Yale University
Department of Computer Science

Finding an Optimal Search Strategy for a Partial Order is NP-Complete

P. D. Laird L. Pitt

YALEU/DCS/TR-493

September 1986

This work was partially funded by the National Science Foundation, under grants MCS8002447 and DCS 8404226.

Abstract

The Partial Order Search (POS) problem is to find an optimal search strategy to locate an arbitrary element of a given partially ordered set S. Questions that may be asked during the search are of the form, "Is s the target element?" ($s \in S$); the response states the relationship between the queried element s and the target: "=", "<", ">", or "*incomparable*". An optimal strategy is one that requires the fewest questions in the worst case. POS is shown to be NP-Complete by exhibiting a polynomial-time reduction from the Exact 3-Cover problem.

Finding an Optimal Search Strategy for a Partial Order is NP-Complete

P. D. Laird* L. Pitt†

Introduction

The Partial-Order Search Problem (POS) is as follows:

> We are given a set S, a partial order $\leq$ on S, and and integer k. Is there a strategy which identifies an arbitrary element of S by asking at most k questions of the form "s?", with answers of the form ">", "<", "=", or "$\not\sim$" according to whether the target element is greater than, less than, equal to, or incomparable to s?
>
> We may assume (without loss of generality) that $1 \leq k \leq |S|$.

This problem generalizes the familiar binary-search problem for which the ordering is total. When the set being searched has n elements, the binary-search strategy identifies an arbitrary element in $\lfloor \log n \rfloor + 1$ queries; thus the answer to the corresponding decision problem is "yes" iff $k \geq \lfloor \log n \rfloor + 1$. We would like to know if there is an efficient algorithm to find an optimal search strategy for an arbitrary partial order. The NP-completeness result suggests that there is not.

The Exact 3-Cover Problem (X3C) is as follows:

> We are given a set X of $3q$ elements $(x_1, \ldots, x_{3q})$ and a set T of 3-element subsets $t_1, \ldots, t_r$ of X (also called "tests"). Is

This work was partially funded by the National Science Foundation, under grants MCS8002447 and DCS 8404226.

*Department of Computer Science, Yale University, New Haven, CT. 06520.

†Department of Computer Science, University of Illinois, Urbana, IL. 61801.

there a collection $\hat{t}_1, \ldots, \hat{t}_q$ of tests in T that partition X into subsets of size 3?

We may assume (without loss of generality) that $q \leq r \leq \binom{3q}{3}$.

X3C is known to be NP-complete ([GJ79]). In [HR76] X3C is reduced to the problem of constructing optimal binary decision trees, thereby proving the latter NP-complete. The proof below is also a reduction from X3C to decision trees for the partial-order search strategy, but the specific reduction and the proof technique (adversary) are quite different.

We model a search strategy for an instance $(S, \leq, k)$ of POS by a decision tree. The internal nodes are labeled by questions ("s?", where $s \in S$), and have two, three, or four outgoing edges labeled by possible responses ($<$, $>$, $=$, or $\not\sim$). The leaf nodes are labeled by elements of S. The height of the tree is the maximum number of questions needed to identify an element uniquely. There is no need to consider less than optimal trees, so we may assume that every element of S appears at exactly one leaf and that no useless questions occur internally. In particular, the size of a (correct) decision tree is $\mathcal{O}(|S|)$.

To see that POS is in NP, we envision a non-deterministic Turing Machine M which solves the POS decision problem on input of S, the relation $\leq$, and the integer k in binary. In time polynomial in $|S|$, M can write down a decision tree and check that

- no more than k questions occur on any path.
- every s in S occurs on some leaf.
- each path correctly and uniquely identifies the leaf.

We now show that POS is NP-complete by showing how to reduce an arbitrary instance of X3C to POS in polynomial time.

The Reduction

Let (X, T) be an instance of X3C. Construct POS$(S, \leq, q+3)$ as follows:

S consists of the $3q$ elements of X, plus the set $\{a, b, c\}$ of new elements (not in X), plus the set of tests T, plus another set T' of new elements distinct from T with $|T| = |T'|$, plus a set Y of additional elements, enough so that the cardinality

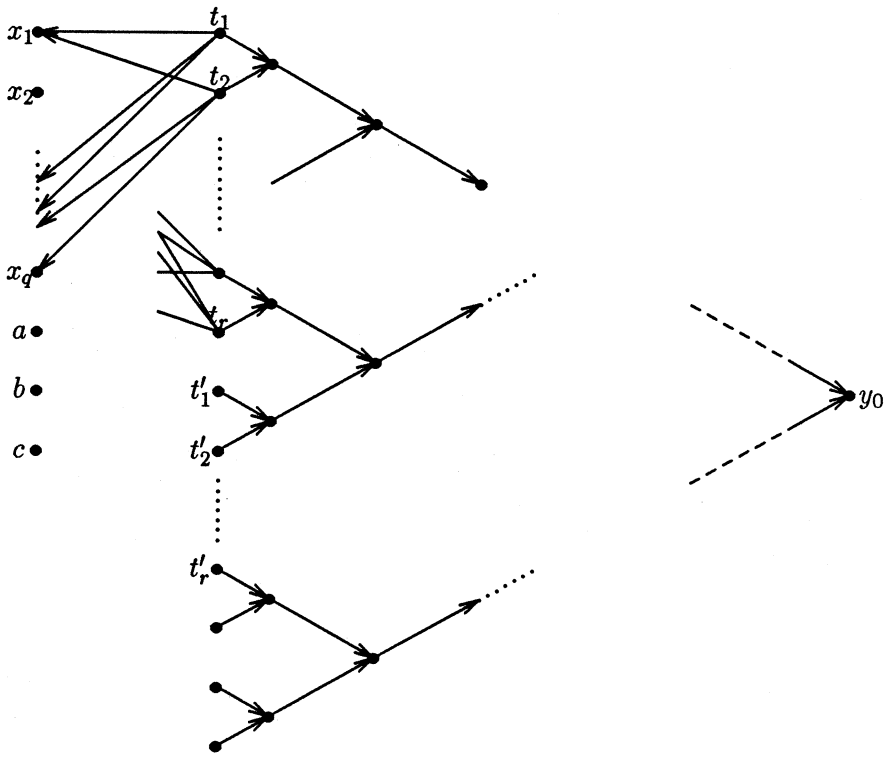

Figure 1: The partial order $\leq$.

$|T \cup T' \cup Y| = 2^m - 1$, where $m = \lceil \log |T| \rceil + 2$. With this number, a complete binary tree can be built from the elements of $T \cup T' \cup Y$, with T and T' at the leaves (see Figure 1). Note that $m \leq \lceil \log \binom{3q}{3} \rceil + 2 \in \mathcal{O}(\log q)$.

The partial order on S is constructed as follows:

- Each $t \in T$ is greater than the x's it covers.
- a, b, and c are incomparable to all other elements.
- The elements of T, T', and Y form a complete binary tree, with y_0 as the root (minimal element), and the elements of $T \cup T'$ (and possibly some Y's as well) as leaves (maximal elements).

Note that the x's are incomparable to all but the elements of T. Asking a

question about an element of T' or Y that returns an answer of "$\not<$" yields no information about the x's, and vice versa.

We claim that there is a POS decision tree of height $q+3$ iff there is an exact 3-cover, for all but finitely many q.

Suppose there is an exact 3-cover. Let $\hat{t}_1, \ldots, \hat{t}_q \in T$ be a 3-cover. A decision tree for identifying an arbitrary element is as follows:

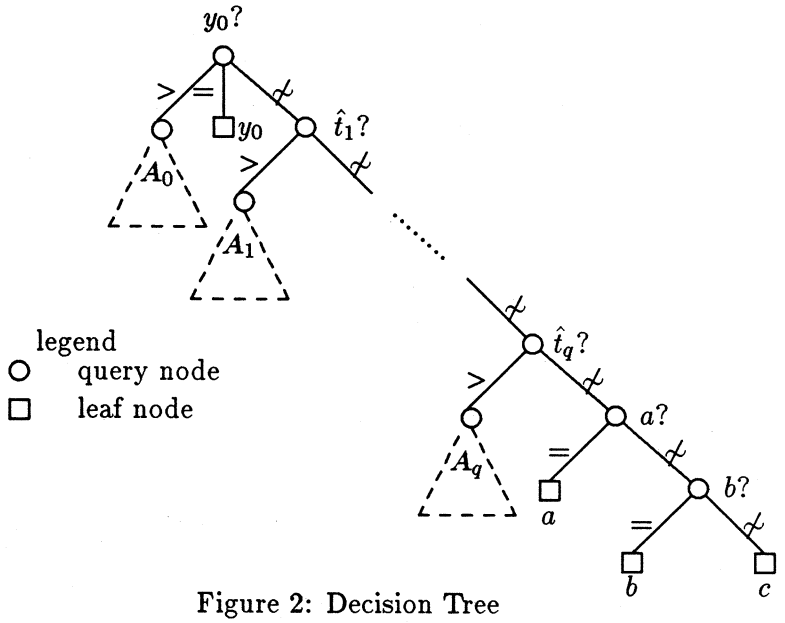

Figure 2: Decision Tree

Here, A_0 is a tree of height $m-1$ that identifies an element of $T \cup T' \cup Y$, and A_i are trees of height 2 that identify the three elements of X covered by $tHat_i$ (for $1 \leq i \leq q$).

Since $m \in \mathcal{O}(\log q)$, the height of A_0 is less than the height of the rest of the tree for almost all q. Thus the length of the longest path of the entire decision tree (and hence the maximum number of questions) is $q+3$.

The correctness of this decision tree is apparent. Note that it is not the only correct one of height $q+3$ (for example, the y_0 and $\hat{t}_1$ tests can be interchanged). But we claim that at least $q+4$ questions are required in the worst case if there is no exact 3-cover. The argument employs an adversary to show that $q+3$ questions are insufficient to identify an element uniquely.

Adversary: The adversary maintains two sets A and B. Initially, $A = X \cup \{a, b, c\}$ and $B = T \cup T' \cup Y$. In response to each question (determined by the decision tree), the adversary removes some elements from A and/or B and responds to the question. Specifically,

1. If the question is "x?" (where $x \in X \cup \{a, b, c\}$),
   - If A has any elements other than x, then remove x from A, remove from B all those elements $t \in B$ which dominate x, and reply "$\not<$".
   - Else if B has an element b such that $b \not< x$, then remove x from A and all elements of B that dominate x, and reply "$\not<$".
   - Else if $|B| \geq 1$, then reply ">" and remove x from A.
   - Else reply "=".
2. If the question is "$\hat{t}$?" (where $\hat{t}$ tests $\{x_1, x_2, x_3\}$),
   - If A contains any elements other than $\{x_1, x_2, x_3\}$, then reply "$\not<$", replace A by $A - \{x_1, x_2, x_3\}$, and remove from B all elements comparable to $\hat{t}$.
   - Else if B contains any elements incomparable to $\hat{t}$, then reply "$\not<$", replace A by $\emptyset$, and remove from B the element $\hat{t}$ and all elements comparable to it.
   - Else if $A \cup B = \{\hat{t}\}$, then reply "=".
   - Else reply "<" and remove $\hat{t}$ from B.
3. If the question is "y?" (where $y \notin X \cup T$),
   - If $A \neq \emptyset$, then reply "$\not<$" and remove from B all elements comparable to y.
   - Else if $B = \{y\}$, then reply "=".
   - Else reply "$\not<$", ">", or "<" according to whether the most elements of B are incomparable to, greater than, or less than y. Remove from B all elements inconsistent with the response.

We now observe the following:

- The adversary is correct: it removes from A and B all elements inconsistent with the response, and replies "=" when there is only one possible response. (Note that the correctness of the adversary depends on the assumption that no useless questions are asked.)

- No more than three elements of A are removed in response to one question. Thus q is an immediate lower bound on the number of questions required. This improves to $q+2$ as soon as we observe that $\{a,b,c\}$ are removed from A only by questions "a?", "b?", and "c?".

- At least one question about the elements in $T' \cup Y$ is required, for otherwise questions of types (1) and (2) will eliminate from B only those elements comparable to the t's in T, leaving the elements of T' (at least q of them) and some of the Y's in the tree. Thus $q+3$ is a lower bound on the number of questions required.

- Suppose there is no exact 3-cover. Then at least $q+1$ elements of T are required to cover the $3q$ elements of X. Hence, to remove all but one element of A requires at least $q+3$ questions:

  - If the target is in $\{a,b,c\}$, then $q+1$ questions are needed to eliminate the X's and two to rule out the other two elements.
  - If the target is in X, then three questions are needed to eliminate $\{a,b,c\}$, and at least q questions to eliminate all but one of the x's.

Together with the previous observation that at least one more question is required, this proves that $q+4$ questions are required if there is no exact 3-cover.

Thus the POS problem is solvable in $q+3$ questions iff there is an exact 3-cover, for sufficiently large q. This completes the proof.

Acknowledgment

The authors are grateful to Dana Angluin for suggesting the problem and discussing this work.

References

[GJ79] Garey, M. R. and D. S. Johnson.
Computers and Intractability,
W. H. Freeman and Co. (1979), San Francisco.

[HR76] Hyafil, L. and R. L. Rivest.
Constructing optimal binary decision trees is NP-complete.
Information Processing Letters 5: 15 - 17 (1976).