correctness

Lemma 4 In Procedure POPROD, the returned value V is a matching between J and I.

Proof We prove by induction on the size of J. The base case $|J| \leq 1$ is obvious. Inductively, suppose $|J| > 1$, and that the first recursive call results in CASE 2. Then V_2 constructed in step (b) is clearly a matching between J_2 and I_2. In step (c), by induction hypothesis, V_i is a matching between J_i and I_i for $i \in \{1,3\}$. Thus, V is a matching between J and I. A similar argument can be given when the first recursive call results in CASE 3.□

Lemma 5 Let V be the returned value in Procedure POPROD, when X, J, I are the input arguments. If $(k,k') \in V$, $(m,m') \in V$ and $y_k <_P y_m$, then $x_{k'} < x_{m'}$.

Proof We prove inductively on the size of the set J. The base case $|J| \leq 1$ is obvious. Inductively, suppose $|J| > 1$, and that the first recursive call results in CASE 2. Suppose that $(k,k') \in V_i$ and $(m,m') \in V_j$. As $y_k <_P y_m$, we have $i \leq j$ (since $(B_{P_J,1}, B_{P_J,2}, B_{P_J,3})$ is a 3-partition of P_J by definition). If $i < j$, then $x_{k'} < x_{m'}$, as $x_{k'} \in X_{I_i}, x_{m'} \in X_{I_j}$, and $X_{I_i} < X_{I_j}$. If $i = j \in \{1, 3\}$, then the lemma holds by the induction hypothsis. The case $i = j = 2$ does not arise, since no two distinct y_k and y_m in $B_{P_J,2}$ are comparable in P_J. A similar argument can be given when the first recursive call results in CASE 3.□

This proves that Procedure POPROD defines a decision tree algorithm for P-production, when we set $J = I = \{1, 2, \ldots, n\}$ in the input arguments. To complete the proof of Theorem 4, it remains to analyze the number of comparisons used in this procedure. This will be done in the next section.

5 Analysis of POPROD

Let $f_P(J)$ be the maximum number of comparisons used in POPROD(X, J, I) for any I and any relative ordering of the elements in X. Let $\lambda_2 = 5000c_2$, and $\lambda_3 = 100c_3$. In this section, we will prove

$$f_P(J) \leq \lambda_2 n + \lambda_3 \log_2\Big(\frac{|J|!}{\mu(P_J)}\Big). \tag{7}$$

This will complete the proof of Theorem 4.

5.1 Two Lemmas

We digress to prove two auxiliary lemmas before proving (7). We need a classical theorem due to Dilworth [D].

Dilworth's Theorem [D]. Let $P = (<_P, W)$ be any partial order, and width(P)=$m > 0$. Then W can be written as the disjoint union of m nonempty sets $W_i = \{w_{i,1}, w_{i,2}, \dots, w_{i,t_i}\}$, $1 \le i \le m$, such that $w_{i,1} <_P w_{i,2} <_P \cdots <_P w_{i,t_i}$ for all i.

Proof See [D].□

Let $P = (<_P, Y)$ be any partial order on a nonempty set $Y = \{y_1, y_2, \dots, y_n\}$. Let $\mathcal{B}_P = (B_{P,1}, B_{P,2}, B_{P,3})$ and $\mathcal{D}_P = (D_{P,1}, D_{P,2})$ be the two special k-partitions defined in Section 4.1. Suppose that $K, K' \subseteq \{1, 2, \dots, n\}$ are two nonempty sets of equal cardinality. Let $W = \{w_j \mid j \in K\}$, and $Q = (<_Q, W)$ be a partial order on W. A matching V between K and K' is said to be *consistent with* Q, if $((k,k') \in V) \wedge ((m,m') \in V) \wedge (w_k <_Q w_m)$ implies $k' < m'$. Let $\Delta(Q, K, K')$ be the set of matchings between K and K' that are consistent with Q. Clearly,

$$|\Delta(Q, K, K')| = \mu(Q). \tag{8}$$

Lemma 6 Let J_1, J_2, J_3 be such that $B_{P,i} = \{y_j \mid j \in J_i\}$ for $i \in \{1, 2, 3\}$. Then

$$\frac{\mu(P_{J_1})}{|J_1|!} \frac{\mu(P_{J_3})}{|J_3|!} \ge \frac{\mu(P)}{|Y|!}.$$

Proof We first discuss the case when both J_1 and J_3 are nonempty. Let $\sigma \in \Delta(P)$. For each $i \in \{1, 2, 3\}$, let $I_i = \{\sigma(j) \mid j \in J_i\}$, and define a mapping $\sigma_i : J_i \to I_i$ by $\sigma_i(j) = \sigma(j)$ for all $j \in J_i$. It is easy to verify that $\sigma_i \in \Delta(P_{J_i}, J_i, I_i)$.

In this way, each $\sigma \in \Delta(P)$ is associated with a 6-tuple $(I_1, I_2, I_3, \sigma_1, \sigma_2, \sigma_3)$, where I_1, I_2, I_3 form a partition of $\{1, 2, \dots, n\}$ with $|I_i| = |J_i|$, and $\sigma_i \in \Delta(P_{J_i}, J_i, I_i)$. Furthermore, as can be easily verified, σ is uniquely determined by the 6-tuple. Counting the number of such 6-tuples, we obtain

$$\begin{aligned} \mu(P) &\le \frac{|Y|!}{|J_1|!\,|J_2|!\,|J_3|!} \mu(P_{J_1})\, \mu(P_{J_2})\, \mu(P_{J_3}) \\ &\le |Y|! \frac{\mu(P_{J_1})}{|J_1|!} \frac{\mu(P_{J_3})}{|J_3|!}. \end{aligned}$$

This proves the lemma for the case when both J_1 and J_3 are nonempty.

The lemma is obviously true when both J_1 and J_3 are empty. For the case when exactly one J_i is empty, $i \in \{1, 3\}$, we can easily modify the above proof to prove the lemma, by omitting all the references to the quantities σ_i, I_i. This completes the proof of Lemma 6.□

Lemma 7 Let J_1, J_2 be such that $D_{P,i} = \{y_j \mid j \in J_i\}$ for $i \in \{1, 2\}$. If $\beta(P) \le \lceil n/100 \rceil$, then

$$\frac{\mu(P_{J_1})}{|J_1|!} \frac{\mu(P_{J_2})}{|J_2|!} \ge (1.01)^{|Y|} \frac{\mu(P)}{|Y|!}.$$

Proof Let $m = \beta(P) \geq 1$. Then $m \leq \lceil n/100 \rceil$ and thus $n \geq 100$. Clearly, both J_i are nonempty. By Dilworth's Theorem, P can be covered by m chains of lengths, say, $\ell_1, \ell_2, \ldots, \ell_m > 0$. Thus, each 2-partition $(A_1, A_2) \in \mathcal{M}_P$ can be specified by integers $k_1, k_2, \ldots, k_m$, where k_i is the number of elements of A_1 on the i-th chain. After standard manipulations for optimizing expressions, we obtain

$$\begin{aligned} |\mathcal{M}_P| &\leq \prod_{1 \leq i \leq m} (1 + \ell_i) \\ &\leq (\frac{2n}{m})^m \\ &\leq (200)^{\lceil n/100 \rceil}. \end{aligned} \tag{9}$$

Let $I_1 = \{j \mid 1 \leq j \leq \lceil n/2 \rceil\}$ and $I_2 = \{j \mid 1 \leq j \leq \lfloor n/2 \rfloor\}$. For each $\sigma \in \Delta(P)$, let $J_i' = \{j \mid \sigma(j) \in I_i\}$ and $V_i = \{(j, \sigma(j)) \mid j \in J_i'\}$ for $i \in \{1, 2\}$. Then, $(Y_{J_1'}, Y_{J_2'}) \in \mathcal{M}_P$. Also, for each i, V_i is a matching between J_i' and I_i, and is consistent with $P_{J_i'}$. In this way, each $\sigma \in \Delta(P)$ is associated with a quadruple (J_1', J_2', V_1, V_2), where $(Y_{J_1'}, Y_{J_2'}) \in \mathcal{M}_P$, and $V_i \in \Delta(P_{J_i'}, J_i', I_i)$ for $i \in \{1, 2\}$. Furthermore, it is easy to see that σ is uniquely determined by the quadruple. From (8), we obtain

$$\begin{aligned} \mu(P) &\leq \sum_{(Y_{J_1'}, Y_{J_2'}) \in \mathcal{M}_P} \mu(P_{J_1'}) \mu(P_{J_2'}) \\ &\leq |\mathcal{M}_P| \, \mu(P_{J_1}) \mu(P_{J_2}). \end{aligned} \tag{10}$$

From (9) and (10) and the fact $n \geq 100$, we obtain

$$\begin{aligned} \mu(P) &\leq (200)^{\lceil n/100 \rceil} \mu(P_{J_1}) \mu(P_{J_2}) \\ &\leq \frac{1}{(1.01)^n} \binom{n}{\lceil n/2 \rceil} \mu(P_{J_1}) \mu(P_{J_2}). \end{aligned}$$

Rearranging terms in the above expression gives the inequality to be proved in the lemma.□

5.2 The Analysis

Given P, J, where $J \neq \emptyset$, we construct a *cost tree* $V_{P,J}$. Each node v will be associated with a triplet $\eta(v) = (\delta(v), \alpha(v), S(v))$, where $\delta(v) \in \{0, 2, 3\}$, $\alpha(v)$ is a nonnegative integer, and $S(v) \subseteq Y$. We will say that v is of *type* $\delta(v)$ and *weight* $\alpha(v)$.

The cost trees are recursively constructed. If $|J| = 1$ with, say, $J = \{j\}$, then $V_{P,J}$ consists of a single node v with $\eta(v) = (0, 0, \{y_j\})$.

If $\Big(|J| > 1\Big) \wedge \Big(\text{width}(P_J) > \lceil |J|/100 \rceil\Big)$, then the root v of $V_{P,J}$ has $\eta(v) = (3, c_3|J|, B_{P_{J,2}})$, and for each nonempty $B_{P_{J,i}}$, $i \in \{1, 3\}$, there is a son v_i of the root such that the subtree rooted at v_i is V_{P,J_i}, where J_i is the set of j with $y_j \in B_{P_{J,i}}$.

If $\Big(|J|>1\Big) \wedge \Big(\text{ width}(P_J) \leq \lceil |J|/100 \rceil\Big)$, then the root v of $V_{P,J}$ has $\eta(v) = (2, c_2|J|, \emptyset)$, and for each $i \in \{1,2\}$, there is a son v_i of the root such that the subtree rooted at v_i is V_{P,J_i}, where J_i is the set of j with $y_j \in D_{P_{J,i}}$. (Note that in this case both J_i are nonempty.)

We have defined the cost tree $V_{P,J}$. An example of a cost tree is shown in Figure 2; square nodes are of type 0, oval nodes with c_2 besides them are of type 2, and those with c_3 besides them are of type 3.

We now relate $f_P(J)$ to the cost tree. Let $a_i(P,J)$ be the total weight of type-i nodes in $V_{P,J}$. That is, let $a_i(P,J) = \sum_{v,b(v)=i} \alpha(v)$ for $i \in \{2,3\}$.

Lemma 8 $f_P(J) \leq a_2(P,J) + a_3(P,J)$.

Proof We prove by induction on the size of $|J|$. If $|J| = 1$, then $f_P(J) = a_2(P,J) = a_3(P,J) = 0$, and the lemma holds. Inductively, assume that $|J| = m > 1$, and that we have proved the lemma for all J with size less than m. If $\text{width}(P_J) > \lceil |J|/100 \rceil$, then in the execution of Procedure POPROD, for any I and X, CASE 2 occurs at the top level, and thus

$$f_P(J) \leq c_3|J| + \sum_{i \in \{1,3\}, J_i \neq \emptyset} f_P(J_i),$$

where J_i are the sets of j such that $y_j \in B_{P_{J,i}}$. Applying the induction hypothesis, we have

$$\begin{aligned} f_P(J) &\leq c_3|J| + \sum_{i \in \{1,3\}, J_i \neq \emptyset} (a_2(P,J_i) + a_3(P,J_i)) \\ &= a_2(P,J) + a_3(P,J). \end{aligned}$$

A similar argument works when $\text{width}(P_J) \leq \lceil |J|/100 \rceil$. This completes the inductive step of the proof.□

We now analyze $a_i(P,J)$.

Lemma 9 $a_3(P,J) \leq 100c_3|J|$.

Proof We first state two facts which can be easily verified inductively. For all type-3 internal nodes v,

$$|S(v)| > \lceil \alpha(v)/(100c_3) \rceil, \tag{11}$$

For any two distinct internal nodes v and v',

$$S(v) \cap S(v') = \emptyset \tag{12}$$

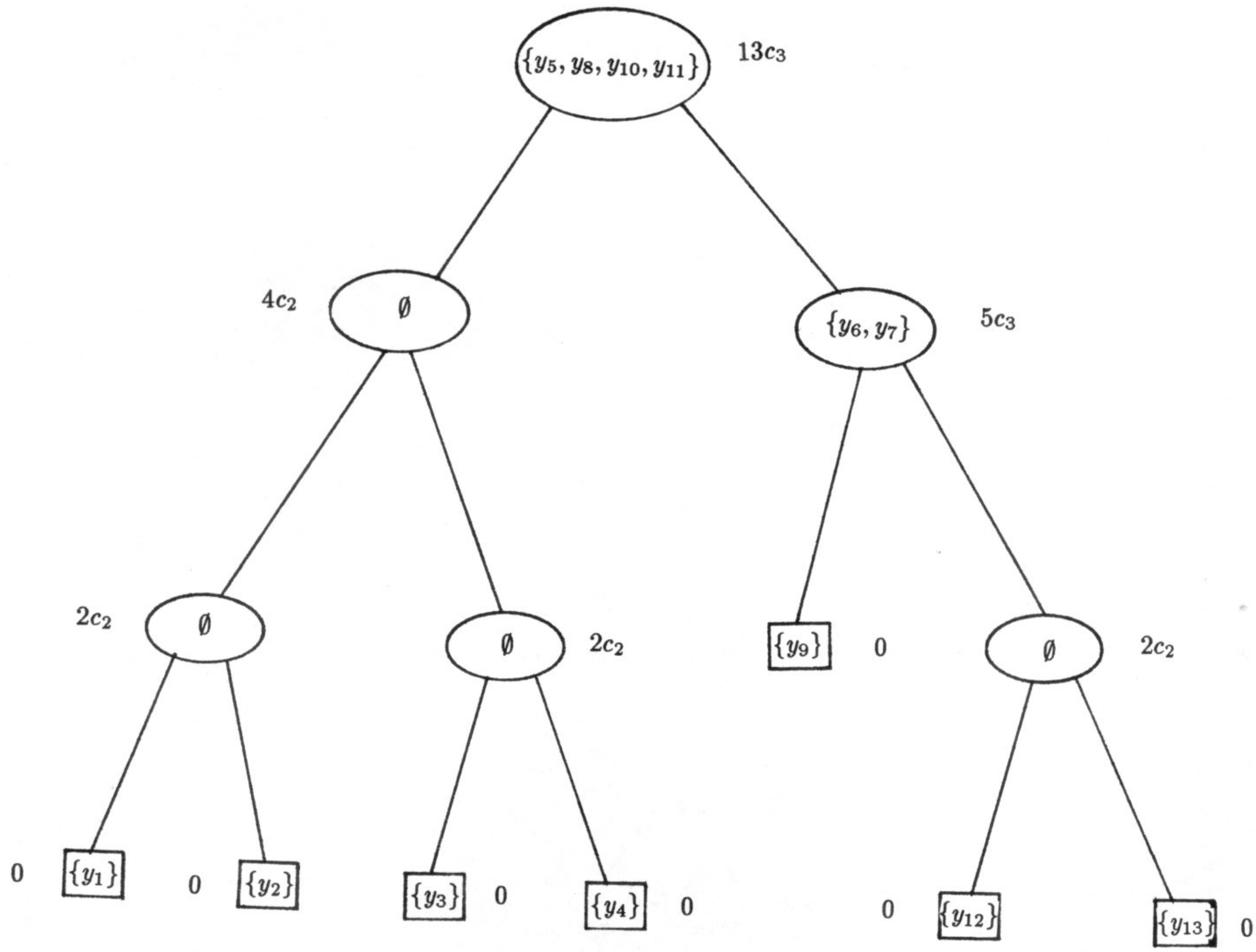

Figure 2 $V_{P,J}$ with the P in Figure 1 and $J = \{1, 2, \ldots, 13\}$; $S(v)$ are shown inside the nodes v, and $\alpha(v)$ are shown just outside v.

It follows from (11) and (12) that

$$\begin{aligned} a_3(P,J) &= \sum_{v,b(v)=3} \alpha(v) \\ &\leq 100c_3 \sum_{v,b(v)=3} |S(v)| \\ &\leq 100c_3|Y_J| \\ &= 100c_3|J|. \end{aligned}$$

This proves the lemma.□

Lemma 10 $a_2(P,J) \leq 5000c_2 \log_2\Big(|J|!/\mu(P_J)\Big)$.

Proof We prove the lemma inductively on the size of J. If $|J| = 1$, then $a_2(P,J) = 0$, $|J| = \mu(P_J) = 1$, and the lemma is valid. Inductively, suppose that $|J| = m > 1$ and that we have proved the lemma for all J, P with $|J| < m$.

If $\text{width}(P_J) > \lceil |J|/100 \rceil$, then in the execution of Procedure POPROD, for any I and X, CASE 2 occurs at the top level. Let J_i is the set of j with $y_j \in B_{P_{J,i}}$. Applying the induction hypothesis to each son v_i, and keeping in mind that we employ the convention that $0! = \mu(P_\emptyset) = 1$, we obtain

$$\begin{aligned} a_2(P,J) &= \sum_{i\in\{1,3\}, J_i\neq\emptyset} a_2(P,J_i) \\ &\leq 5000c_2 \sum_{i\in\{1,3\}} \log_2\Big(\frac{|J_i|!}{\mu(P_{J_i})}\Big) \\ &= 5000c_2 \log_2\Big(\frac{|J_1|!|J_3|!}{\mu(P_{J_1})\mu(P_{J_3})}\Big). \end{aligned}$$

Applying Lemma 6 to the partial order P_J, we then have

$$a_2(P,J) \leq 5000c_2 \log_2\Big(\frac{|J|!}{\mu(P_J)}\Big).$$

If $\text{width}(P_J) \leq \lceil |J|/100 \rceil$, CASE 3 occurs in the execution of POPROD. Let J_i be the set of j with $y_j \in D_{P_{J,i}}$. Both J_i are nonempty in this case. We have

$$\begin{aligned} a_2(P,J) &\leq c_2|J| + 5000c_2 \log_2\Big(\frac{|J_1|!}{\mu(P_{J_1})}\Big) + 5000c_2 \log_2\Big(\frac{|J_2|!}{\mu(P_{J_2})}\Big) \\ &= c_2 m + 5000c_2 \log_2\Big(\frac{|J_1|!|J_2|!}{\mu(P_{J_1})\mu(P_{J_2})}\Big). \end{aligned}$$

Applying Lemma 7 to P_J, we obtain

$$\begin{aligned} a_2(P,J) &\leq c_2 m + 5000c_2 \log_2\Big(\frac{m!}{\mu(P_J)}\frac{1}{(1.01)^m}\Big). \\ &\leq 5000c_2 \log_2\Big(\frac{m!}{\mu(P_J)}\Big). \end{aligned}$$

This completes the inductive step of the proof.□

Inequality (7), and hence Theorem 4, follows immediately from the preceding three lemmas. This comlpletes the proof of Theorem 2.

6 Remarks

In this paper we have determined up to a constant factor the complexity of a class of problems involving partial orders, in terms of a familiar combinatorial quantity $\mu(P)$. It is of interest to explore the complexity of other classes of problems involving partial orders. An excellent survey of this topic can be found in Saks [Sa]. We list below some open problems directly related to our present discussion.

(a) Can we charaterize the complexity of producing σ that satisfies more general constraints than a single partial order P? For example, let $P_1, P_2, \ldots, P_m$ be partial orders on $Y = \{y_1, y_2, \ldots, y_n\}$. What is the complexity of producing, for any input $X = \{x_1, x_2, \ldots, x_n\}$, a σ such that for some i, $y_j <_{P_i} y_k$ implies $x_{\sigma(j)} < x_{\sigma(k)}$ for all j, k?

(b) The results in this paper imply that the randomized decision tree complexity for P-production is asymptotically of the same order of magnitude as the worst-case complexity. Is this true for more general class of production problems, such as the one mentioned in (a)? It is also of interest to study the question of whether randomization helps for other types of partial order problems.

(c) The present paper gives an existence proof of a near-optimal height decision tree for P-production. If the partial order itself is also given as an input, is there a polynomial time algorithm (counting all the book-keeping steps) that uses a near-optimal number of comparisons for producing a partial order?

References

[Ab] N. Abramson, *Information Theory and Coding*, McGraw Hill, 1963.

[Ai] M. Aigner, "Producing posets," *Discrete Mathematics*, **35** (1981), pp. 1-15.

[BFPRT] M. Blum, R. Floyd, V. Pratt, R. Rivest, and R. Tarjan, "Time bounds for selection," *Journal of Computer and System Sciences*, **7** (1973), pp. 448-461.

[D] R. P. Dilworth, "A decomposition theorem for partially ordered set," *Annals of Mathematics*, **2** (1950), pp. 161-166.

[FR] R. W. Floyd and R. L. Rivest, "Expected time bounds for selection," *Communications of ACM*, **18** (1975), pp. 165-172.

[Kn] D. E. Knuth, *The Art of Computer Programming, Vol 3: Searching and Sorting*, Addison-Wesley, 1973.

[Sa] M. E. Saks, "The information theoretic bound for problems on ordered sets and graphs" in *Graphs and Order*, edited by I. Rival, D. Reidel Publishing Company, 1985, pp. 137-168.

[Sch] A. Schönhage, "The production of partial orders," *Astérisque* **38-39** (1976), pp. 229-246.