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 \( P=\langle P, Y \rangle \) be a partial order on a set \( Y = \{y_1, y_2, \ldots, y_n\} \) of \( n \) elements. The problem of \emph{P-production} is, given an input of \( n \) distinct numbers \( x_1, x_2, \ldots, x_n \), find a permutation \( \sigma \) of \( (1, 2, \ldots, n) \) 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 \( C(P) = O(\bar{C}(P)) \). 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 [BFPR]). 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 $\sigma$ of $(1, 2, \ldots, n)$ 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 $\ell$ is a permutation $\sigma_\ell$ of $(1, 2, \ldots, n)$. Given any input $\bar{x} = (x_1, x_2, \ldots, x_n)$ of distinct numbers, we traverse a path $\xi(T, \bar{x})$ in $T$ from the root down, making comparisons and branching according to the outcomes, until a leaf $\ell_{\bar{x}}$ is reached. We call $T$ an algorithm for $P$-production if, for every $\bar{x}$, $y_i <_P y_j$ implies $x_{\rho(i)} < x_{\rho(j)}$ where $\rho = \sigma_\ell$. Let $\text{cost}(T, \bar{x})$ denote the number of comparisons made by $T$ along the path $\xi(T, \bar{x})$, and let $\text{cost}(T)$ be $\max_{\bar{x}} \text{cost}(T, \bar{x})$. Denote by $A_P$ the family of all algorithms for $P$-productions. The minimax complexity $C(P)$ of $P$-production is defined as

$$
\min \{ \text{cost}(T) \mid T \in A_P \}.
$$

Let $\Gamma_n$ be the set of all permutations of $(1, 2, \ldots, n)$. A permutation $\rho$ 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 $T \in A_P$, the average cost of $T$ is defined as $\text{cost}'(T) = \frac{1}{n!} \sum_{\rho \in \Gamma_n} \text{cost}(T, \bar{x}_{\rho})$, where $\bar{x}_{\rho} = (\rho(1), \rho(2), \ldots, \rho(n))$. The minimean complexity of $P$-production is defined as $\bar{C}(P) = \min \{\text{cost}'(T) \mid T \in A_P\}$.

A partial order $P = (<P, Y)$ is said to be connectd, if for every two distinct elements $y$ and
In this paper we will prove the following results:

**Theorem 1** For all \( P, \tilde{C}(P) = \Omega\left(n - \beta(P) + \log_2\left(\frac{n!}{\mu(P)}\right)\right).\)

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

**Theorem 3** For all \( P, C(P) = \Theta(\tilde{C}(P)).\)

Theorem 2 proves the conjecture of Saks [Sa] mentioned earlier. Since \( C(P) \geq \tilde{C}(P) \) 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** \( \tilde{C}(P) \geq \log_2\left(\frac{n!}{\mu(P)}\right). \)

**Proof** Let \( T \in \mathcal{A}_P \). We will prove

\[
\text{cost}'(T) \geq \log_2\left(\frac{n!}{\mu(P)}\right). \tag{1}
\]

For each leaf \( \ell \) of \( T \), let \( Q_\ell \) be the partial order on \( X \) generated by the constraints \( x_i > x_j \) along the path from the root to \( \ell \). As \( T \in \mathcal{A}_P \), each \( Q_\ell \) 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 \( \rho \) is uniformly chosen from \( \Gamma_n \), then \( q_\ell \) is the probability that the traversed path \( \xi(T, \tilde{x}_\rho) \) in \( T \) will end in the leaf \( \ell \). Let \( d_\ell \) be the distance from the root to \( \ell \). Then,

\[
\sum_\ell q_\ell = 1, \tag{3}
\]

\[
\text{cost}'(T) = \sum_\ell q_\ell d_\ell. \tag{4}
\]
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 \( q_e \). 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,

\[
\text{cost}'(T) \geq \sum_e q_e \log_2 \frac{1}{q_e}.
\]

Inequality (1) now follows from (2) and (3). \( \Box \)

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

**Proof** Suppose the lemma is false. Then there exists \( T \in A_P \) and an input \( \tilde{x} = (x_1, x_2, \ldots, x_n) \), such that \( \text{cost}(T, \tilde{x}) < n - \beta(P) \). Let \( \sigma \) 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 \( T \in A_P \), 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 \( Q \). \( \Box \)

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

\[
C(P) \leq \lambda \left( n - 1 + \log_2 \left( \frac{n!}{\mu(P)} \right) \right).
\]

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

\[
\sum_i n_i = n, \quad \mu(P) = \left( \frac{n!}{n_1!n_2! \cdots n_c!} \right) \mu(P_1) \mu(P_2) \cdots \mu(P_c).
\]


We now describe an algorithm $T$ for $P$-production. Let $N_0 = 0$. Define $N_i = \sum_{1 \leq k \leq i} n_i$, and $I_i = \{j \mid N_{i-1} < j \leq N_i\}$ for $1 \leq i \leq c$. Given any input set $X = \{x_i \mid 1 \leq i \leq n\}$, consider for each $1 \leq i \leq c$, 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 $\sigma_i : I_i \rightarrow I_i$ 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 $T \in A_P$, since the output permutation $\sigma$ satisfies the constraint that $y_j <_P y_k$ implies $x_{\sigma(j)} < x_{\sigma(k)}$.

Using (5) and (6), we obtain

$$\text{cost}(T) \leq \lambda \sum_{1 \leq i \leq c} \left( n_i - 1 + \log_2 \left( \frac{n_i!}{\mu(P_i)} \right) \right) = \lambda \left( n - c + \log_2 \left( \frac{n_1! n_2! \cdots n_c!}{\mu(P_1) \mu(P_2) \cdots \mu(P_c)} \right) \right) = \lambda \left( n - c + \log_2 \left( \frac{n!}{\mu(P)} \right) \right).$$

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 $P = \langle P, Y \rangle$ be a partial order. A subset $A \subseteq Y$ is an independent set of $P$, if $y \nless_{P} y'$ is true for all distinct $y, y' \in A$. The width of $P$, denoted by $\text{width}(P)$, 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 $k \geq 2$ 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 $y \nless_{P} y'$, $y \in A_i$ and $y' \in A_j$ imply $i \neq j$. We are interested in two special $k$-partitions. Let $B_P = \{B_{P,1}, B_{P,2}, B_{P,3}\}$, where $B_{P,1} = \{y \mid y \nless_{P} y' \text{ for some } y' \in Y_P\}$, $B_{P,2} = Y_P$, and $B_{P,3} = Y - Y_P - B_{P,1}$. Clearly, $B_P$ is a 3-partition. Note that some $B_{P,i}$ may be empty. To describe the second partition, let $M_P$ be the set of all 2-partitions $(A_1, A_2)$ of $P$ such that $|A_1| = \lfloor n/2 \rfloor$ and $|A_2| = \lceil n/2 \rceil$. Let $D_P = \{D_{P,1}, D_{P,2}\}$ be a member of $M_P$ such that $\mu(P_1) \mu(P_2)$ is maximum over all possible $(A_1, A_2) \in M_P$, where $P_i$ is the partial order $P$ restricted to $A_i$.

Notations For the rest of the paper, $P = \langle P, Y \rangle$ 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 \( \sum_{1 \leq i \leq k} n_i = |I| \). 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 \( I_2 = \{i \mid x_i \geq x_j\} \). 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 \( X_{I_1} \) satisfying \( X_{I_1} < X_{I_2} < X_{I_3} \), or use comparisons to divide \( X \) into two parts \( X_{I_1} \) 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 \( B_{P,3} \) 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 independent 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 \( X_{I_1} \) 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. $y_2 < y_4$. 
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 \( X_l \) 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 \( \sigma \) in the form of a set \( \{(i, \sigma(i)) \mid 1 \leq i \leq n\} \); the correctness requirement is that \( x_i <_P y_j \) 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 \( \sigma \) in the output.

**Procedure POPROD** \((X, J, I)\);

**CASE 1.** \(|J| \leq 1\):
- if \( J = I = \emptyset \), then return \( \emptyset \);
- if \( J = \{j\}, I = \{i\} \), then return \( \{(j, i)\} \);

**CASE 2.** \((|J| > 1) \land \left( \text{width}(P_J) > |J|/100\right)):

(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 \( 1 \leq i \leq 3 \);
[Comments: This can be done, by Lemma 3.]

(b) Suppose \( B_{P_J,i} = \{y_j \mid j \in I_i\} \), for \( 1 \leq i \leq 3 \);
- let \( V_2 \leftarrow \{(k_s, i_s) \mid 1 \leq s \leq |J_2|\} \), 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) \( V_1 \leftarrow \text{POPROD}(X, J_1, I_1) \);
(d) \( V_3 \leftarrow \text{POPROD}(X, J_3, I_3) \);

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

**CASE 3.** \((|J| > 1) \land \left( \text{width}(P_J) \leq |J|/100\right)):

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

(b) Suppose \( D_{P_J,i} = \{y_j \mid j \in I_i\} \), for \( 1 \leq i \leq 2 \);
(c) \( V_1 \leftarrow \text{POPROD}(X, J_1, I_1) \);
(d) \( V_2 \leftarrow \text{POPROD}(X, J_2, I_2) \);
(d) return \( V \leftarrow V_1 \cup V_2 \);

7
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 $x_{k'} \in X_{i}, x_{m'} \in X_{j}$, and $X_{i} < X_{j}$. If $i = j \in \{1, 3\}$, then the lemma holds by the induction hypothesis. 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

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

(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].

8
Dilworth’s Theorem [D]. Let $P = (\prec, W)$ be any partial order, and $\text{width}(P) = m > 0$. Then $W$ can be written as the disjoint union of $m$ nonempty sets $W_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,t_i}\}$, $1 \leq i \leq m$, such that $w_{i,1} \prec w_{i,2} \prec \ldots \prec w_{i,t_i}$ for all $i$.

**Proof** See [D].

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

$$|\Delta(Q, K, K')| = \mu(Q). \quad (8)$$

**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

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

**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 $\sigma_i : J_i \rightarrow I_i$ 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 $\{1, 2, \ldots, n\}$ with $|I_1| = |J_1|$, and $\sigma_i \in \Delta(P_{J_i}, J_i, I_i)$. Furthermore, as can be easily verified, $\sigma$ is uniquely determined by the 6-tuple. Counting the number of such 6-tuples, we obtain

$$\mu(P) \leq \frac{|Y|!}{|J_1|! |J_2|! |J_3|!} \frac{\mu(P_{J_1}) \mu(P_{J_2}) \mu(P_{J_3})}{\mu(P) |J_1|! |J_2|! |J_3|!}.$$

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 $\sigma_i, I_i$. 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 $\beta(P) \leq \lceil n/100 \rceil$, then

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

\[
|\mathcal{M}_P| \leq \prod_{1 \leq i \leq m} (1 + \ell_i) \\
\leq \left( \frac{2n}{m} \right)^m \\
\leq (200)^{\lceil n/100 \rceil}. \tag{9}
\]

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 \( \sigma \) is uniquely determined by the quadruple. From (8), we obtain

\[
\mu(P) \leq \sum_{(Y_{J'_1}, Y_{J'_2}) \in \mathcal{M}_P} \mu(P_{J'_1}) \mu(P_{J'_2}) \\
\leq |\mathcal{M}_P| \mu(P_{J'_1}) \mu(P_{J'_2}). \tag{10}
\]

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

\[
\mu(P) \leq (200)^{\lceil n/100 \rceil} \mu(P_{J'_1}) \mu(P_{J'_2}) \\
\leq \frac{1}{(1.01)^n} \left( \frac{n}{\lceil n/2 \rceil} \right) \mu(P_{J'_1}) \mu(P_{J'_2}).
\]

Rearranging terms in the above expression gives the inequality to be proved in the lemma.\( \square \)

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 \(|J| > 1\) \(\land\) \(\left( \text{width}(P_J) > \lceil |J|/100 \rceil \right) \), then the root \( v \) of \( V_{P,J} \) has \( \eta(v) = (3, c_3 \cdot |J|, B_{P,J_3}) \), 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 \(|J| > 1\) and \(\text{width}(P_J) \leq \lceil |J|/100 \rceil\), 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 \(a_i(P, J) = \sum_{v, \eta(v) = i} \alpha(v)\) 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 \(\text{width}(P_J) \leq \lceil |J|/100 \rceil\), then in the execution of Procedure POPROD, for any \(I\) and \(X\), CASE 2 occurs at the top level, and thus

\[
f_P(J) \leq c_3|J| + \sum_{i \in \{1, 2\}, J_i \neq \emptyset} f_P(J_i),
\]

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

\[
f_P(J) \leq c_3|J| + \sum_{i \in \{1, 2\}, J_i \neq \emptyset} (a_2(P, J_i) + a_3(P, J_i)) = a_2(P, J) + a_3(P, J).
\]

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

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 \(v\),

\[
|S(v)| > \lceil \alpha(v)/(100c_3) \rceil,
\]  \hspace{1cm} (11)

For any two distinct internal nodes \(v\) and \(v'\),

\[
S(v) \cap S(v') = \emptyset
\]  \hspace{1cm} (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
\[
a_3(P, J) = \sum_{v, h(v) = 3} \alpha(v) \\
\leq 100c_3 \sum_{v, h(v) = 3} |S(v)| \\
\leq 100c_3 |Y_J| \\
= 100c_3 |J|.
\]

This proves the lemma. □

**Lemma 10** \( a_2(P, J) \leq 5000c_2 \log_2 \left( \frac{|J|!}{\mu(P_J)} \right) \).

**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 \( \text{width}(P_J) \leq \lfloor |J|/10 \rfloor \), 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
\[
a_2(P, J) = \sum_{i \in \{1, 3\}, J_i \neq \emptyset} a_2(P, J_i) \\
\leq 5000c_2 \sum_{i \in \{1, 3\}} \log_2 \left( \frac{|J_i|!}{\mu(P_{J_i})} \right) \\
= 5000c_2 \log_2 \left( \frac{|J_1|!|J_3|!}{\mu(P_{J_1})\mu(P_{J_3})} \right).
\]

Applying Lemma 6 to the partial order \( P_J \), we then have
\[
a_2(P, J) \leq 5000c_2 \log_2 \left( \frac{|J|!}{\mu(P_J)} \right).
\]

If \( \text{width}(P_J) \leq \lfloor |J|/100 \rfloor \), 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
\[
a_2(P, J) \leq c_2 |J| + 5000c_2 \log_2 \left( \frac{|J_1|!}{\mu(P_{J_1})} \right) + 5000c_2 \log_2 \left( \frac{|J_3|!}{\mu(P_{J_1})\mu(P_{J_3})} \right) \\
= c_2 m + 5000c_2 \log_2 \left( \frac{|J_1|!|J_2|!}{\mu(P_{J_1})\mu(P_{J_3})} \right).
\]

Applying Lemma 7 to \( P_J \), we obtain
\[
a_2(P, J) \leq c_2 m + 5000c_2 \log_2 \left( \frac{m!}{\mu(P_J)(1.01)^m} \right) \\
\leq 5000c_2 \log_2 \left( \frac{m!}{\mu(P_J)} \right).
\]
This completes the inductive step of the proof. □

Inequality (7), and hence Theorem 4, follows immediately from the preceding three lemmas. This completes 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 μ(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 characterize the complexity of producing σ that satisfies more general constraints than a single partial order P? For example, let P1, P2, …, Pm be partial orders on Y = {y1, y2, …, yn}. What is the complexity of producing, for any input X = {x1, x2, …, xn}, a σ such that for some i, y_j <_{P_i} y_k implies x_{σ(j)} < x_{σ(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

