On the Comparison Cost of Partial Orders

Joseph C. Culberson * Gregory J. E. Rawlins †

October 29, 1992

Abstract

A great deal of effort has been directed towards determining the minimum number of binary comparisons sufficient to produce various partial orders given some partial order. For example, the *sorting problem* considers the minimum number of comparisons sufficient to construct a total order starting from n elements. The *merging problem* considers the minimum number of comparisons sufficient to construct a total order from two total orders. The *searching problem* can be seen as a special case of the merging problem in which one of the total orders is a singleton. The *selection problem* considers the minimum number of comparisons sufficient to select the i^{th} largest of n elements. Little, however, is known about the minimum number of comparisons sufficient to produce an arbitrary partial order. In this paper we briefly survey the known results on this problem and we present some first results on partial orders which can be produced using either restricted types of comparisons or a limited number of comparisons.

1 Introduction

Many comparison-based problems in the analysis of algorithms can been viewed as questions about the optimal production of a particular partial order given another partial order using only binary comparisons. For example, the sorting and merging problems can be seen as questions about the minimum number of binary comparisons sufficient to produce a total order given either a set of n elements or two total orders. The Ford-Johnson sorting algorithm ([11]), and its improvements due to Manacher ([26, 27, 28]), and Bui and Thanh ([7, 40]) yield tight upper bounds on the difficulty of sorting. The merging problem has received a similar amount of attention (see for example, [9, 13, 15, 16, 17, 23, 27, 31, 30, 39]). Much effort has also been expended on finding optimal algorithms to select the i^{th} largest of n elements ([2, 4, 19, 22, 32, 35]); to construct heaps ([8, 12]); to construct priority queues ([6, 41]); to search in a partial or total order ([5, 22, 18, 24, 25]); and so on. However, except for the work of Schönhage ([34]) and Aigner ([1]) little is known about the minimum number of comparisons sufficient to produce an arbitrary partial order.

Schönhage investigated the class of partial orders which model the selection problem assuming an arbitrarily large set of elements to work with. In particular, he investigated the cost of such

*Supported by Natural Sciences and Engineering Research Council Grant No. A-8053. Room 3-38 Assiniboia Hall, Department of Computing Science, University of Alberta, Edmonton, Alberta, T6G2E7, Canada. email: joe@alberta.uucp

†Supported by Natural Sciences and Engineering Research Council Grant No. A-5692. 101 Lindley Hall, Department of Computer Science, Indiana University, Bloomington, Indiana 47405, USA. email: rawlins@iuvax.cs.indiana.edu

partial orders under "mass production," that is, the asymptotic cost of a partial order when we produce arbitrarily many copies of it simultaneously. Aigner investigated the cost of arbitrary partial orders assuming a set of n elements and, in particular, the cost of a class of partial orders which generalizes the selection problem. In this paper we are concerned with the cost of arbitrary partial orders assuming a set of n elements. Since it seems computationally hard to determine the minimum number of comparisons sufficient to produce an arbitrary partial order we only determine upper bounds on the cost of arbitrary partial orders using restricted kinds of comparisons. Further, we investigate a particular class of partial orders, called linear partial orders, which we can construct optimally.

2 Posets

A *partial order* $>$ is an irreflexive and transitive (and hence asymmetric) binary relation. A *partially ordered set*, or *poset* is a structure $(\mathcal{A}, >)$ with a partial order $>$ defined on the elements of $\mathcal{A}$. For convenience we refer to the structure $(\mathcal{A}, >)$ as just $\mathcal{A}$. We use the calligraphic letters $\mathcal{A}, \mathcal{B}, \mathcal{C}$, and $\mathcal{D}$ to stand for posets. All posets discussed are assumed to be *finite* and their elements are assumed to be chosen from a totally ordered set, that is, if $x \neq y$ then either $x > y$ or $y > x$. However, we do not necessarily know this order initially and can only determine whether $x > y$ or $y > x$ by *comparing* x and y. The action of comparing x and y is denoted by $x : y$.

x and y are said to be *related* in $\mathcal{A}$ if either $x > y$ or $y > x$ is in $\mathcal{A}$, otherwise x and y are *unrelated* in $\mathcal{A}$. x *covers* y ($x \succ y$) in $\mathcal{A}$ if $x > y$ in $\mathcal{A}$ and $x > z > y$ in $\mathcal{A}$ implies that $z = x$ or $z = y$. If x and y are unrelated then we write $x \parallel y$. $x \in \mathcal{A}$ is said to be a *singleton* if it is unrelated to every other element in $\mathcal{A}$. The set of singletons of $\mathcal{A}$ is denoted $singletons(\mathcal{A})$. $\mathcal{A}$ is said to be in *reduced form* if it contains no singletons. Two elements form a *pair* if they are only related to each other. The *dual* of $\mathcal{A}$ is the poset $\mathcal{A}^*$ for which $x > y$ in $\mathcal{A}$ if and only if $y > x$ in $\mathcal{A}^*$.

x is said to be a *maximal* element of $\mathcal{A}$ if $\nexists y \ni y > x$ in $\mathcal{A}$. x is said to be a *minimal* element of $\mathcal{A}$ if $\nexists y \ni x > y$ in $\mathcal{A}$. Observe that a singleton is both maximal and minimal. The set of maximals (minimals) of $\mathcal{A}$ is denoted $maximals(\mathcal{A})$ ($minimals(\mathcal{A})$). A *path* in $\mathcal{A}$ is a sequence of elements $x_1, x_2, \ldots, x_{n+1}$ such that $x_i \succ x_{i+1}$ in $\mathcal{A}$ for all $1 \leq i \leq n$. A path consisting of $n+1$ elements is said to have *length* n. $\forall x \in \mathcal{A}$, $depth(x)$ is the length of the longest path in $\mathcal{A}$ which has x as its smallest element. x is said to be a *second maximal* element of $\mathcal{A}$ if $depth(x) = 1$ in $\mathcal{A}$.

A *chain* (*antichain*) is a poset in which all elements are pairwise related (unrelated). The chain and antichain on n elements is denoted by $\mathcal{R}_n$ and $\mathcal{U}_n$, respectively. The *length* of $\mathcal{A}$ is the length of the longest chain in $\mathcal{A}$. The *width* of $\mathcal{A}$ is the maximum number of elements in any antichain in $\mathcal{A}$ minus one. A *linear extension* of $\mathcal{A}$ is a chain $\mathcal{B}$ for which $x > y$ in $\mathcal{B}$ if $x > y$ in $\mathcal{A}$. The set of linear extensions of $\mathcal{A}$ is denoted $extensions(\mathcal{A})$.

$\forall x, y \in \mathcal{A}$, $above(x)$ is the set of elements which are larger than x in $\mathcal{A}$; $above(x \neg y)$ is the set of elements which are larger than x and not y in $\mathcal{A}$; and $above(x \wedge y)$ is the set of elements which are larger than both x and y in $\mathcal{A}$. Similarly, we have $below(x)$, $below(x \neg y)$, and $below(x \wedge y)$. These sets are illustrated for two unrelated elements x and y in figure 1. We will on occasion use $below(x)$ and $above(x)$ *as posets*, that is, the elements in $below(x)$ and $above(x)$ with the inherited partial order.

We represent posets as transitively closed directed graphs $\mathcal{A} = (V(\mathcal{A}), E(\mathcal{A}))$, where $V(\mathcal{A})$ is a set of vertices corresponding to the set of elements in $\mathcal{A}$ and $(x, y) \in E(\mathcal{A}) \iff x > y$ in $\mathcal{A}$. We depict posets using Hasse diagrams, in which the graph is transitively reduced and a relation (x, y)

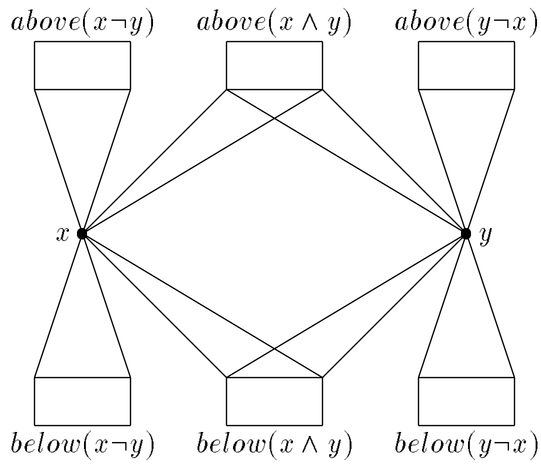

Figure 1: $above(x\neg y)$ and $above(x \wedge y)$

is represented by x being above y and connected to it by a line. We use the terms *dag* and *tdag* to refer to directed acyclic graphs and transitively closed directed acyclic graphs, respectively. We say that $\mathcal{A}$ is *connected* if it is connected when viewed as an undirected graph. The set of connected components of $\mathcal{A}$ is denoted $components(\mathcal{A})$.

3 Operations on Posets

If $x \succ y$ then $\mathcal{A} \setminus (x, y)$ is the poset obtained by deleting the relation (x, y) from $\mathcal{A}$. $\mathcal{A} \setminus x$ is the poset obtained by deleting the element x and all relations in which it appears from $\mathcal{A}$.

Suppose that $x \parallel y$ in $\mathcal{A}$. After comparing x and y, if $x > y$ then we denote the outcome poset by $\mathcal{A}_{x>y}$, else we denote it by $\mathcal{A}_{y>x}$. That is, for each poset $\mathcal{A}$ and $\forall x \parallel y$ in $\mathcal{A}$ we associate two posets $\mathcal{A}_{x>y}$ and $\mathcal{A}_{y>x}$ where, for example,

$$E(\mathcal{A}_{x>y}) = E(\mathcal{A}) \cup \{(u, v) \mid u \in \{x\} \cup above(x\neg y), v \in \{y\} \cup below(y\neg x)\}$$

Given two posets $\mathcal{A}$ and $\mathcal{B}$ the *sum* of $\mathcal{A}$ and $\mathcal{B}$ (denoted $\mathcal{A}+\mathcal{B}$) is the poset obtained by taking the disjoint union of $\mathcal{A}$ and $\mathcal{B}$; the *product* of $\mathcal{A}$ and $\mathcal{B}$ (denoted $\mathcal{A} \times \mathcal{B}$) is the poset obtained from $\mathcal{A}+\mathcal{B}$ by adding the relations (u, v) for all $u \in \mathcal{A}, v \in \mathcal{B}$ (see figure 2). We extend sum and product to finitely many posets via the operators $\sum$ and $\prod$. A poset describable by a finite sequence of sums and products of smaller posets is called a *series-parallel* poset ([38]).

Given two posets $\mathcal{A}$ and $\mathcal{B}$ for each $a \in \mathcal{A}$ we define the *a-composition of* $\mathcal{A}$ *with respect to* $\mathcal{B}$ as follows: replace each $b_i \in \mathcal{B}$ with the poset $\mathcal{A}_i$, where $\mathcal{A}_i$ is isomorphic to $\mathcal{A}$ under some bijection f_i, and for each $b_i < b_j$ in $\mathcal{B}$ add the relations $f_i(a) < f_j(a)$ together with the transitively induced relations (see figure 2).

$$\begin{aligned}\mathcal{A} \otimes_a \mathcal{B} \;&=\; \sum_i \mathcal{A}_i \bigcup \{(f_i(a), f_j(a)) \mid (b_i, b_j) \in \mathcal{B}\} \\ &\bigcup \{(u, v) \mid (b_i, b_j) \in \mathcal{B}, u \in above(f_i(a)), v \in below(f_j(a))\}\end{aligned}$$

We call $\mathcal{A}$ the *base poset*, $\mathcal{B}$ the *pattern poset* and a the *pivot*. A poset describable by a finite sequence of compositions of smaller posets is called a *factorable posets*.

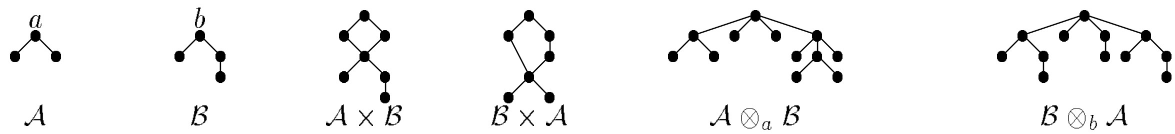

Figure 2: Operations on Posets

4 The Poset of Posets and Comparison Algorithms

Let $\mathcal{P}(n)$ be the set of all posets on n elements. Given two posets $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{P}(n)$ we say that $\mathcal{A}$ *contains* $\mathcal{B}$ ($\mathcal{A} \geq \mathcal{B}$) if there is an order-preserving injection from $\mathcal{B}$ into $\mathcal{A}$. $\mathcal{P}(n)$ forms a poset under containment with $\mathcal{R}_n$ as unique maximal element and $\mathcal{U}_n$ as unique minimal element. If $\mathcal{A} \geq \mathcal{B}$ in $\mathcal{P}(n)$ then we say that $\mathcal{B}$ is a *subposet* of $\mathcal{A}$. If there is an order-preserving bijection between $\mathcal{A}$ and $\mathcal{B}$ then we say that $\mathcal{A}$ and $\mathcal{B}$ are *isomorphic* ($\mathcal{A} \simeq \mathcal{B}$). If we wish to speak of a specific bijection f we use the relation $\mathcal{A} \simeq_f \mathcal{B}$. We use the notation $\mathcal{A} \leq \mathcal{B}$ to mean $\mathcal{B} \geq \mathcal{A}$ and the notation $\mathcal{A} < \mathcal{B}$ to mean that $\mathcal{A} \leq \mathcal{B}$ and $\mathcal{A} \not\simeq \mathcal{B}$. Since $\mathcal{P}(n)$ is a poset, we freely use the notation developed in the previous sections. For example, we say that $\mathcal{A}$ *covers* $\mathcal{B}$ ($\mathcal{A} \succ \mathcal{B}$) in $\mathcal{P}(n)$ if $\mathcal{A} \geq \mathcal{B}$ and and $\mathcal{A} \geq \mathcal{C}, \mathcal{C} \geq \mathcal{B}$ in $\mathcal{P}(n)$ implies that $\mathcal{C} \simeq \mathcal{A}$ or $\mathcal{C} \simeq \mathcal{B}$. Note, however, that there are some differences in interpretation; for example, if $\mathcal{A} \in \mathcal{P}(n)$ then $below(\mathcal{A})$ is the set of posets in $\mathcal{P}(n)$ which are subposets of $\mathcal{A}$ *including* $\mathcal{A}$ *itself*.

A poset $\mathcal{A}$ is *graded* if there exists a function $f : \mathcal{A} \rightarrow \mathbf{N}$ such that $\forall x > y$ in $\mathcal{A}$, $x \succ y$ in $\mathcal{A}$ if and only if $f(x) = f(y) + 1$. Aigner ([1]) makes the following observation which we here prove:

Lemma 4.1 *$\mathcal{P}(n)$ is graded by $f(\mathcal{A}) = |E(\mathcal{A})|$.*

Proof: Consider $\mathcal{A}, \mathcal{B} \in \mathcal{P}(n)$ where $\mathcal{A} > \mathcal{B}$ in $\mathcal{P}(n)$.

Suppose that $\mathcal{A} \succ \mathcal{B}$ in $\mathcal{P}(n)$. Consider any injection of $\mathcal{B}$ into $\mathcal{A}$. There must be $x, y \in \mathcal{A} \ni x \succ y$ in $\mathcal{A}$, and $x \not\succ y$ in the embedding of $\mathcal{B}$ in $\mathcal{A}$ else $\mathcal{A} \simeq \mathcal{B}$. Thus, $\mathcal{A} > \mathcal{A} \setminus (x, y) \geq \mathcal{B}$. Hence, $|E(\mathcal{A})| = |E(\mathcal{A} \setminus (x, y))| + 1 \geq |E(\mathcal{B})|$. If there was another cover $u \succ v$ in $\mathcal{A}$ not in the embedding of $\mathcal{B}$ in $\mathcal{A}$ then there would exist a poset $\mathcal{C} = (\mathcal{A} \setminus (x, y)) \setminus (u, v) \in \mathcal{P}(n)$ such that $\mathcal{A} > \mathcal{C} > \mathcal{B}$ in $\mathcal{P}(n)$, which implies that $\mathcal{A} \not\succ \mathcal{B}$, a contradiction. Hence, there can only be one cover in $\mathcal{A}$ not in any injection of $\mathcal{B}$ in $\mathcal{A}$ and thus, $|E(\mathcal{A})| = |E(\mathcal{B})| + 1$.

Conversely, suppose that $|E(\mathcal{A})| = |E(\mathcal{B})| + 1$. Suppose that there exists a $\mathcal{C}$ such that $\mathcal{A} > \mathcal{C} > \mathcal{B}$ in $\mathcal{P}(n)$. Then, $|E(\mathcal{A})| > |E(\mathcal{C})| > |E(\mathcal{B})|$. Which implies that $|E(\mathcal{A})| > |E(\mathcal{B})| + 1$, a contradiction. Thus, $\mathcal{A} \succ \mathcal{B}$ in $\mathcal{P}(n)$. ∎

Thus, $\mathcal{P}(n)$ is graded into *levels* by the number of relations function and $length(\mathcal{P}(n)) = \binom{n}{2}$.

An *algorithm, $\mathcal{T}$, on* $\mathcal{P}(n)$ is a finite, rooted, labeled, complete binary tree. Each node in $\mathcal{T}$ has an associated poset in $\mathcal{P}(n)$ as its label. If $\mathcal{A}$ is the label of a non-leaf node then for some $x \parallel y$ in $\mathcal{A}$ one of the two children of that node is labeled with the poset $\mathcal{A}_{x>y}$ and the other is labeled with the poset $\mathcal{A}_{y>x}$. Let $leaves(\mathcal{T})$ be the set of leaf labels of $\mathcal{T}$. Define the *output* of $\mathcal{T}$ as follows:

$$output(\mathcal{T}) = \bigcap \{below(\mathcal{A}) \mid \mathcal{A} \in leaves(\mathcal{T})\}$$

$\mathcal{T}$ is said to *produce* $\mathcal{B}$ if $\mathcal{B} \in output(\mathcal{T})$. The *cost of* $\mathcal{B}$ *given* $\mathcal{A}$ is the minimum path length of all possible algorithms on $\mathcal{P}(n)$ which produce $\mathcal{B}$ and whose roots are labeled with $\mathcal{A}$. We denote this number by $C(\mathcal{A};\mathcal{B})$. An algorithm on $\mathcal{P}(n)$ which produces $\mathcal{B}$ given $\mathcal{A}$ and which has minimum path length among all such algorithms on $\mathcal{P}(n)$ is said to be *optimal* for $\mathcal{B}$.

In table 3 we phrase many of the selection and sorting problems in terms of the cost function and series-parallel posets.

Problem	Function to be Determined
Merging	$C(\sum_i \mathcal{A}_i;\mathcal{B})$
Searching	$C(\mathcal{A}+\mathcal{U}_1;\mathcal{B})$
Partition	$C(\mathcal{A};\prod_i \mathcal{B}_i)$
Selection	$C(\mathcal{A};\mathcal{B}\times\mathcal{U}_1\times\mathcal{C})$
Sorting	$C(\mathcal{A};\mathcal{R}_n)$

Figure 3: Sorting Problems

Schönhage investigated the special case of the selection problem—$C(\mathcal{U}_n;\mathcal{U}_i \times \mathcal{U}_1 \times \mathcal{U}_{n-i-1})$; Aigner ([1]) investigated the special case of the partition problem—$C(\mathcal{U}_{\Sigma_i n_i};\prod_i \mathcal{U}_{n_i})$; Linial and Saks ([24, 25]) investigated the searching problem; Linial ([23]) investigated the sorting problem for the special case of $width(\mathcal{A}) = 1$; and Kahn and Saks [18] investigated the general sorting problem.

The following results are straightforward:

Lemma 4.2

$$\begin{aligned}
&\mathcal{A} \simeq \mathcal{A}^* \Longrightarrow C(\mathcal{A};\mathcal{B}) = C(\mathcal{A};\mathcal{B}^*)\\
&\mathcal{A} < \mathcal{B} \Longrightarrow C(\mathcal{C};\mathcal{A}) \le C(\mathcal{C};\mathcal{B})\\
&\mathcal{A} < \mathcal{B} \Longrightarrow C(\mathcal{B};\mathcal{C}) \le C(\mathcal{A};\mathcal{C})\\
&C(\mathcal{A};\mathcal{B}) \ge 0\\
&C(\mathcal{A};\mathcal{B}) \ge |components(\mathcal{A})| - |components(\mathcal{B})|\\
&C(\mathcal{A};\mathcal{B}+\mathcal{C}) \le C(\mathcal{A};\mathcal{B}) + C(\mathcal{U}_{|V(\mathcal{C})|};\mathcal{C})
\end{aligned}$$

The bound $C(\mathcal{A};\mathcal{B}) \ge |components(\mathcal{A})| - |components(\mathcal{B})|$ is the "connectivity" lower bound first mentioned by Kirkpatrick ([19]), from this follows the elementary result that

$$\forall n \ge 1\ , \quad C(\mathcal{U}_n;\mathcal{U}_1 \times \mathcal{U}_{n-1}) = n-1$$

With regard to the last upper bound of lemma 4.2 the reader may conjecture that equality holds in the following relation:

$$C(\mathcal{U}_{\sum_i |V(\mathcal{A}_i)|};\sum_i \mathcal{A}_i) \le \sum_i C(\mathcal{U}_{|V(\mathcal{A}_i)|};\mathcal{A}_i)$$

However, the following examples by Paterson ([35]) and Schönhage ([34]), respectively, demonstrate that this does not hold. In figure 4, $C(\mathcal{U}_7;\mathcal{A}) = 8$, $C(\mathcal{U}_1;\mathcal{B}) = 0$, and $C(\mathcal{U}_5;\mathcal{C}) = 6$; yet $C(\mathcal{U}_8;\mathcal{A}+\mathcal{B}) = 7$ and $C(\mathcal{U}_{10};\mathcal{C}+\mathcal{C}) \le 11$.

$\mathcal{A} \in \mathcal{P}(n)$ is said to be $\mathcal{P}(n)$*-strong* if there is no $\mathcal{B} \in \mathcal{P}(n)$ such that $C(\mathcal{U}_n;\mathcal{A}) = C(\mathcal{U}_n;\mathcal{B})$ and for which $\mathcal{A} < \mathcal{B}$ in $\mathcal{P}(n)$. If $\mathcal{A}$ is not $\mathcal{P}(n)$*-strong* it is said to be $\mathcal{P}(n)$*-weak*. The smallest example of a weak poset is $\mathcal{A} = \mathcal{U}_1 \times \mathcal{U}_3$. This poset is contained in the poset $\mathcal{B} = \mathcal{U}_1 \times (\mathcal{U}_1 + \mathcal{R}_2)$ in $\mathcal{P}(4)$ yet $C(\mathcal{U}_4;\mathcal{A}) = C(\mathcal{U}_4;\mathcal{B}) = 3$. In figure 5, $\mathcal{A} = \mathcal{U}_2 \times \mathcal{U}_3$ is a subposet of $\mathcal{B} = \mathcal{U}_2 \times (\mathcal{U}_1 + \mathcal{R}_2)$ in $\mathcal{P}(5)$ yet $C(\mathcal{U}_5;\mathcal{A}) = C(\mathcal{U}_5;\mathcal{B}) = 5$.

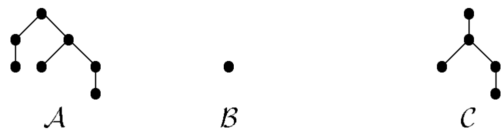

Figure 4: Additivity Counterexamples

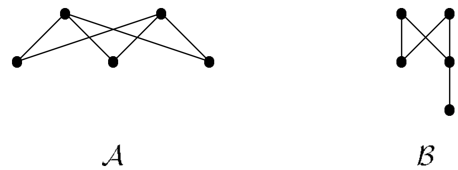

Figure 5: Weak and Strong Posets

5 Previous Work

From the observation that if $x \parallel y$ in $\mathcal{A}$ then

$$|extensions(\mathcal{A})| = |extensions(\mathcal{A}_{x>y})| + |extensions(\mathcal{A}_{y>x})|$$

Schönhage ([34]) showed that[1]

$$C(\mathcal{A};\mathcal{B}) \geq \lceil \lg(|extensions(\mathcal{A})|/|extensions(\mathcal{B})|)\rceil$$

This lower bound generalizes an idea of Hwang and Lin (quoted in [22]).

By observing that

$$\frac{|extensions(\mathcal{A}+\mathcal{B})|}{|extensions(\mathcal{A}\times\mathcal{B})|} = \binom{|V(\mathcal{A})|+|V(\mathcal{B})|}{|V(\mathcal{A})|}$$

We see that this lower bound gives as corollaries the well-known "information-theoretic" bounds on sorting, searching, and merging:

$$\begin{aligned} C(\mathcal{U}_n;\mathcal{R}_n) &\geq \lceil \lg n! \rceil \\ C(\mathcal{U}_1+\mathcal{R}_{n-1};\mathcal{R}_n) &= \lceil \lg n \rceil \\ C(\mathcal{R}_m+\mathcal{R}_{n-m};\mathcal{R}_n) &\geq \left\lceil \lg \binom{n}{m} \right\rceil \end{aligned}$$

In [34] Schönhage also showed that

$$C(\sum_{i=1}^{p}\mathcal{U}_1\times\mathcal{U}_{n_i};\mathcal{R}_2\times\mathcal{U}_{r-t-2}+\mathcal{U}_t) \geq p-t-2+\left\lceil \lg(\sum_{i=1}^{p}2^{n_i}-t)\right\rceil \text{ where } r=\sum_{i=1}^{p}(1+n_i)\ , \quad t, n_i \geq 0 \forall i$$

From this bound we obtain the result first obtained by combining the results of Kislitsyn ([20]) and Schreier ([36]) that

$$\forall n \geq 2\ , \quad C(\mathcal{U}_n;\mathcal{R}_2\times\mathcal{U}_{n-2}) = n + \lceil \lg n \rceil - 2$$

[1] Throughout this paper we use "lg" to mean the logarithm base 2.

From the observation that

$$\mathcal{A} \succ \mathcal{B} \Longrightarrow C(\mathcal{U}_n;\mathcal{A}) \leq C(\mathcal{U}_n;\mathcal{B}) + 1$$

Aigner ([1]) showed that

$$C(\mathcal{U}_n;\mathcal{A}) \leq |E(\mathcal{A})|$$

We give a constructive proof of this result later in the paper.

In [1] Aigner also showed that

$$C(\mathcal{U}_n;\mathcal{A}) \geq \lceil 3n/2 \rceil - |maximals(\mathcal{A})| - |minimals(\mathcal{A})|$$

This result generalizes Pohl's result ([32]) that

$$\forall n \geq 1\ , \quad C(\mathcal{U}_n;\mathcal{U}_1 \times \mathcal{U}_{n-2} \times \mathcal{U}_1) = \lceil 3n/2 \rceil - 2$$

and it may in its turn be generalized to

$$C(\mathcal{A};\mathcal{B}) \geq |maximals(\mathcal{A})| + |minimals(\mathcal{A})| - |maximals(\mathcal{B})| - |minimals(\mathcal{B})| - \lfloor |singletons(\mathcal{A})|/2 \rfloor$$

5.1 Other Results

In the following section we state the best results known for various selection and sorting problems. For each result, the first known appearance in print is given after the result.

Optimal Results:

$$\forall n \geq 2 \qquad C(\mathcal{U}_n;\mathcal{U}_2 \times \mathcal{U}_{n-2}) = n + \lceil \lg(n-1) \rceil - 2 \qquad [37]$$

$$\forall n \geq 6 \qquad C(\mathcal{U}_n;\mathcal{U}_3 \times \mathcal{U}_{n-3}) = n + 2\lfloor \lg n \rfloor - 3 + \begin{cases} 0 & 2^{k-1} < n-2 \leq 2^k \\ 1 & 2^k < n-2 \leq 5.2^{k-2} \\ 2 & \text{otherwise} \end{cases} \qquad [2]$$

$$\forall n \geq 6 \qquad C(\mathcal{U}_n;\mathcal{U}_2 \times \mathcal{U}_1 \times \mathcal{U}_{n-3}) = n + 2\lfloor \lg n \rfloor - 3 + \begin{cases} 0 & 2^{k-1} < n-1 \leq 2^k \\ 1 & 2^k < n-1 \leq 5.2^{k-2} \\ 2 & \text{otherwise} \end{cases} \qquad [2]$$

$$\forall n \geq 6 \qquad C(\mathcal{U}_n;\mathcal{R}_3 \times \mathcal{U}_{n-3}) = n + 2\lfloor \lg n \rfloor - 3 + \begin{cases} 0 & n = 2^k \\ 1 & 2^k < n \leq 5.2^{k-2} \\ 2 & \text{otherwise} \end{cases} \qquad [2]$$

$$\forall n \geq 2 \qquad C(\mathcal{R}_2 + \mathcal{R}_{n-2};\mathcal{R}_n) = \left\lceil \tfrac{7}{12} \lg(n-1) \right\rceil + \left\lceil \tfrac{14}{17} \lg(n-1) \right\rceil \qquad [13, 16]$$

$$\forall m \geq 1 \qquad C(\mathcal{R}_m + \mathcal{R}_{n-m};\mathcal{R}_n) = n - 1\ , \quad \forall\, 2m \leq n \leq 5m/2 + 1 \qquad [39]$$

Let $M(3,n)$ be the largest value that can be merged with 3 elements *in n steps*:

$$M(3,n) = \begin{cases} \left\lfloor \frac{43}{7} 2^{\lfloor n/3 \rfloor - 2} \right\rfloor - 2 & n \equiv 0(3) \\ \left\lfloor \frac{107}{7} 2^{\lfloor n/3 \rfloor - 3} \right\rfloor - 2 & n \equiv 1(3) \\ \left\lfloor \frac{172^{\lfloor n/3 \rfloor} - 6}{7} \right\rfloor - 1 & n \equiv 2(3) \end{cases} \quad , \quad \forall\, n \geq 9 \quad [31, 15]$$

Mönting ([30]) has optimal results for $M(4,n)$ and very tight bounds for $M(5,n)$.

Lower Bounds:

$$C(\mathcal{U}_n;\mathcal{U}_{i-1}\times\mathcal{U}_1\times\mathcal{U}_{n-i}) \geq \begin{cases} n+i-3+\sum_{j=0}^{i-2}\left\lceil \lg\frac{n-i+2}{i+j}\right\rceil & 1\leq i\leq n/3 \\ \left\lfloor\frac{3n+i+1}{2}\right\rfloor - 3 & n/3 < i \leq n/2 \end{cases} \quad [19]$$

$$C(\mathcal{U}_n;\mathcal{U}_{i-1}\times\mathcal{U}_1\times\mathcal{U}_{n-i}) \geq \left\lceil \lg\binom{n}{i}\frac{2^{n-j}}{n-i+1}\right\rceil \text{ where } j = 2\sqrt{\lg\frac{\binom{n}{i}}{n-i+1}+3} - 3 \quad [4]$$

Upper Bounds:

$$\begin{aligned}
C(\mathcal{U}_n;\mathcal{R}_i\times\mathcal{U}_{n-i}) &\leq n-i\ +\sum_{j=n-i+2}^{n}\lceil\lg j\rceil & [20]\\
C(\mathcal{U}_n;\mathcal{U}_{i-1}\times\mathcal{U}_1\times\mathcal{U}_{n-i}) &\leq n-i+(i-1)\lceil\lg(n-i+2)\rceil & [14]\\
C(\mathcal{U}_n;\mathcal{U}_{(n-1)/2}\times\mathcal{U}_1\times\mathcal{U}_{(n-1)/2}) &\leq 3n+o(n) & [35]\\
C(\mathcal{U}_n;\mathcal{R}_n) &\leq n\lceil\lg(3n/4)\rceil - \lfloor 2^{\lfloor\lg 6n\rfloor}/3\rfloor + \lfloor\tfrac{1}{2}\lg 6n\rfloor & [11]\\
C(\mathcal{R}_m+\mathcal{R}_{n-m};\mathcal{R}_n) &\leq m(1+\lfloor\lg k\rfloor)+\lfloor mk/2^{\lfloor\lg k\rfloor}\rfloor - 1,\ \forall n\geq 2m, k=\tfrac{n-m}{m} & [17]
\end{aligned}$$

Recursions:

$$\begin{aligned}
C(\mathcal{U}_n;\mathcal{U}_i\times\mathcal{U}_{n-i}) &\leq C(\mathcal{U}_n;\mathcal{U}_{i-1}\times\mathcal{U}_1\times\mathcal{U}_{n-i})\\
C(\mathcal{U}_n;\mathcal{U}_{i-1}\times\mathcal{U}_1\times\mathcal{U}_{n-i}) &\leq C(\mathcal{U}_n;\mathcal{R}_i\times\mathcal{U}_{n-i})\\
C(\mathcal{R}_m+\mathcal{R}_{n-m};\mathcal{R}_n) &\leq C(\mathcal{R}_{m+1}+\mathcal{R}_{n-m};\mathcal{R}_n)\\
C(\mathcal{R}_{m+k}+\mathcal{R}_{n-m-k};\mathcal{R}_n) &\leq C(\mathcal{R}_m+\mathcal{R}_{n-m-k};\mathcal{R}_n)+C(\mathcal{R}_k+\mathcal{R}_{n-m-k};\mathcal{R}_n)\\
C(\mathcal{R}_m+\mathcal{R}_{n-m};\mathcal{R}_n) &\leq C(\mathcal{R}_m+\mathcal{R}_{\lfloor(n-m)/2\rfloor};\mathcal{R}_{m+\lfloor(n-m)/2\rfloor})+m
\end{aligned}$$

6 Poset Production Processes

We are interested in efficiently generating upper bound costs for large classes of posets. Suppose that we have at our disposal three comparisons and a given poset $\mathcal{A}$. We wish to efficiently determine a set of posets producible from $\mathcal{A}$ within three comparisons. Choose a pair of unrelated elements in $\mathcal{A}$ and attempt to compare them. We shall only retain those posets that are guaranteed (in various senses to be explained later) to be produced by *either* outcome of the comparison.

We then repeat this process for each pair of unrelated elements in $\mathcal{A}$, always retaining only those posets that are guaranteed under our criteria. As a result we will have some set of posets which we can provably produce in one comparison from $\mathcal{A}$.

We use the idea in the proof of lemma 8.1 to produce posets. For each $x \parallel y$, compare x and y and let the poset $\mathcal{A}_x$ as defined in the proof be the poset produced. If $\mathcal{P}_n$ is the set of posets produced in n comparisons, then our algorithm produces the set of posets $\mathcal{P}_{n+1} = \bigcup_{\mathcal{A}\in\mathcal{P}_n}\{\mathcal{A}_x \mid x \parallel y \text{ in } \mathcal{A}\}$ on the $(n+1)^{th}$ comparison. This process is called the *relation construction process*.

We then take each of these posets and perform the same process on each, obtaining a second set of posets. Note that there may be posets which are producible from $\mathcal{A}$ in two comparisons which will *not* be in our set of posets. However, we are guaranteed that this set of posets can be produced in two comparisons starting from $\mathcal{A}$. Finally, using this second set as our input set of posets, we produce the set of posets which can be produced within three comparisons.

We refer to this process of producing new posets as an *Order-1* (O_1) *process*, because we only retain information we have generated from one comparison. An O_2 process would perform two comparisons before looking for the posets that could be guaranteed.

O_2 processes are much more complicated than O_1 processes in that a and b may be unrelated in one outcome of a comparison but be related in the other outcome of the same comparison. For example, if $a < c$ and $d < b$ and we compare c and d then if $c < d$ we can conclude that $a < b$. But if $c > d$ then no relationship between a and b can be inferred. In an O_2 process we can take advantage of this by specifying the second comparison in terms of the result of the preceding one. We then select those posets which are guaranteed (in a manner similar to the as yet unspecified O_1 results).

It would appear that our results in this case are only defined after an even number of comparisons. For example, the set of posets which can be produced in three comparisons starting from some poset $\mathcal{A}$ can be done in two ways. We could produce all the posets possible using one comparison, then using this as our base set produce all the posets possible using the remaining two comparisons in an O_2 manner. Or we could do all possible "double comparisons" first, and find the set of posets, and then do all the single comparisons on this set. We have no guarantee that the results would be identical, and so to be complete we define the O_2 set of posets obtainable in n comparisons to be the union of the sets obtainable through a "double comparison" from $n-2$ comparisons and those obtained from $n-1$ comparisons in one comparison.

In general, the complexity for O_k processes rises with k. We use the notation $O_k(S,n)$ to mean the set of posets obtainable from an initial set S in n comparisons using an O_k process. Then,

$$O_k(S,n) = \begin{cases} O_n(S,n) & 1 \leq n \leq k \\ \cup_{i=1}^{k} O_i(O_k(S,n-i),i) & n > k \end{cases}$$

Finally, $O_k(S,k)$ is generated by performing all the possible k-sequences of comparisons, with the choice of comparisons $2 \ldots k$ depending upon the outcomes of preceding comparisons.

This paper is restricted to an examination of O_1 processes only. Clearly, if a poset is accepted after an O_1 comparison then it will be accepted after an O_i comparison sequence for any $i \geq 2$. Our aim is to derive *upper bounds* on the cost of producing an arbitrary poset. The *O_i-cost* of an n element poset $\mathcal{A}$ is the minimum number of O_i-comparisons necessary to produce $\mathcal{A}$ starting from $\mathcal{U}_n$. Since a poset on n elements can only contain at most $\binom{n}{2}$ relations then if we were to perform an $O_{\binom{n}{2}}$ process starting on $\mathcal{U}_n$ we would produce all possible posets on n elements in the minimum number of comparisons possible.

7 The Full Containment Process

The first poset production process we consider is the *full containment process* defined as follows: A comparison $x : y$ on a poset $\mathcal{A}$ is accepted if either $\mathcal{A}_{x>y} \geq \mathcal{A}_{y>x}$ or $\mathcal{A}_{y>x} \geq \mathcal{A}_{x>y}$. Otherwise, we say the comparison did not produce any new poset and we do not count it. Clearly, if x and y belong to two disjoint isomorphic components of $\mathcal{A}$ then $\mathcal{A}_{x>y} \simeq \mathcal{A}_{y>x}$ and the comparison $x : y$ is always accepted.

Lemma 7.1 *If $\mathcal{A} \simeq_f \mathcal{A}$ where f is a non-trivial automorphism then then $\forall x$ in $\mathcal{A}$, $\mathcal{A}_{x>f(x)} \simeq \mathcal{A}_{f(x)>x}$.*

Proof: Trivial. ∎

We call these last comparisons *equal environment* comparisons. Note that in general $\mathcal{A}$ and $\mathcal{B}$ need not necessarily be disjoint.

Initially, we start with $\mathcal{U}_n$. The outcomes of a comparison between any two singletons are isomorphic, and the resulting poset consists of a pair and $n-2$ singletons. Now we may compare a singleton with another singleton; the maximal element; or the minimal element. In each case the comparison is accepted, that is, there is an outcome which contains the other (the second row of figure 13 gives the three outcomes ignoring a dual poset).

For each of these three posets we may compare any two unrelated elements. The comparison may be between two non-singletons; a singleton and a non-singleton; or two singletons. There are nine posets added at this step (see the third row of figure 13). In the fourth and fifth comparison we produce 27 and 81 new posets, respectively. This sequence 1, 3, 9, 27, 81 is very suggestive. Unfortunately, in the sixth comparison we produce 283 not 243 new posets. Surprisingly, there are precisely 1, 3, 9, 29, and 92 posets which cost 1, 2, 3, 4 and 5 comparisons, respectively. Thus of the first 134 posets ordered by cost starting from $\mathcal{U}_n$ the O_1 full containment process produces all but 15 optimally (seven of these are duals of others).

The algorithm illustrated in figure 6 produces the first two posets for which the O_1 full containment process does not give the exact cost. Neither of the two outcomes of the first comparison (top dashed line) in figure 6 is contained in the other yet poset $\mathcal{A}$ in figure 7 is contained in both outcomes. Thus, this poset is producible in 4 comparisons (and this is optimal), yet it cannot be produced by an O_1 full containment process in less than 5 comparisons.

Further, in one more comparison on both of the two leaves of this algorithm (bottom two dashed lines in the figure), we have another algorithm which produces poset $\mathcal{B}$ in figure 7. Thus, this poset is producible in 5 comparisons (and this is optimal), yet it cannot be produced by an O_1 full containment process in less than 6 comparisons. However, *in the O_2 full containment process* $\mathcal{B}$ can be produced optimally in 5 comparisons.

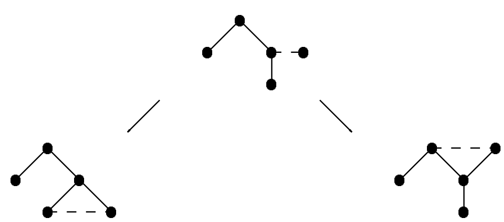

Figure 6: A Simple Algorithm

$\mathcal{A}$ $\mathcal{B}$

Figure 7: The First Two Failures of O_1 Full Containment

THEOREM **7.1** *Let x be unrelated to y in $\mathcal{A}$.*
If $above(x) = above(x \wedge y)$ and $below(y) = below(x \wedge y)$ then $\mathcal{A}_{x>y} \leq \mathcal{A}_{y>x}$.

Proof: If $above(x) = above(x \wedge y)$ and $below(y) = below(x \wedge y)$ then the poset produced from $\mathcal{A}_{y>x}$ by deleting (u,v) for all $u \in \{x\} \cup above(y)$, $v \in below(x \neg y)$, and $u \in above(y)$, $v \in \{y\} \cup below(x \neg y)$, is isomorphic to $\mathcal{A}_{x>y}$. Thus $\mathcal{A}_{x>y} \leq \mathcal{A}_{y>x}$. ∎

Corollary 7.1 *If x and y are maximal and minimal elements of $\mathcal{A}$ where $x \parallel y$ in $\mathcal{A}$ then $\mathcal{A}_{x>y} \leq \mathcal{A}_{y>x}$.*

Proof: Since y is minimal, $below(y) = below(x \wedge y) = \emptyset$. ∎

Corollary 7.1 has been independently discovered by Atkinson ([3]).

Corollary 7.2 *If x is a maximal element and y is a singleton in $\mathcal{A}$ then $\mathcal{A}_{x>y} \leq \mathcal{A}_{y>x}$.*
Dually, if x is a minimal element and y is a singleton in $\mathcal{A}$ then $\mathcal{A}_{y>x} \leq \mathcal{A}_{x>y}$.

Proof: A singleton may be treated as a maximal or minimal element. ∎

THEOREM **7.2** *Let x be a maximal element of $\mathcal{A}$ and let y be an element unrelated to x in $\mathcal{A}$.*
If $below(y) \simeq below(x)$ then $\mathcal{A}_{x>y} \leq \mathcal{A}_{y>x}$.
Dually, if x is a minimal element and $above(y) \simeq above(x)$ then $\mathcal{A}_{y>x} \leq \mathcal{A}_{x>y}$.

Proof: ∎

Thus, under the full containment process, we can always make progress by comparing equal environment elements; maximals and minimals; or maximals and minimals with various classes of elements. We now show that we are guaranteed to produce every poset using the O_1 full containment process starting from $\mathcal{U}_n$.

Lemma 7.2 *In any connected non-singleton poset $\mathcal{A}$, there exists x, y, where x is maximal and $x \succ y$ in $\mathcal{A}$, such that $\mathcal{A} \setminus (x, y)$ is either*

1. *a connected poset, or*
2. *a connected poset and a singleton.*

Proof: The proof is by induction on $|E(\mathcal{A})|$.

Basis: The lemma is clearly true for $|E(\mathcal{A})| = 1$.

Induction: Assume that the lemma is false for some connected poset $\mathcal{A}$ with $|E(\mathcal{A})| > 1$. Select any (x, y) where x is maximal and $x \succ y$. Suppose that deleting (x, y) disconnects $\mathcal{A}$ into connected components $\mathcal{B}$ and $\mathcal{C}$ with x and y in separate components, where neither $\mathcal{B}$ nor $\mathcal{C}$ are singletons.

Consider any (u, v), where u is maximal and $u \succ v$ in $\mathcal{C}$. However, u is maximal and $u \succ v$ in $\mathcal{A}$ also, otherwise deleting (x, y) could not disconnect $\mathcal{A}$. Also, y must be minimal in $\mathcal{A}$, otherwise the deletion of (x, y) could not disconnect $\mathcal{A}$. Thus, re-inserting (x, y) will not change the depth of u or v.

Thus deleting (u, v) from $\mathcal{C}$ will disconnect $\mathcal{C}$, since, deleting (u, v), by assumption, disconnects $\mathcal{A}$.

Deleting (u, v) from $\mathcal{C}$, and replacing (x, y) will join exactly one of the components from $\mathcal{C}$ to $\mathcal{B}$. Thus, deleting (u, v) from $\mathcal{A}$ results in two components $\mathcal{B}'$ and $\mathcal{C}'$, with $|V(\mathcal{C}')| < |V(\mathcal{C})|$. We repeat this argument on $\mathcal{B}'$ and $\mathcal{C}'$. Eventually, since $|V(\mathcal{A})|$ is finite, we must find a $\mathcal{C}''$ such that $|V(\mathcal{C}'')| < 2$, a contradiction. ∎

THEOREM **7.3** *The O_1 full containment process produces every poset.*

Proof: For each connected component of the poset to be produced delete its covers one at a time without violating the conditions of lemma 7.2. To construct each connected component, do the comparisons in reverse order. ■

The posets in figure 7 demonstrate that although every poset is guaranteed to be produced, they may not be produced *optimally*. Poset $\mathcal{A}$ costs 4 comparisons yet can only be built in 5 O_1 full containment comparisons. Similarly, poset $\mathcal{B}$ costs 5 comparisons yet can only be built in 6 O_1 full containment comparisons.

8 The Relation Construction Process

The following lemma demonstrates how to construct a poset with at least one more relation than a given poset by comparing any two unrelated elements.

Lemma 8.1 *If $x \parallel y$ in $\mathcal{A}$ then $\exists \mathcal{B} \ni \mathcal{B} \leq \mathcal{A}_{x>y} \cap \mathcal{A}_{y>x}$ and $|E(\mathcal{B})| = |E(\mathcal{A})| + 1$.*

Proof: Compare x and y and form the poset $\mathcal{A}_x$ from $\mathcal{A}_{x>y}$ by replacing (u, x) with (u, y) for all $u \in above(x \neg y)$ and (y, v) with (x, v) for all $v \in below(y \neg x)$. Similarly, form the poset $\mathcal{A}_y$ from $\mathcal{A}_{y>x}$. Thus, for example,

$$E(\mathcal{A}_x) = E(\mathcal{A}_{x>y}) \setminus (\{(u, x) \mid u \in above(x \neg y)\} \cup \{(y, v) \mid v \in below(y \neg x)\})$$

We demonstrate in lemma 8.2 below that $\mathcal{A}_x$ and $\mathcal{A}_y$ are posets. That is, the above construction process is not inconsistent. The Hasse diagram of $\mathcal{A}_x$ for figure 1 is shown in figure 8.

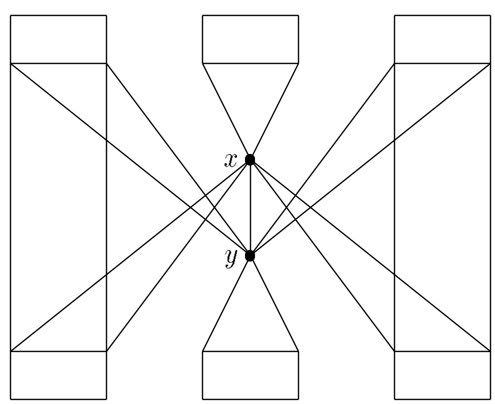

Figure 8: Comparing Unrelated Elements Always Adds A Relation

It is straightforward to show that: $\mathcal{A}_x \simeq \mathcal{A}_y$; $\mathcal{A}_x \leq \mathcal{A}_{x>y} \cap \mathcal{A}_{y>x}$; and, that $|E(\mathcal{A}_x)| = |E(\mathcal{A})| + 1$. ■

Note that more than one new relation may be in a subposet of the two outcomes. For example, when comparing the maxima of two pairs we gain two relations since the outcomes are isomorphic without removing any relations (this is an equal environment comparison). However, the above relation construction process does not take this into account. Observe also that although $\mathcal{A}$ and $\mathcal{A}_x$ will appear on adjacent levels of $\mathcal{P}(n)$ ($\mathcal{P}(n)$ is graded by $|E(\mathcal{A})|$) we cannot then conclude that $\mathcal{A} \leq \mathcal{A}_x$ in $\mathcal{P}(n)$. Similarly, if $\mathcal{A} \leq \mathcal{B}$ in $\mathcal{P}(n)$ we cannot then conclude that $|E(\mathcal{A})| \geq |E(\mathcal{B})| + 1$.

As an example of the utility of the relation construction process note that the following recursion, first observed by Sobel ([37]), follows immediately:

$$C(\mathcal{U}_n; \mathcal{U}_i \times \mathcal{U}_{n-i}) \leq C(\mathcal{U}_{n-1}; \mathcal{U}_{i-1} \times \mathcal{U}_1 \times \mathcal{U}_{n-i-1}) + 1$$

Several results follow from this algorithm.

Lemma 8.2 *$\mathcal{A}_x$ is a poset.*

Proof: The proof is by induction on $|V(\mathcal{A})|$.

Basis: The lemma is easily checked for $|V(\mathcal{A})| \leq 4$.

Induction: We are concerned with what the algorithm does when changing the relations of $\mathcal{A}$. For all $(u, x) \in E(\mathcal{A})$, the algorithm adds (u, y) and deletes (u, x), unless $(u, y) \in E(\mathcal{A})$. Similarly, for all $(y, u) \in E(\mathcal{A})$ the algorithm adds (x, u) and deletes (y, u), unless $(x, u) \in E(\mathcal{A})$. Finally, (x, y) is added. No other changes are made. We need to show that if $(u, v), (v, w) \in E(\mathcal{A}_x)$ then $(u, w) \in E(\mathcal{A}_x)$.

First, consider (x, y). If $(u, x) \in E(\mathcal{A}_x)$, then $u \in above(x \wedge y)$ and thus $(u, y) \in E(\mathcal{A}_x)$. The argument is similar for the relations (y, u).

Second, we consider the deletion of relations (u, x). Suppose that such a deletion leaves $(u, v), (v, x) \in E(\mathcal{A}_x)$. If $(v, x) \in E(\mathcal{A}_x)$ then $v \in above(x \wedge y)$ and thus $(v, y) \in E(\mathcal{A})$. This implies that $(u, y) \in E(\mathcal{A})$ and so $u \in above(x \wedge y)$. But then (u, x) would not be deleted, a contradiction. A similar argument holds for the deletion of relations (y, u).

Finally, consider the addition of relations (u, y). First, if there is a relation (v, u) then there must have existed (v, x). Then either (v, y) is added or $v \in above(x \wedge y)$. In either case $(v, y) \in E(\mathcal{A}_x)$. Second, if there is a relation $(y, v) \in E(\mathcal{A}_x)$, then there must be a relation $(x, v) \in E(\mathcal{A}_x)$ by an argument symmetric to the argument for the deletion of relations (u, x) above. A similar argument holds for the addition of relations (x, u). ■

THEOREM **8.1** *The relation construction process produces all posets with n relations in n comparisons.*

Proof: The proof is by induction on $|E(\mathcal{A})|$.

Basis: The lemma is easily checked for $|E(\mathcal{A})| \leq 4$.

Induction: Assume that poset $\mathcal{A}$ with n relations is the smallest poset not produced in n comparisons. Since $\mathcal{A}$ is finite it must have at least one maximal element x. Let y be one of the elements x covers. Since $\mathcal{A}$ is a smallest counterexample, $\mathcal{A} \setminus (x, y)$ is produced by the algorithm in at most $n - 1$ comparisons.

Observe that $below(y) = below(x \wedge y)$ and that x is maximal in $\mathcal{A} \setminus (x, y)$, therefore by theorem 7.1 if we compare x, and y then $\mathcal{A}$ will be produced in no more than n comparisons, a contradiction. ■

The relation construction process provides a constructive proof of Aigner's result ([1]) that $C(\mathcal{U}_n; \mathcal{A}) \leq |E(\mathcal{A})|$.

9 The Top-Down Construction Process

THEOREM **9.1** *Every connected poset with n relations can be produced in n comparisons using the relation construction process, all of which involve a maximal element and either another maximal element or a second maximal element. Further, both elements must be in the same component or at least one is a singleton.*

Proof: Following lemma 7.2, delete the relations one at a time without violating the conditions. To construct the poset, do the comparisons in reverse order. ■

This theorem leads to the *top-down process*, in which we only construct connected posets using connected posets at each stage of the construction, and, our comparisons are restricted to being comparisons between maximals and maximals or maximals and second maximals. It is an interesting exercise to work back from $\mathcal{R}_n$, and observe the order in which the comparisons are done to produce this poset. This is just sorting by repeatedly finding the minimum of the remaining elements.

Observe that this is a very restrictive construction process. For example, we cannot produce $\mathcal{U}_1 \times (\mathcal{U}_1 + \mathcal{R}_2)$ in 3 comparisons using this process since we are not allowed to compare the maxima of two pairs (at least one must be a singleton if there are to be two components).

CONJECTURE **9.1** *Every connected poset with n relations can be produced using the relation construction process in n comparisons under the following conditions:*

1. *all comparisons involve a maximal element and either another maximal element or a second maximal element,*

2. *the compared elements are in the same component or at least one is a singleton,*

3. *there is an initial phase in which every comparison involves at least one singleton, and,*

4. *there is a final phase in which no comparison involves any singletons.*

What this means is that we first do comparisons between singletons and non-singletons until we have added the correct number of elements to form the target poset, then we restrict ourselves to comparisons between non-singletons. All of these comparisons obey the above rules.

This can be stated as a two phase algorithm. To produce all connected posets with m relations and n elements:

1. Produce all posets with $n-1$ relations and n elements.

2. Perform depth 0 and 1 comparisons as above until m relations have been created.

10 The Maximal Subposet Process

Call two posets *unrelated* in $\mathcal{P}(n)$ if neither is contained in the other in $\mathcal{P}(n)$. The algorithm in figure 9 shows that there can be distinct posets $\mathcal{A}, \mathcal{B}, \mathcal{C}$ such that $\mathcal{B}$ and $\mathcal{C}$ are unrelated maximal subposets of $\mathcal{A}_{x>y} \cap \mathcal{A}_{y>x}$ for some $x \parallel y$ in $\mathcal{A}$. Here, $\mathcal{A} = \mathcal{U}_1 \times \mathcal{U}_2 + \mathcal{U}_1 \times (\mathcal{U}_1 + \mathcal{R}_2)$, $\mathcal{B} = \mathcal{U}_1 \times (\mathcal{U}_3 + \mathcal{U}_1 \times \mathcal{U}_2)$, $\mathcal{C} = \mathcal{U}_1 \times (\mathcal{U}_2 + \mathcal{R}_2 + \mathcal{R}_2)$ (see figure 10).

Observations: Let $\mathcal{D}$ be the "one relation added" poset. Clearly, $\mathcal{D} \leq \mathcal{A}_{x>y} \cap \mathcal{A}_{y>x}$, but

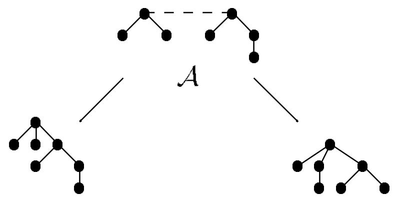

Figure 9: Producing Distinct Maximal Posets

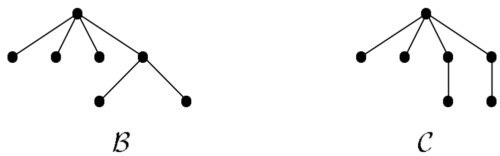

Figure 10: Two Distinct Maximal Subposets

1. $\mathcal{D}$ is not necessarily maximal in the intersection (a counterexample is $\mathcal{A} = \mathcal{R}_2 + \mathcal{R}_2, \mathcal{D} = \mathcal{U}_1 \times \mathcal{U}_3$) and,
2. $\mathcal{D}$ is not necessarily a subposet of $\mathcal{A}$ (a counterexample is $\mathcal{A} = \mathcal{R}_3 + \mathcal{U}_1, \mathcal{D} = \mathcal{U}_2 \times \mathcal{U}_2$).

Note that this last observation is also true of $\mathcal{B}$ and $\mathcal{C}$, for example, the counterexample also applies to maximal elements of the intersection, since for that example $\mathcal{D}$ is the unique maximal element of the intersection. As a converse, observe that we also cannot assume that $\mathcal{A} \leq \mathcal{B}$ or $\mathcal{A} \leq \mathcal{C}$ as the same counterexample shows.

Conjecture **10.1** $\mathcal{D} \leq \mathcal{B} \cap \mathcal{C}$

We now pose the following related question: Given an integer k and two unrelated posets $\mathcal{A}$ and $\mathcal{B}$ each on n elements such that:

- $C(\mathcal{U}_n; \mathcal{A}) = C(\mathcal{U}_n; \mathcal{B})$,
- $\mathcal{A}$ produces outcome posets $\mathcal{A}_1, \mathcal{A}_2, \ldots$, in k comparisons
- $\mathcal{B}$ produces outcome posets $\mathcal{B}_1, \mathcal{B}_2, \ldots$, in k comparisons
- $\forall \mathcal{A}_i \;\; \exists \mathcal{B}_j \;\; \ni \mathcal{B}_j \geq \mathcal{A}_i$,
- $\exists \mathcal{B}_i \ni \forall \mathcal{A}_j \;\; \mathcal{A}_j \not\geq \mathcal{B}_i$,

then we say $\mathcal{B}$ is *k-fecund* with respect to $\mathcal{A}$. Are there any fecund posets?

11 Linear and Max-Min Posets

A connected poset on n elements is said to be *linear* if it costs $n-1$ comparisons to build starting from $\mathcal{U}_n$. Observe that it costs at least $n-1$ comparisons to produce a connected poset on n nodes. Surprisingly, when viewed as an undirected graph, the Hasse diagram of a linear poset is not necessarily a tree. Poset $\mathcal{A}$ in figure 7 is the first example of this, since $C(\mathcal{U}_5; \mathcal{A}) = 4$ yet it is a connected poset on 5 elements.

Lemma 11.1 *Any length 0 or length 1 poset on n elements with $n-1$ relations, is a linear poset.*

Proof: Straightforward. ■

Linear posets have the following *linearity property* with respect to the cost function:

Lemma 11.2 *If $\{\mathcal{A}_i\}$ is any set of linear posets then*

$$C(\mathcal{U}_{\sum |V(\mathcal{A}_i)|}; \sum \mathcal{A}_i) = \sum C(\mathcal{U}_{|V(\mathcal{A}_i)|}; \mathcal{A}_i)$$

Proof: The lemma follows from the fifth and the sixth results of lemma 4.2. ■

This lemma has the important consequence that it is unnecessary to determine the exact cost of disconnected posets whose separate components are all linear. Thus, some of the posets listed in figure 13 are redundant since we may deduce their cost directly from their components.

The following construction scheme produces all factorable linear posets: Make $k \geq 2$ copies of any linear poset on n elements. Construct a linear poset on any k equal environment elements chosen from each of the k posets. By the above lemma the resultant poset on kn elements is linear.

Observe that we may use the same construction scheme to produce posets without first starting with a linear poset. Indeed, the initial poset which we duplicate may even be *disconnected*, all that is necessary is that all comparisons involve equal environment elements within one component.

A poset is said to be a *max-min poset* if it has a unique maximal element and a unique minimal element and it costs $\lceil 3n/2 \rceil - 2$ comparisons to build starting from $\mathcal{U}_n$.

Max-min posets share a *decomposability* property with linear posets, in that, two max-min posets on n elements can be combined to yield a new max-min poset (for n even).

Also, there is a restricted version of the same linearity property with respect to the cost function:

Lemma 11.3 *If $\{\mathcal{A}_i\}$ is any set of max-min posets each on an even number of elements then*

$$C(\mathcal{U}_{\sum |V(\mathcal{A}_i)|}; \sum \mathcal{A}_i) = \sum C(\mathcal{U}_{|V(\mathcal{A}_i)|}; \mathcal{A}_i)$$

Proof: Straightforward. ■

Lemma 11.4 *If $\mathcal{A}$ is a linear or max-min poset on n elements then length$(\mathcal{A}) \leq \lfloor \lg n \rfloor$.*

Proof: ■

The advantage of characterizing special classes of posets whose cost is known is that these characterizations can then be used to infer bounds on the cost of arbitrary posets. For example, given a connected poset on n elements whose length is greater than $\lfloor \lg n \rfloor$ we know immediately that this poset must cost at least n comparisons to build starting from $\mathcal{U}_n$.

Atkinson ([3]) has independently shown that any sequence of $n-1$ comparisons which results in a connected poset on n elements has length no more than $\lfloor \lg n \rfloor$.

12 Open Questions

1. We have shown that each of the production processes produces the poset $\mathcal{A}$ in at most $|E(\mathcal{A})|$ comparisons. However, this is a very crude upper bound; improve this bound to something which is $O(\lg(n!/|extensions(\mathcal{A})|))$.

   How does the structure of $\mathcal{M}(n)$ determine which posets can be produced after c comparisons using either of the construction schemes we have given?

2. Find a relation between $C(\mathcal{A})$, $C(\mathcal{A} \setminus (x, y))$ and $C(\mathcal{A} \setminus z)$ where $x \succ y$ and z is maximal or minimal in $\mathcal{A}$.

3. Table 12 lists the numbers of reduced posets on n elements which cost c comparisons to build for small n and c.

   Is the sequence of trailing digits 1, 2, 8, 28, ... related to the number of reduced posets of each cost (1, 3, 9, 29, ...)? Are there at least 3^{c-1} posets of cost c? Is each column and row unimodal? If so, does the row mode always occur at $c = n - 1$?

   How many strong, linear, and strong linear posets are there on n elements? How many reduced posets are there on n elements which cost c comparisons to build starting from $\mathcal{U}_n$?

4. If a connected poset costs c comparisons and, among all connected posets of cost c it has the maximum number of relations, is it necessarily linear? As a special case we ask a question that belongs to folklore and was first posed by Aigner ([1]). If we wish to build a particular connected poset does there always exist an optimal algorithm which starts with $\lfloor n/2 \rfloor$ comparisons between singletons?

References

[1] Aigner, M.; "Producing Posets," *Discrete Mathematics,* **35**, 1-15, 1981.

[2] Aigner, M.; "Selecting the Top Three Elements," *Discrete Applied Mathematics,* **4**, 247-267, 1982.

[3] Atkinson, M. D.; "The Complexity of Orders," *Proceedings of the NATO Advanced Study Institute on Algorithms and Order,* to appear, 1988.

[4] Bent, S. W. and John, J. W.; "Finding the Median Requires $2n$ Comparisons," *Proceedings of the* 17^{th} *Annual ACM Symposium on Theory of Computing,* 213-216, 1985.

[5] Borodin, A., Guibas, L. J., Lynch, N. A. and Yao, A. C.; "Efficient Searching Using Partial Ordering," *Information Processing Letters,* **12**, 71-75, 1981.

[6] Brown, M. R.; "Implementation and Analysis of Binomial Queue Algorithms," *SIAM Journal of Algorithms,* **7**, 298-319, 1978.

[7] Bui, T. D. and Thanh, M.; "Significant Improvements to the Ford-Johnson Algorithm for Sorting," **BIT**, **25**, 70-75, 1985.

[8] Carlsson, S.; "Heaps," Doctoral Dissertation, Lund University, Sweden, 1987.

[9] Christen, C.; "Improving the Bounds on Optimal Merging," *Proceedings of the* 19^{th} *Annual IEEE Symposium on Foundations of Computer Science,* 259-266, 1978.

[10] Dhar, D.; "Asymptotic Enumeration of Partially Ordered Sets," *Pacific Journal of Mathematics,* **90**, 299-305, 1980.

[11] Ford, L. R., Jr. and Johnson, S. M.; "A Tournament Problem," *American Mathematical Monthly,* **66**, 387-389, 1959.

[12] Gonnet, G. H. and Munro, J. I.; "Heaps on Heaps," *SIAM Journal of Computing,* **15**, 964-971, 1986.

[13] Graham, R. L.; "On Sorting By Comparisons," *Proceedings of the* 2^{nd} *Atlas Conference,* 263-269, 1971.

[14] Hadian, A. and Sobel, M.; "Selecting the t^{th} Largest Using Binary Errorless Comparisons," *Colloquia Mathematica Societatis János Bolyai,* **4**, 585-599, 1969.

[15] Hwang, F. K.; "Optimal Merging of 3 Elements with n Elements," *SIAM Journal of Computing,* **9**, 298-320, 1980.

[16] Hwang, F. K. and Lin, S.; "Optimal Merging of 2 Elements with n Elements," *Acta Informatica,* **1**, 145-158, 1971.

[17] Hwang, F. K. and Lin, S.; "A Simple Algorithm for Merging Two Disjoint Linearly-Ordered Sets," *SIAM Journal of Computing,* **1**, 31-39, 1972.

[18] Kahn, J. and Saks; "Balancing Poset Extensions," *Order*, **1**, 113-126, 1984.

[19] Kirkpatrick, D. G.; "A Unified Lower Bound for Selection and Set Partitioning Problems," *Journal of the ACM,* **28**, 150-165, 1981.

[20] Kislitsyn, S. S.; "On The Selection of the k^{th} Element of an Ordered Set by Pairwise Comparison," (in Russian) *Sibirskii Mat. Zhurnal,* **5**, 557-564, 1964.

[21] Kleitman, D. J. and Rothschild, B. L.; "Asymptotic Enumeration of Partial Orders," *Transactions of the American Mathematical Society,* **205**, 205-220, 1975.

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

[23] Linial, N.; "The Information Theoretic Bound is Good for Merging," *SIAM Journal of Computing,* **13**, 795-801, 1984.

[24] Linial, N. and Saks, M.; "Searching Ordered Structures," *Journal of Algorithms,* **6**, 86-103, 1985.

[25] Linial, N. and Saks, M.; "Every Poset Has a Central Element," *Journal of Combinatorial Theory Series A.,* **40**, 195-210, 1985.

[26] Manacher, G. K.; "The Ford-Johnson Sorting Algorithm is Not Optimal," *Journal of the ACM,* **26**, 441-456, 1979.

[27] Manacher, G. K.; "Significant Improvements to the Hwang-Lin Merging Algorithm," *Journal of the ACM,* **26**, 434-440, 1979.

[28] Manacher, G. K.; "Further Results on Near-Optimal Sorting," *Proceedings of the Allerton Conference on Communication, Control and Computing,* 949-960, 1979.

[29] Mohring, R. H.; "Algorithmic Aspects of Comparability Graphs and Interval Graphs," *Graphs and Order: The Role of Graphs in the Theory of Ordered Sets and Applications,* (I. Rival ed.) NATO ASI Series, Series C: Mathematical and Physical Sciences Vol. 147, 41-102, 1984.

[30] Mönting, J. S.; "Merging of 4 and 5 Elements with n Elements," *Theoretical Computer Science,* **14**, 19-37, 1981

[31] Nisselbaum, Y.; "On Merging N Ordered Elements With Three Elements," *Sigact News,* 14-16, Winter 1978.

[32] Pohl, I.; "A Sorting Problem and Its Complexity," *Communications of the ACM,* **15**, 462-464, 1972.

[33] Prömel, H. J.; "Counting Unlabeled Structures," *Journal of Combinatorial Theory Series A.,* **44**, 83-93, 1987.

[34] Schönhage, A.; "The Production of Partial Orders," *Astérisque,* **38-39**, 229-246, 1976.

[35] Schönhage, A., Paterson, M. and Pippenger, N.; "Finding the Median," *Journal of Computer and System Sciences,* **13**, 184-199, 1976.

[36] Schreier, J.; "On Tournament Elimination Systems," (in Polish) *Mathesis Polska,* **7**, 154-160, 1932.

[37] Sobel, M.; "On an Optimal Search for the t Best Using Binary Errorless Comparisons: The Selection Problem," Technical Report No. 114, Department of Statistics, University of Minnesota, 1968.

[38] Stanley, R. P.; *Enumerative Combinatorics,* Wadsworth & Brooks, Monterey, California, 1986.

[39] Stockmeyer, P. K. and Yao, F. F.; "On the Optimality of Linear Merge," *SIAM Journal of Computing,* **9**, 85-90, 1980.

[40] Thanh, M. and Bui, T. D.; "An Improvement to the Binary Merge Algorithm," **BIT**, **22**, 454-461, 1982.

[41] Vuillemin, J.; "A Data Structure for Manipulating Priority Queues," *Communications of the ACM,* **21**, 309-315, 1978.

13 Appendix 1—Enumeration Results

Table 11 was created for each n by starting with $\mathcal{R}_n$ and systematically removing covers to create the next level of posets. Once these are all generated the next level is generated by systematically removing covers from each of the posets in the current level. This process is repeated until $\mathcal{U}_n$ is generated. Mohring [29] gives $|\mathcal{P}(n)|$ for $n \leq 10$. Combining results from [33, 21] and [38] (ex. 3, pg. 154) we see that

$$|\mathcal{P}(n)| \sim \frac{A}{n}\left(\frac{e}{n}\right)^n 2^{n^2/4+3n/2}\ , \quad \text{where } A \approx 0.80587793 (n \text{ even})$$

Observe that there are: $\binom{n-1}{0}$ posets with $\binom{n}{2}$ relations; $\binom{n-1}{1}$ posets with $\binom{n}{2}-1$ relations; $\binom{n-1}{2}$ posets with $\binom{n}{2}-2$ relations; $\binom{n-1}{3}+2n-5$ posets with $\binom{n}{2}-3$ relations; $\binom{n-1}{4}+2n^2-10n+11$ posets with $\binom{n}{2}-4$ relations. Dhar [10] presents some asymptotic results on the number of posets on n elements with a fixed fraction of $\binom{n}{2}$ relations. Note that, for $n \geq 10$, there are 1, 1, 3, 7, 19 and 47, posets with 0, 1, 2, 3, 4, and 5 relations respectively.

$r \setminus n$	1	2	3	4	5	6	7	8	9	10
0	1	1	1	1	1	1	1	1	1	1
1		1	1	1	1	1	1	1	1	1
2			2	3	3	3	3	3	3	3
3			1	4	6	7	7	7	7	7
4				3	10	16	18	19	19	19
5				3	10	25	38	44	46	47
6				1	12	36	74	107	124	130
7					9	43	113	208	287	329
8					6	46	167	381	636	841
9					4	44	209	619	1257	1946
10					1	35	243	915	2311	4251
11						28	249	1219	3830	8526
12						17	239	1506	5891	15891
13						10	204	1705	8294	27259
14						5	168	1792	10921	43572
15						1	123	1767	13363	64851
16							83	1621	15419	90614
17							54	1402	16687	119179
18							29	1136	17119	148255
19							15	874	16578	174838
20							6	629	15309	196135
21							1	434	13421	209729
22								274	11253	214283
23								166	8999	209692
24								94	6897	196824
25								46	5054	177576
26								21	3551	154148
27								7	2386	128998
28								1	1528	104101
29									939	81200
30									541	61145
31									300	44566
32									153	31401
33									69	21414
34									28	14096
35									8	8974
36									1	5492
37										3240
38										1836
39										986
40										506
41										237
42										99
43										36
44										9
45										1
	1	2	5	16	63	318	2045	16999	183231	2567284

$c \setminus n$	2	3	4	5	6	7	8	9	10	11	12	13	14	
1	1													1
2		2	1											3
3		1	5	2	1									9
4			4	14	8	2	1							29
5			1	15	41	24	8	2	1					92
6				11	$\geq$61	$\geq$124	$\geq$82	$\geq$28	8	2	1			$\geq$317
7				5						$\geq$28	8	2	1	
	1	3	11	47	255	1727	14954							

Figure 12: Number of Reduced Posets On n Elements Which Cost c Comparisons

14 Appendix 2—Lists of Posets

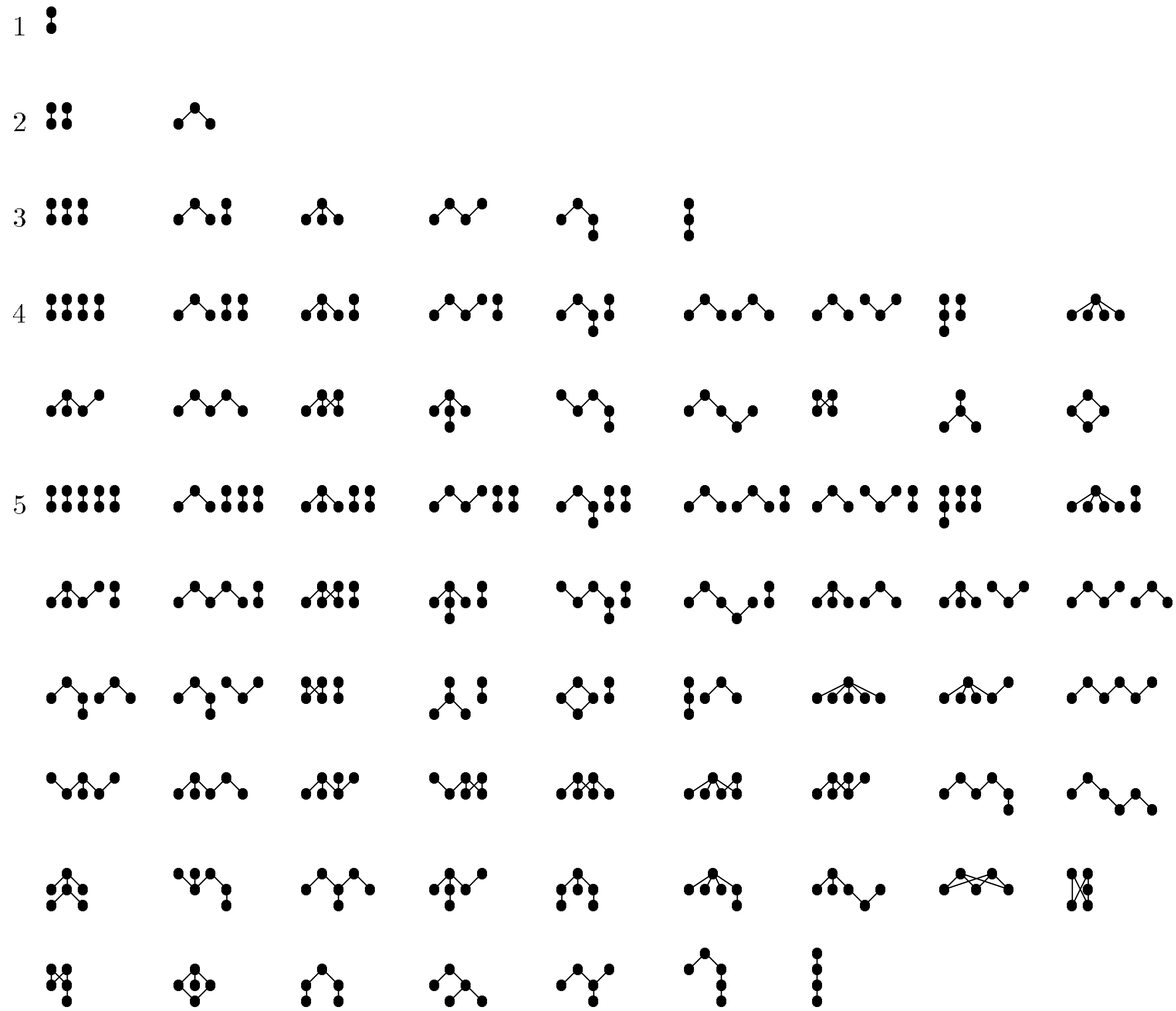

Figure 13: The 134 Reduced Posets Which Cost Less Than 6 Comparisons (Ignoring Duals)