

Discrete Mathematics 35 (1981) 1-15
North-Holland Publishing Company

PRODUCING POSETS

Martin AIGNER

II. Mathematisches Institut, Freie Universität Berlin, Königin-Luise-Str. 24-26, D-1000 Berlin 33

Received 26 September 1980

Revised 3 March 1981

Many of the well-known selection and sorting problems can be understood as the production of certain partial orders, using binary comparisons. The paper discusses the complexity of the production of arbitrary posets all of a given size n (on a totally ordered ground-set). Further investigation is undertaken on an important class of posets, called partitions, which includes all selection problems treated in the literature.

1. Introduction

Many of the well-known selection and sorting problems can be understood as the production of certain partial orders using binary comparisons. For example, the posets in Fig. 1 correspond to the problems: Select the first 3 elements in

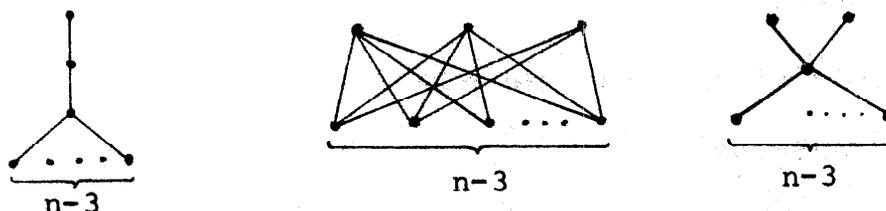


Fig. 1.

order, without order and select the 3rd element, respectively, out of a chain on n elements. Schönhage investigated in [10] the cost of the production of an arbitrary poset (on a given totally ordered reservoir) concentrating on the influence of the size of the reservoir and on the effect of "mass production".

In this paper, we are interested in the cost of arbitrary posets all of a given size n (on a totally ordered reservoir). After giving the basic definitions in Section 2 we derive some general bounds for the cost functions in Section 3. From Section 4 on we concentrate on an important class of posets, called partitions, which includes all the selection problems treated in the literature so far. Sections 5 and 6 contain a sample of recursions and bounds for the cost of partitions. After presenting an example in Section 7 we close with some remarks on the parallel case in Section 8 and on some open problems in Section 9.

2. Producing posets

Let n be a fixed natural number and let $\mathcal{O}(n)$ be the set of all (non-isomorphic) posets on n elements. We order $\mathcal{O}(n)$ by

$$P \leq Q: \Leftrightarrow \exists \text{ monotone injection from } P \text{ to } Q.$$

With this order relation $\mathcal{O}(n)$ itself becomes a poset with the antichain A_n on n elements as unique minimal element and the chain C_n on n elements as unique maximal element.

On the base-set (usually called the reservoir) we are given a fixed total order, unknown to us. Let $P \in \mathcal{O}(n)$. Our goal is to determine P with certainty by a sequence of comparisons between pairs of elements. Any such algorithm T corresponds in the usual way to a rooted binary tree with A_n as the root and with the nodes of the tree corresponding to the posets that have been determined up to then. The condition that T should determine P with certainty shall mean that P can be embedded monotonically into all the posets corresponding to the end-nodes of the tree. Hence we give the following definition, suggested by Schönhage [10]:

Definition. Let $P \in \mathcal{O}(n)$. An algorithm T produces P iff for all end-posets Q_i of T we have $P \leq Q_i$. The length $l(T)$ of T is the height of the corresponding tree. The average length $\bar{l}(T)$ is defined analogously assuming that all order relations are equally probable.

Definition. Let $P \in \mathcal{O}(n)$. The (serial) cost $C(P)$ of P is

$$C(P) = \min_T l(T)$$

where the minimum is extended over all algorithms T which produce P . Similarly, the (serial) average cost $\bar{C}(P)$ of P is

$$\bar{C}(P) = \min_T \bar{l}(T)$$

with the minimum again extended over all algorithms T which produce P .

We may also consider algorithms where at every stage (often called rounds) we may perform as many disjoint comparisons as possible, i.e., at most $\lfloor \frac{1}{2}n \rfloor$. Here, we count the number $l_p(T)$ of rounds needed in T to reach any end-node.

Definition. Let $P \in \mathcal{O}(n)$. The parallel cost $C_p(P)$ of P is

$$C_p(P) = \min_T l_p(T),$$

and similarly, the *parallel average cost* $\bar{C}_p(P)$ of P is

$$\bar{C}_p(P) = \min_T \bar{l}_p(T)$$

where T runs through all parallel algorithms producing P .

With these definitions we may formulate the

Main problems. Determine the cost functions $C(P)$, $\bar{C}(P)$, $C_p(P)$ and $\bar{C}_p(P)$ for $P \in \mathcal{O}(n)$.

Example 1. Let M and N be the smallest non-distributive lattices as in Fig. 2.

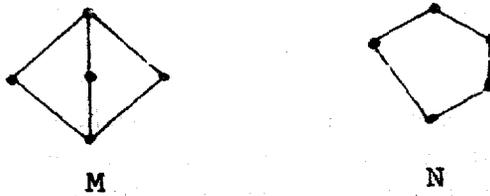


Fig. 2.

Here the results are:

$$C(M) = 6, \quad \bar{C}(M) = 5\frac{2}{3}, \quad C_p(M) = \bar{C}_p(M) = 3;$$

$$C(N) = 6, \quad \bar{C}(N) = 5\frac{2}{3}, \quad C_p(N) = 4, \quad \bar{C}_p(N) = 3\frac{1}{4}.$$

The lower bounds $C(M) \geq 6$, $C(N) \geq 6$ follow from 3.5(ii) below. As an example, let us verify $C(N) \leq 6$, $\bar{C}(N) \leq 5\frac{2}{3}$. After performing two disjoint comparisons and then comparing the maximal elements we arrive at the poset of Fig. 3.

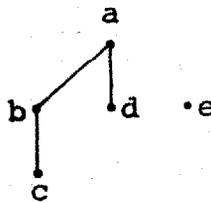


Fig. 3.

Next we compare d and e . If $d > e$ then by comparing c and e , we produce in either case N (after altogether 5 comparisons). If, on the other hand, $d < e$, then we make the comparisons $a : e$ and $c : d$. Anyone of the resulting 4 posets is $\geq N$ in $\mathcal{O}(5)$ whence we have produced N after altogether 6 comparisons. Thus $C(N) \leq 6$. Since in one third of the time we needed 5 comparisons and two thirds of the time we needed 6 comparisons, we conclude $\bar{C}(N) \leq 5\frac{2}{3}$.

Up until Section 8 we will solely concentrate on the serial case.

3. Bounds for the cost functions

There are a few simple bounds on the numbers $C(P)$ etc. which can be derived by looking at the poset $\mathcal{O}(n)$. So let us first make a few observations on $\mathcal{O}(n)$. The following statement is clear.

(3.1) $\mathcal{O}(n)$ is a *graded* poset where the rank of P equals the number of relations $a < b$ existing in P , i.e., $r(P) = |\{(a, b) : a < b\}|$. Hence, in particular, $r(\mathcal{O}(n)) = \binom{n}{2}$.

An interesting and unsolved question concerns the cardinality of $\mathcal{O}(n)$ (see Birkhoff [4, Problem 2]). The smallest values are $|\mathcal{O}(2)| = 2$, $|\mathcal{O}(3)| = 5$, $|\mathcal{O}(4)| = 16$, $|\mathcal{O}(5)| = 63$. No nontrivial bounds are known for the growth of $|\mathcal{O}(n)|$ for $n \rightarrow \infty$.

Let us agree on the following notation and terminology. P^* is the dual of P . $P = P_1 + \dots + P_k$ means that P is the direct sum of the P_i 's, $\text{comp}(P)$ is the number of these summands, called *components*. If a and b are incomparable in P then we write $a \parallel b$. A pair a, b , where b covers a in P , denoted by $a < \cdot b$, is called an *edge* of P . $h(P)$ is the number of edges in P . For instance, in Fig. 2 $h(M) = 6$ and $h(N) = 5$. Let a and b be two incomparable elements in P . $P + (a < b)$ denotes the poset which is the transitive closure of the relations in P together with the new relation $a < b$. Of course, we always have $P < P + (a < b)$ in $\mathcal{O}(n)$.

With this notation we obviously have

(3.2) $P < \cdot Q$ implies $Q = P + (a < b)$ where $a < \cdot b$ in Q , $a \parallel b$ in P .

Furthermore, we have:

(3.3) Suppose $P < \cdot Q$ with $Q = P + (a < b)$ then $Q \leq P + (b < a)$.

Proof. The mapping $\varphi: Q \rightarrow P + (b < a)$ with $\varphi(x) = x$ for all $x \neq a, b$ and $\varphi(a) = b$, $\varphi(b) = a$ is a monotone injection. \square

3.4 Proposition. Let $P, Q \in \mathcal{O}(n)$.

- (i) $P \leq Q$ in $\mathcal{O}(n) \Rightarrow C(P) \leq C(Q)$, $\bar{C}(P) \leq \bar{C}(Q)$,
- (ii) $P < \cdot Q$ in $\mathcal{O}(n) \Rightarrow C(Q) \leq C(P) + 1$, $\bar{C}(Q) \leq \bar{C}(P) + 1$,
- (iii) $n\text{-comp}(P) \leq \bar{C}(P) \leq C(P) \leq r(P)$,
- (iv) $C(P) = C(P^*)$, $\bar{C}(P) = \bar{C}(P^*)$.

Proof. (i) is implied by the definition of the order in $\mathcal{O}(n)$. (ii) follows easily from (3.3). The inequality $C(P) \leq r(P)$ is now a consequence of (ii) while $n\text{-comp}(P) \leq \bar{C}(P)$ follows from the observation that any comparison reduces the number of components by at most 1. (iv) is obvious by reversing all order relations. \square

Suppose $P = P_1 + \dots + P_k$, then clearly $C(P) \leq C(P_1) + \dots + C(P_k)$. The inequality

in this formula may, however, be strict. The smallest known example for this surprising fact is $P = Q + C_1$ where



Here $C(P) = 7 < C(Q) = 8$ (Paterson).

3.4(iii) provides the simplest lower and upper bounds for the cost of a poset P . The following proposition lists three more lower bounds. For $P \in \mathcal{O}(n)$, let $e(P)$ be the number of monotone embeddings of P into the chain C_n , and let $h(P)$ be, as before, the number of edges of P . Furthermore, denote by $\max P$ and $\min P$ the set of maximal and minimal elements of P , respectively. Observe that $a \parallel b$ in P implies

$$e(P) = e(P + (a < b)) + e(P + (b < a))$$

whence in particular $P \leq Q$ implies $e(P) \geq e(Q)$. Notice, however, that $e(P) \geq e(Q)$ does not imply $C(P) \leq C(Q)$. As an example we have for P and Q in Fig. 4, $e(P) = 9$, $e(Q) = 12$ but $C(P) = 4$, $C(Q) = 5$.

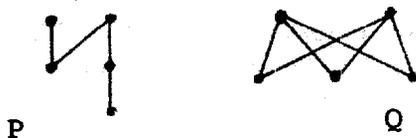


Fig. 4.

3.5 Proposition. Let $P \in \mathcal{O}(n)$ and let T be any optimal P -producing algorithm with Q_1, \dots, Q_t as end-posets. Then

- (i) $\sum_{i=1}^t e(Q_i) = n!$ whence¹ $C(P) \geq \lceil \log n! / e(P) \rceil$,
- (ii) $C(P) \geq \lceil \frac{3}{2}n \rceil - |\max P| - |\min P|$,
- (iii) $C(P) \geq \max_{i=1, \dots, t} h(Q_i)$.

Proof. $\sum_{i=1}^t e(Q_i) = n!$ is implied iteratively from

$$e(P) = e(P + (a < b)) + e(P + (b < a)).$$

Since $e(Q_i) \leq e(P)$ for all i and $t \leq 2^{C(P)}$ we infer

$$n! \leq te(P) \leq 2^{C(P)} e(P).$$

To prove (ii) we define an oracle (adversary rule, see Knuth [9, p. 200]) on T by stipulating

$$a > b \text{ whenever } a \in \max Q, b \notin \max Q \text{ or } b \in \min Q, a \notin \min Q,$$

making arbitrary decisions in all other cases. This rule reduces the total number of

¹ log always means \log_2 .

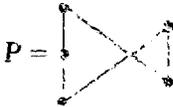
maximal and minimal elements by at most 1 at each stage unless two isolated elements are being compared in which case this number is reduced by 2. The number of comparisons of this last kind is at most $\lfloor \frac{1}{2}n \rfloor$. Suppose, by applying this rule, we finally reach the poset Q after, say, s comparisons. Then $|\max Q| \leq |\max P|$ and $|\min Q| \leq |\min P|$ because $P \leq Q$ in $\mathcal{O}(n)$, and $s \leq C(P)$. Since the total number of maximal and minimal elements is $2n$ at the start (every element being both maximal and minimal) we obtain

$$\begin{aligned} |\max P| + |\min P| &\geq |\max Q| + |\min Q| \geq 2n - s - \lfloor \frac{1}{2}n \rfloor \\ &= \lfloor \frac{1}{2}n \rfloor - s \geq \lfloor \frac{3}{2}n \rfloor - C(P). \end{aligned}$$

(iii) is proved by simply observing that the comparison corresponding to an edge must be performed by the algorithm since it cannot be induced transitively. \square

3.5 Corollary. Suppose $P \in \mathcal{O}(n)$ has a unique maximal and a unique minimal element. Then $C(P) \geq \lfloor \frac{3}{2}n \rfloor - 2$.

Example 2. The lattices M and N of Example 1 satisfy $C(M) \geq 6$, $C(N) \geq 6$ by 3.5(ii). Since $e(M) = 6$ we observe that bound 3.5(i) only yields $C(M) \geq \lceil \log 5!/6 \rceil = 5$. The bounds in 3.5 can often be combined to obtain sharper results. Consider the poset



Since $e(P) = 8$, 3.5(i) yields $C(P) = \lceil \log 5!/8 \rceil = 4$, as does 3.5(ii). Suppose $C(P) = 4$. Since $h(P) = 5$, an optimal algorithm must have as end-posets only posets Q with $Q > P$ by 3.5(iii). But for any such poset Q we have $e(Q) \leq 6$ whence by 3.5(i) $C(P) \geq \lceil \log 5!/6 \rceil = 5$. On the other hand, we can easily produce P with 5 comparisons, thus proving $C(P) = 5$.

4. Partitions

For arbitrary posets P there is little hope at present to obtain results which go significantly beyond Propositions 3.4 and 3.5 (except for very small n). So let us specialize in the sequel to the following class of posets.

Definition. Let $n = k_1 + \dots + k_d$ be an ordered partition of n into d positive integers k_i . To this partition there corresponds a unique poset in $\mathcal{O}(n)$ consisting of d groups G_1, \dots, G_d with $|G_i| = k_i$, $i = 1, \dots, d$, such that $a > b$ iff $a \in G_i$, $b \in G_j$ with $i < j$. This poset is called the *partition of type* (k_1, \dots, k_d) and we denote its complexity by $C(n; k_1, \dots, k_d)$. Sometimes we shorten the notation to

$k = (n; k_1, \dots, k_d)$ and $C(k)$. We call $n = \sum_{i=1}^d k_i$ the *cardinality* of the partition k and d its *order*.

Three types have received special attention in the theory of sorting (see [9]):

(A) The partition $(n; \underbrace{1, \dots, 1}_t, n-t)$ -sorting the first t elements in order.

(B) The partition $(n; t-1, 1, n-t)$ -selecting the t th element.

(C) The partition $(n; t, n-t)$ -sorting the first t -elements without regard to order.

The cost functions for these types are commonly denoted by

$$W_t(n) := C(n; 1, \dots, 1, n-t),$$

$$V_t(n) := C(n; t-1, 1, n-t),$$

$$U_t(n) := C(n; t, n-t).$$

Propositions 3.4(iii), (iv) and 3.5(i), (ii) now read:

$$C(n; k_1, \dots, k_d) = C(n; k_d, \dots, k_1), \quad (4.1)$$

$$n-1 \leq C(n; k_1, \dots, k_d) \leq \sum_{1 \leq i < j \leq d} k_i k_j, \quad (4.2)$$

$$C(n; k_1, \dots, k_d) \geq \left\lceil \log \frac{n!}{k_1! \cdots k_d!} \right\rceil, \quad (4.3)$$

$$C(n; k_1, \dots, k_d) \geq \lceil \frac{n}{2} \rceil - k_1 - k_d. \quad (4.4)$$

Let $\mathcal{P}(n)$ be the set of all partitions in $\mathcal{O}(n)$. It is a well-known fact in combinatorics (see [1, p. 200]) that $|\mathcal{P}(n)| = 2^{n-1}$ and that $\mathcal{P}(n)$ as a sub-poset of $\mathcal{O}(n)$ has, in fact, the following structure:

4.5 Proposition. We have $\mathcal{P}(n) \cong \mathcal{B}(n-1)$ where $\mathcal{B}(n-1)$ is the lattice of subsets of an $(n-1)$ -set. The rank of a partition of type $(n; k_1, \dots, k_d)$ in the lattice $\mathcal{P}(n)$ is $d-1$.

Example 3. In Fig. 5 the lattices $\mathcal{P}(n)$, $n = 2, 3, 4, 5$, are listed where, for brevity, the partitions are denoted $k_1 k_2 \cdots k_d$. The circled numbers next to the lattice elements are the costs of the corresponding partitions.

We notice that there is a natural ordering of each level, namely the lexicographic one, by which the partitions of a given rank are arranged symmetrically around the middle. The leftmost chain $(n) < (1, n-1) < (1, 1, n-2) < \cdots < (1, 1, \dots, 1)$ corresponds to problem (A), the rank 1-level to problem (C), and problem (B) to that of the rank $n/2$ -level.

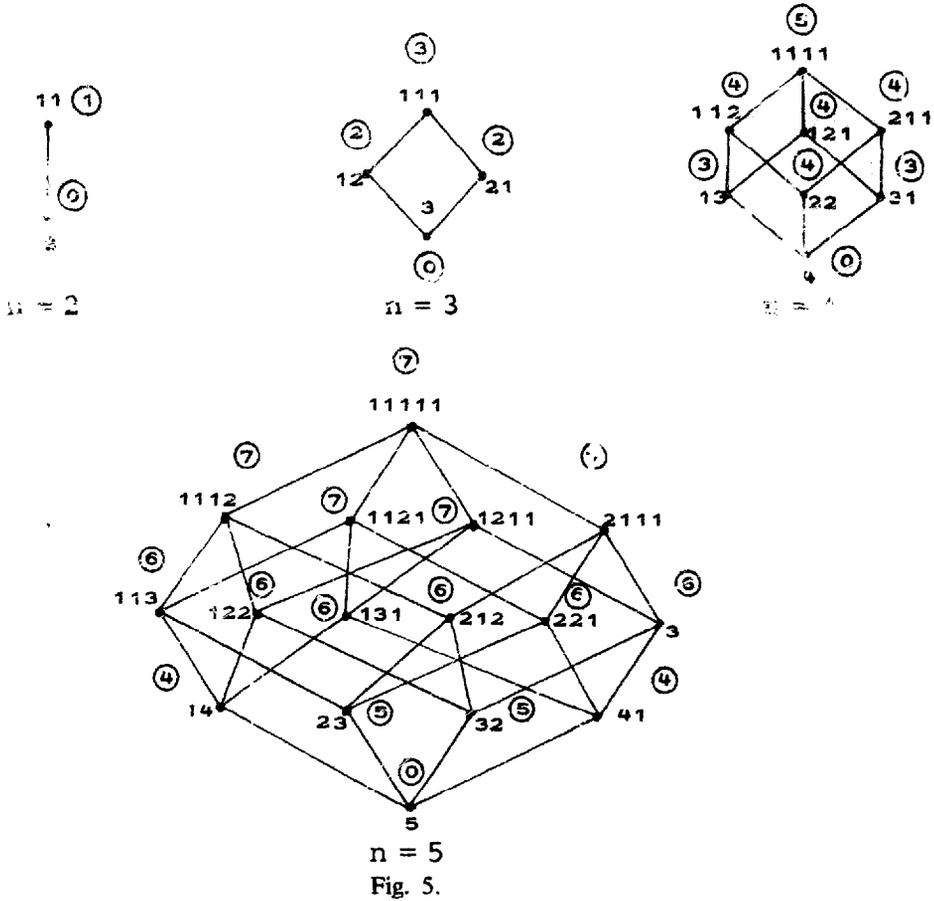


Fig. 5.

By looking at the lattice $\mathcal{P}(n)$ we can immediately deduce a few further facts:

$$\begin{aligned} \text{If } k < \cdot l, \text{ i.e., } k_1 = l_1, \dots, k_{i-1} = l_{i-1}, k_i = l_i + l_{i+1}, \dots, k_d \\ = l_{d+1} \text{ then } C(n; l) \leq C(n; k) + U_i(k_i). \end{aligned} \tag{4.6}$$

$$\begin{aligned} \text{Let } l = (l_1, \dots, l_e). \text{ Then } C(n; l) \leq \min_{i=1, \dots, e-1} (C(n; k_i) + U_i(l_i + l_{i+1})) \\ \text{where } k_i = (l_1, \dots, l_i + l_{i+1}, \dots, l_e) \text{ for } i = 1, \dots, e-1. \end{aligned} \tag{4.7}$$

4.8 Proposition. $C(k \vee l) \leq C(k) + C(l)$.

Proof. By $C(k)$ comparisons we obtain a poset which is structured according to k . Now consider an optimal algorithm for l . Every time elements are compared which have already been ordered according to k the algorithm goes in the appropriate direction. After at most $C(l)$ comparisons the partition $k \vee l$ is then produced. \square

5. Recursions

Formula (4.6) of the last section is a recursive inequality in the lattice $\mathcal{P}(n)$ for fixed n . Quite often, however, one is interested in a recursion of a fixed partition

type and varying n , e.g., $W_i(n)$, $V_i(n)$ or $U_i(n)$ for fixed i . This section contains a sample of recursions of this nature. For brevity, we will employ freely the usual language of tournaments, players, rounds, matches and oracles.

$$C(n; k_1, \dots, k_d) < \begin{cases} C(n+1; k_1, \dots, k_d+1) \\ C(n+1; k_1+1, \dots, k_d). \end{cases} \quad (5.1)$$

Proof. Consider an optimal tournament for $(n+1; k_1, \dots, k_d+1)$ and let the oracle declare the loser of the first game to be the worst player. The matches played after the first constitute then a tournament for $(n; k_1, \dots, k_d)$. The second inequality is established in analogous fashion. \square

$$C(n+1; k_1, \dots, k_d+1) \leq C(n; k_1, \dots, k_d) + \left(\sum_{i=1}^{d-1} k_i - 1 \right) \quad \text{if } \sum_{i=1}^{d-1} k_i \geq 3, \quad (5.2)$$

$$C(n+1; k_1+1, \dots, k_d) \leq C(n; k_1, \dots, k_d) + \left(\sum_{i=2}^d k_i - 1 \right) \quad \text{if } \sum_{i=2}^d k_i \geq 3.$$

Proof. Out of the $n+1$ players take any $\sum_{i=1}^{d-1} k_i + 1$ and determine the worst, say s , who surely belongs to the block containing the worst $k_d + 1$ players. This takes $\sum_{i=1}^{d-1} k_i$ games. Discarding s we set up a tournament for $(n; k_1, \dots, k_d)$ where we can use one previous match as our starting match, thereby saving one game. \square

Note. If $\sum_{i=1}^{d-1} k_i \leq 2$ then we either have $(n+1; 1, n)$ or $(n+1; 1, 1, n-1)$ or $(n+1; 2, n-1)$. In all these cases the cost is known.

$$C\left(n+1; \sum_{i=1}^{s-1} k_i, k_s+1, \sum_{i=s+1}^d k_i\right) \leq C(n; k_1, \dots, k_d) + k_{s-1} + k_{s+1} \quad \text{for } 1 \leq s \leq d. \quad (5.3)$$

Proof. Use $C(n; k_1, \dots, k_d)$ comparisons to obtain the partition $(n; k_1, \dots, k_d)$. Let x be the new element, and determine the smallest element y of the k_{s-1} -block and x . If $y \neq x$ then we are done. Otherwise determine the largest element of the k_{s+1} -block and x . The last two steps take at most $k_{s-1} + k_{s+1}$ comparisons. \square

6. Upper bounds

The recursions listed in the last section allow us in principle to derive upper bounds since we know the values of $W_1(n)$, $W_2(n)$ and $U_2(n)$ (see below). These bounds are, of course, exceedingly crude except for very small n . Whereas no general method applicable to any given partition $(n; k_1, \dots, k_d)$ is in sight as yet we can make a few observations about some cases of special interest.

(a) *Order 2.* By employing the technique of Hadian-Sobel [5] the following

bound can be obtained:

$$U_t(n) \leq (n-t) + (t-1) \lceil \log(n-t+1) \rceil \tag{6.1}$$

where we may, of course, replace t by $n-t$ if this yields a better bound. (6.1) is precise for $t=1$ and 2 ; for the complete result of $U_3(n)$ see [2].

(b) *Order 3.*

$$C(n; 1, 1, n-2) = (n-2) + \lceil \log n \rceil. \tag{6.2} \quad [8]$$

$$C(n; 1, n-2, 1) = \lceil \frac{3}{2}n \rceil - 2. \tag{6.3} \quad (\text{Pohl [9, p. 220]})$$

Proof. We have already seen \geq in 3.6. On the other hand, the algorithm which pairs off as many elements as possible and then picks the maximal element among the winners and the minimal element among the losers uses precisely this many comparisons. \square

Let $a \geq 2, b \geq 2$. Then

$$C(n; 1, a, b) \leq \begin{cases} (n-1) + \lceil \frac{1}{2}b \rceil + (a-1) \lceil \log(b+1) \rceil, \\ (n-1) + \lceil \frac{1}{2}a \rceil + (b-1) \lceil \log(a+1) \rceil, \end{cases} \tag{6.4}$$

$$C(n; 1, a, b) \leq \begin{cases} (n-1) + a \lceil \log(b+1) \rceil, \\ (2a+1) + (b+1) \lceil \log(a+2) \rceil. \end{cases} \tag{6.5}$$

Proof. By determining the maximal element first we have

$$\begin{aligned} C(n; 1, a, b) &\leq (n-1) + U_a(n-1) - \lceil \frac{1}{2}b \rceil \\ &\leq (n-1) + \lceil \frac{1}{2}b \rceil + (a-1) \lceil \log(b+1) \rceil \end{aligned}$$

since we can use $\lceil \frac{1}{2}b \rceil$ previous comparisons in the algorithm for $(n-1; a, b)$. To prove (6.5) we first determine the partition $(n; a+1, b)$ and then the maximal element of the first block. \square

(6.6) $V_t(n) = C(n; t-1, 1, n-t)$ is the most extensively studied order 3 case. Complete results are known for $t=1, 2, 3$ (see [3, 6, 7, 8]), for the especially interesting median problem $(n; \frac{1}{2}(n-1), 1, \frac{1}{2}(n-1))$, n odd, see e.g. [11, 13].

For partitions $(n; a, b, c)$ with $a, b, c \geq 2, a \leq c$, no general algorithm is known except the obvious one of splitting the set into two parts first and then further partitioning one of the blocks or of determining separately $(n; a, n-a)$ and $(n; n-c, c)$. This yields, e.g., the bound

$$C(n; a, b, c) \leq \lceil \frac{1}{2}(3n-2a-c) \rceil + (a-1) \lceil \log(n-a+1) \rceil + (c-1) \lceil \log(n-c+1) \rceil. \tag{6.7}$$

As an example, for $a=c=1$ we obtain (6.3), which as mentioned there is exact.

As final example let us discuss partitions of the type $(n; a, 1, 1, \dots, 1, b)$ and $(n; 1, \dots, 1, c, 1, \dots, 1)$.

$$C(n; a, 1, \dots, 1, b) \leq \begin{cases} b + a \lceil \log(n - a + 1) \rceil + \sum_{i=b+2}^{n-a} \lceil \log i \rceil \\ a + b \lceil \log(n - b + 1) \rceil + \sum_{i=a+2}^{n-b} \lceil \log i \rceil. \end{cases} \quad (6.8)$$

Proof. We determine first the partition $(n; a, n - a)$ and by using the original tournament single out the 1st, 2nd, ... element of the $(n - a)$ -block one by one. This gives

$$\begin{aligned} C(n; a, 1, \dots, 1, b) &\leq (n - a) + (a - 1) \lceil \log(n - a + 1) \rceil + \sum_{i=b+2}^{n-a+1} \lceil \log i \rceil - (n - a - b) \\ &= b + a \lceil \log(n - a + 1) \rceil + \sum_{i=b+2}^{n-a} \lceil \log i \rceil. \quad \square \end{aligned}$$

For $a = 1$ we obtain the Kisilitsyn bound $K_t(n)$ for $W_t(n)$ which can be slightly improved for a restricted range of t .

For partitions of type

$$(n; \underbrace{1, 1, \dots, 1}_a, c, \underbrace{1, \dots, 1}_b)$$

we may split the set into pairs first and then determine separately

$$(n; \underbrace{1, \dots, 1}_a, n - a) \quad \text{and} \quad (n; n - b, \underbrace{1, \dots, 1}_b)$$

thereby saving a number of comparisons. The best general bound available is

$$C(n; \underbrace{1, \dots, 1}_a, c, \underbrace{1, \dots, 1}_b) \leq (2n - 2^{k-1}) + (a + b - 2)(k - 1) \quad (6.9)$$

for $n = 2^k + r$, $r \geq 1$, $2 \leq a, b \leq 2^{k+1} + 1$ and k large enough.

For $a = 2, b = 1$ this has been further improved.

7. Example

As an illustration let us discuss the case $n = 6$.

$$\begin{aligned} \text{Order 2. } C(6; 1, 5) &= C(6; 5, 1) = 5, \\ C(6; 2, 4) &= C(6; 4, 2) = 7, \\ C(6; 3, 3) &= 7. \end{aligned}$$

All these values are exact (see [2]).

Order 3. $C(6; 1, 1, 4) = C(6; 4, 1, 1) = W_2(6) = 7,$
 $C(6; 1, 2, 3) = C(6; 3, 2, 1) \leq C(5; 1, 2, 2) + 2 = 8, \quad (\text{by (5.2)})$
 $C(6; 1, 3, 2) = C(6; 2, 3, 1) \leq 8, \quad (\text{by (6.4)})$
 $C(6; 1, 4, 1) = 7, \quad (\text{by (6.3)})$
 $C(6; 2, 1, 3) = C(6; 3, 1, 2) = V_3(6) \leq C(5; 2, 1, 2) + 2 = 8,$
 $C(6; 2, 2, 2) \leq C(6; 2, 2, 1) + 3 = 9.$

Order 4. $C(6; 1, 1, 1, 3) = C(6; 3, 1, 1, 1) = W_3(6) = 9,$
 $C(6; 1, 1, 2, 2) = C(6; 2, 2, 1, 1) \leq 5 + (C(5; 1, 2, 2) - 2) = 9,$
 $C(6; 1, 1, 3, 1) = C(6; 1, 3, 1, 1) \leq C(6; 2, 3, 1) + 1 = 9, \quad (\text{by (4.6)})$
 $C(6; 1, 2, 1, 2) = C(6; 2, 1, 2, 1) \leq 5 + (C(5; 2, 1, 2) - 2) = 9,$
 $C(6; 1, 2, 2, 1) \leq 5 + (C(5; 1, 2, 2) - 2) = 9,$
 $C(6; 2, 1, 1, 2) \leq 9.$

The last result is established by the following algorithm. After making 3 disjoint comparisons we match the winners of the first two games and then the loser of this game against the winner of the 3rd game thereby obtaining one of the posets (a) or (b) in Fig. 6.

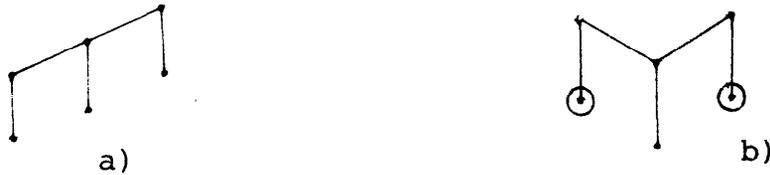


Fig. 6.

If we have (a) then after discarding the top element we need to determine the partition $(5; 1, 1, 1, 2)$. This can be done in $W_3(5) = 7$ comparisons of which 3 relations are already contained in (a). In case (b) we match the two circled elements thus obtaining the poset of Fig. 7 (after 6 comparisons):

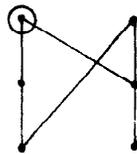


Fig. 7.

Discarding the circled element we now need to determine $(5; 1, 1, 1, 2)$ by using 3 more comparisons and this can be done by merging the vertical left and right chains.

Order 5. $C(6; 1, 1, 1, 1, 2) = C(6; 2, 1, 1, 1, 1) \leq C(6; 1, 2, 1, 2) + 1 = 10$
 $C(6; 1, 1, 1, 2, 1) = C(6; 1, 2, 1, 1, 1) \leq 10$
 $C(6; 1, 1, 2, 1, 1) \leq 10.$

Order 6. $C(6; 1, 1, 1, 1, 1, 1) = 10$.

By using inequalities like $C(6; 2, 3, 1) \geq C(6; 1, 1, 3, 1) - 1$ one can show that all these bounds are exact, except for $C(6; 2, 2, 2)$ where a separate argument is needed.

8. Parallel cost

Let us briefly consider parallel algorithms, i.e., algorithms, where in every round we are allowed to make up to $\lfloor \frac{1}{2}n \rfloor$ disjoint comparisons. $C_p(n; k_1, \dots, k_d)$ denotes then the parallel cost of the partition $(n; k_1, \dots, k_d)$.

Just as in Sections 4 and 5 a few properties of C_p are immediately clear.

$$C_p(n; k_1, \dots, k_d) = C_p(n; k_d, \dots, k_1), \quad (8.1)$$

$$C_p(n; k_1, \dots, k_d) \geq \frac{1}{\lfloor \frac{1}{2}n \rfloor} \left\lceil \log \frac{n!}{k_1! \cdots k_d!} \right\rceil. \quad (8.2)$$

Also, since we are able to make any comparison whatsoever within $n - 1$ rounds if n is even and within n rounds if n is odd, we have

$$C_p(n; k_1, \dots, k_d) \leq \begin{cases} n - 1 & \text{if } n \text{ is even} \\ n & \text{if } n \text{ is odd} \end{cases} \quad \text{for any } (k_1, \dots, k_d). \quad (8.3)$$

We may apparently assume that every round consists of $\lfloor \frac{1}{2}n \rfloor$ matches where some of them might be superfluous because of previously established relations.

Since again

$$C_p(n; k) \leq C_p(n; l) \quad \text{for } k \leq l \text{ in } \mathcal{P}(n) \quad (8.4)$$

we may replace (8.3) by

$$S_p(n) := C_p(n; 1, 1, \dots, 1) \leq \begin{cases} n - 1 & n \text{ even,} \\ n & n \text{ odd.} \end{cases} \quad (8.5)$$

The recursions analogous to (5.1) now read:

$$\begin{aligned} \text{(i)} \quad & C_p(n; k_1, \dots, k_d) \leq C_p(n + 1; k_1, \dots, k_d + 1), \\ \text{(ii)} \quad & C_p\left(\lfloor \frac{1}{2}n \rfloor; k_1, \dots, k_s, \lfloor \frac{1}{2}n \rfloor - \sum_{i=1}^s k_i\right) \leq C_p(n; k_1, \dots, k_d) - 1 \\ & \text{for } \sum_{i=1}^s k_i \leq \lfloor \frac{1}{2}n \rfloor, \end{aligned} \quad (8.6)$$

with the obvious symmetric relations holding as well.

Proof. (i) is clear by the same argument as in the proof of (5.1). In an optimal algorithm for parallel selection of k choose the oracle so that any loser of the first

round loses to any winner (and the possible unmatched player) of that round. With this oracle a parallel algorithm results for $(\lceil \frac{1}{2}n \rceil; k_1, \dots, k_s, \lceil \frac{1}{2}n \rceil - \sum_{i=1}^s k_i)$. \square

Example 4. For $t \leq \lceil \frac{1}{2}n \rceil$ we have

$$\begin{aligned} U_{p,t}(\lceil \frac{1}{2}n \rceil) &\leq U_{p,t}(n) - 1, \\ V_{p,t}(\lceil \frac{1}{2}n \rceil) &\leq V_{p,t}(n) - 1, \quad W_{p,t}(\lceil \frac{1}{2}n \rceil) \leq W_{p,t}(n) - 1. \end{aligned} \quad (8.7)$$

The following results have been established in [3].

$$W_{p,t}(n) \leq \lceil \log n \rceil + (t - 1). \quad (8.8)$$

The analysis in [3] shows that (8.8) probably holds with equality for $n \geq 2^{t-1} + 1$ and fixed t . This conjecture has been verified for $t = 1, 2, 3, 4, 5$.

Let $S_p(n)$ be the parallel cost of the full chain on n elements. Then it is shown in [3] that for $n = 2^k + r$, $0 \leq r < 2^k$, $k \geq 3$

$$S_p(n) \leq \begin{cases} \binom{k+1}{2} & \text{if } r = 0, \\ \binom{k+1}{2} + \lceil \log r \rceil + 1 & \text{if } r > 0. \end{cases} \quad (8.9)$$

The small values are $S_p(3) = S_p(4) = 3$, $S_p(5) = S_p(6) = 5$, $S_p(7) = 6$. Asymptotically, (8.9) means

$$S_p(n) = O(\log^2 n). \quad (8.10)$$

As for lower bounds, it was proved in [3] that

$$S_p(n) \geq \lceil \log n \rceil + \lfloor \log n \rfloor \quad \text{for } n \geq 5. \quad (8.11)$$

It seems reasonable to conjecture that (8.10) is closer to the true growth of $S_p(n)$ than (8.11).

9. Problems

Apart from the obvious problem of computing C and C_p for other types of posets and of considering the average case we list a few problems which are suggested by the posets $\mathcal{O}(n)$ and $\mathcal{P}(n)$.

1. Relate C to other measures of complexity, such as dimension, width or breadth. In particular, are there "unavoidable" edges, i.e., comparisons that must be performed in any optimal algorithm?
2. Suppose P is connected, i.e., $\text{comp } P = 1$. It is true that there always exists an optimal algorithm which makes $\lfloor \frac{1}{2}n \rfloor$ disjoint comparisons first?

3. Are there other relations involving $C(\mathbf{k})$, $C(\mathbf{l})$, $C(\mathbf{k} \vee \mathbf{l})$ and possibly $C(\mathbf{k} \wedge \mathbf{l})$ beyond (4.8)?
4. Let $\mathbf{k}, \mathbf{l} \in \mathcal{P}(n)$. Prove or disprove $r(\mathbf{k}) \leq r(\mathbf{l}) \Rightarrow C(\mathbf{k}) \leq C(\mathbf{l})$.
5. Let $\mathbf{k} = k_1 \cdots k_d$, $\mathbf{l} = l_1 \cdots l_d \in \mathcal{P}(n)$ be of the same rank. Prove or disprove: $e(\mathbf{k}) \geq e(\mathbf{l}) \Rightarrow C(\mathbf{k}) \leq C(\mathbf{l})$.
An affirmative answer would, in particular, imply that the sequences $(U_t(n): t = 1, \dots, n)$ and $(V_t(n): t = 1, \dots, n)$ are unimodal.
6. Prove or disprove equality in (8.8) for $n \geq 2^{t-1} + 1$.
7. What is the growth of $S_p(n)$ for $n \rightarrow \infty$?

References

- [1] M. Aigner, *Combinatorial Theory*, Grundlehren Math. 234 (Springer-Verlag, Berlin, 1979).
- [2] M. Aigner, *Selecting the top three elements*, to appear.
- [3] M. Aigner, *Parallel complexity of sorting problems*, to appear.
- [4] G. Birkhoff, *Lattice Theory*, 3rd ed., Amer. Math. Soc. Coll. Publ., Vol. 25, (AMS, Providence, RI, 1967).
- [5] A. Hadian and M. Sobel, *Selecting the t th largest using binary errorless comparisons*, Coll. Math. Soc. János Bolyai (1969).
- [6] L. Hyafil, *Bounds for selection*, SIAM J. Comput. 5 (1976) 109–114.
- [7] D.G. Kirkpatrick, *Topics in the complexity of combinatorial algorithms*, Comp. Sci. Dept. Techn. Rep. TR 74, Univ. Toronto (1974).
- [8] S.S. Kislitsyn, *On the selection of the k th element of an ordered set by pairwise comparison*, Sibirsk Math. Zk. 5 (1964) 557–564.
- [9] D.E. Knuth, *The Art of Computer Programming*, Vol. 3, *Sorting and Searching* (Addison-Wesley, Reading, MA, 1973).
- [10] A. Schönhage, *The production of partial orders*, Astérisque 38–39 (1976) 229–246.
- [11] A. Schönhage, M. Paterson and N. Pippenger, *Finding the median*, JCSS 13 (1976) 184–199.
- [12] M. Sobel, *On an optimal search for the t best using binary errorless comparisons: The selection problem*, Techn. Rep. No. 114, Dept. Stat. Univ. Minn. (1968).
- [13] C.K. Yap, *New upper bounds for selection*, Comm. ACM. 19 (1976) 501–508.