An Optimal Algorithm for Finding All Convex Subsets in Tournaments *

Marty J. Wolf David J. Haglin

Computer and Information Sciences Department
Mankato State University
Mankato, MN 56002

June 26, 1997

Abstract

A tournament is a complete directed graph. A convex subset is a vertex subset with the property that every two-path beginning and ending inside the convex subset is contained completely within the subset. This paper shows a relationship between convex subsets and transitive closures which leads to an optimal $O(n^3)$-time algorithm for finding all convex subsets in a tournament.

1 Introduction

A *tournament* is a directed graph on n vertices that is obtained by directing all of the edges in a complete undirected graph. A *convex subset* is a subset of the vertices such that any vertex not in the subset either dominates or is dominated by all of the vertices in the convex subset. Alternatively, a subset $C \subseteq V$ is convex if and only if every two path $u \to w \to v$, where $u, v \in C$ implies $w \in C$.

*This research was supported in part by a Mankato State University Academic Affairs Research Fund grant.

Although it is known that a single convex subset can be found in $O(n^3)$ time [2], our algorithm enumerates all of them (potentially $\Theta(n^2)$ [8]) in $O(n^3)$ time. We present an interesting series of reductions that leads to finding all convex subsets in a tournament in $O(n^3)$ time. Because each of the potential $\Theta(n^2)$ convex subsets has $\Theta(n)$ elements we have an algorithm that is optimal to within a constant in the worst case. This improves upon our previous algorithm which finds all convex subsets in $O(n^4)$ time [6]. We also reduce the processor requirement from our previous $O(\log^2 n)$ time, $O(n^4)$ processor parallel algorithm [6] to requiring $O(nM(n)/\log n)$ processors, where $M(n)$ is the sequential time required to multiply to boolean matrices. Currently, the best known bound for $M(n)$ is $O(n^{2.376})$.

Finally, our algorithm (in both the parallel and sequential versions) can be easily modified to provide solutions to variations of the convex subsets problem, including finding all of the convex subsets that contain a given vertex and finding all of the convex subsets that are supersets of a given set with two or more elements.

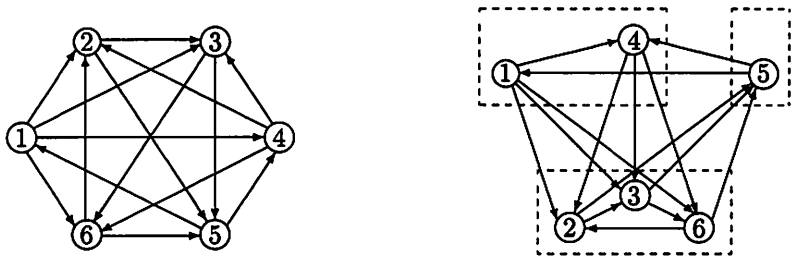
1.1 Background

Cunningham gives an $O(n^4)$ time algorithm [5], later improved by Bouchet to $O(n^3)$ [2], for finding a split of a digraph if one exists. Finding a split in a tournament leads to one convex subset. Astie-Vidal and Matteo were the first to give an algorithm that enumerates all of the convex subsets of a tournament [1]. However, their algorithm applies only to *regular* tournaments (tournaments where the outdegree of each vertex is equal to its in-degree). They claim that the number of "elementary operations" for their algorithm is $O(n^3)$. However, they include operations such as set intersection and subtraction as elementary operations. In this paper, we view those operations (as well as related operations) as taking $O(n)$ time. Their algorithm has an $O(n^4)$ time bound with this interpretation. Our algorithm computes all convex subsets of any tournament (not only the regular tournaments) in $O(n^3)$ time.

1.2 Definitions

Let $T = (V, E)$ be a tournament on n vertices. Assume that $V = \{1, \ldots, n\}$. For vertices $v, w \in V$, we say that v dominates w (denoted $v \to w$) if the edge between v and w is directed from v to w. A *convex subset* in T is a set $C \subseteq V$ such that for any $v \in V - C$ either v dominates every vertex in C or every vertex in C dominates v. This notion is illustrated in Figure 1. Since every subset of V of size 0, 1, and n is convex, these convex subsets are called *trivial*. This paper deals only with finding the nontrivial convex

subsets and, henceforth, we use the term *convex* to mean *nontrivial convex*. We call sets A, B, and C *strongly incomparable* if $A \not\subseteq B \cup C, B \not\subseteq A \cup C$, and $C \not\subseteq A \cup B$.



a) Tournament

b) Winners, Losers, and Convex Subset

Figure 1: Example Tournament with a Convex Subset

Let $S \subset V$ be a set of vertices within a tournament. S partitions the remaining vertices into three sets: winners, losers, and mediocre players. The winners are the vertices that dominate all of the vertices in S. The losers are the vertices dominated by all of the vertices in S. The remaining vertices, those that dominate some of the vertices in S and are dominated by others, are the mediocre players. Let $M(S) = \{ v \in V - S | \exists x, y \in S$ where $x \rightarrow v$ and $v \rightarrow y \}$ be the set of mediocre players relative to S.

Since a nontrivial convex subset contains at least two vertices, we use every two element subset of V as a starting point for computing all of the convex subsets. For $i, j \in V, i \neq j$, let $C_{i,j}(k)$ be defined inductively as follows:

$$\begin{aligned} C_{i,j}(0) &= \{i, j\} \\ C_{i,j}(k) &= C_{i,j}(k-1) \cup M(C_{i,j}(k-1)) \text{ for } k > 0 \end{aligned}$$

We can stop computing each $C_{i,j}(k)$ when $k = n - 1$ or when $C_{i,j}(k) = C_{i,j}(k-1)$. Let $C_{i,j}$ denote the resulting set. This definition immediately suggests an obvious $O(n^5)$ time inductive algorithm. Haglin and Wolf showed that this algorithm finds all convex subsets.

Lemma 1.1 [6] *Every convex subset in T is some $C_{i,j}$.*

Let $r_{i,j}(v)$ denote the *round* of induction when vertex v is brought into $C_{i,j}$. Thus if $v \in C_{i,j}(k)$ and $v \notin C_{i,j}(k-1)$ then $r_{i,j}(v) = k$. We define $r_{i,j}(i) = r_{i,j}(j) = 0$. If a vertex $v \notin C_{i,j}$ then $r_{i,j}(v) = \infty$.

Note that each vertex $v \in C_{i,j} - \{i, j\}$ has at least one "predecessor" vertex, w. We define a function $P_{i,j} : (C_{i,j} - \{i, j\}) \rightarrow V$ such that $P_{i,j}(v) = w$ where w is the lowest numbered vertex satisfying $r_{i,j}(w) = r_{i,j}(v) - 1$ and either $w \rightarrow v \rightarrow i$ or $i \rightarrow v \rightarrow w$.

2 Finding All Convex Subsets

Our new approach to finding all of the convex subsets in a tournament begins by focusing on finding all of the convex subsets that contain a particular vertex. To do this, we develop an auxiliary graph called an *inclusion graph* and then compute its transitive closure using standard algorithms. We show that this process meets our efficiency requirements by proving that there are $O(n)$ edges in the *transitive reduct*, the least graph with the same transitive closure as the strongly connected components graph of the inclusion graph. This computation can be invoked on each of the vertices of the tournament to generate all of the convex subsets of the tournament.

Let $T = (V, E)$ be an n-tournament. For $i \in V$, let $V_i = V - \{i\}$. Let $G_i = (V_i, E_i)$ be the inclusion graph where $E_i = \{(x, y) |$ either $(i, y), (y, x) \in E$ or $(x, y), (y, i) \in E\}$. Essentially an edge (x, y) in G_i indicates that if both x and i are in a convex subset then y must be in the convex subset as well, since y does not share the same relationship with both of the vertices. Essentially, y is mediocre with respect to x and i.

In Lemma 2.1 we show a relationship between the transitive closure of G_i and convex subsets of T. The following notation facilitates the presentation of this relationship. The *transitive closure* of a graph $G = (V, E)$, is the graph $Cl(G) = (V, E_C)$, where $(x, y) \in E_C$ if and only if there is a path from x to y in G. The *transitive reduct* of a graph $G = (V, E)$, is an acyclic graph $R = (V, E_R)$, where R is the least graph with $Cl(R) = Cl(G)$. Let $\Gamma_G(u) = \{ v | (u, v)$ is an edge in $G \}$. In the following lemma C represents all of those vertices reachable from j in G_i.

Lemma 2.1 *Let* $C = \Gamma_{Cl(G_i)}(j) \cup \{i\}$. *Then* $C = C_{i,j}$.

Proof: By the definition of G_i any vertex reachable from $j \in G_i$ must be in $C_{i,j}$. Hence $C \subseteq C_{i,j}$. Suppose $C_{i,j} - C$ is nonempty. Pick $k \in C_{i,j} - C$ such that $r_{i,j}(k)$ is minimized. Observe that $P_{i,j}(k) \in C$. By definition, edge $(P_{i,j}(k), k) \in E_i$. Using this argument inductively shows $C_{i,j} \subseteq C$. ∎

A straightforward computation of the transitive closure leads to an $O(n^4)$ implementation. We speed up the transitive closure computation of G_i by using a two-phase process. The first phase consists of computing the strongly connected component graph, S_i, of G_i. This can be easily done in $O(n^2)$ time with a modified version of depth first search. (See [3] for example.) Thus, each vertex of S_i represents potentially many vertices of G_i. Note that the transitive closure of S_i translates naturally into the transitive closure of G_i. In the second phase, we compute the transitive reduct, R_i, of S_i. By definition, $Cl(R_i) = Cl(S_i)$. The computation of R_i can be completed in $O(n \cdot e_{red})$ time, where e_{red} is the number of edges in R_i [4].

The following lemma is straightforward to prove and appears in Haglin and Wolf. In Theorem 2.3 we use it to prove that each vertex in R_i has indegree at most two. This, in turn, implies that $e_{red} \in O(n)$.

Lemma 2.2 [6] *For any vertex $i \in V$, there do not exist three strongly incomparable convex subsets X, Y, and Z such that $i \in X \cap Y \cap Z$.*

The proof of this fact relies on the restricted ways convex subsets can intersect. Namely, Varlet has shown that if the intersection of two convex subsets is nonempty then the union of those convex subsets is also convex [8].

Theorem 2.3 *Let T, G_i, S_i, and R_i be as defined above. Each vertex in R_i has indegree of at most two.*

Proof: First note that the vertex set of R_i is the same as the vertex set of S_i. Then assume the theorem does not hold. Let v be a vertex in R_i violating the claim. Let the indegree of v be $k \geq 3$. Label with $v_1, v_2, ..., v_k$ those vertices with an edge to v. Let C denote the convex subset of T defined by v (i.e., $C_{v,i}$) and let $C_1, C_2, ...C_k$ be the convex subsets of T defined by $v_1, v_2, \ldots, v_k$ (i.e., $C_{v_1,i}, \ldots C_{v_1,k}$). Because of the way G_i is built, $C \subset C_j$ for $1 \leq j \leq k$. Since R_i is a transitive reduct, no v_i is on a path from some v_j to v, $j \neq i$. Thus, $C_i \cap C_j = C$ for all $i \neq j$. Since any triple from $C_1, C_2, ...C_k$ is strongly incomparable Lemma 2.2 is violated, giving the desired result. ∎

Thus, the transitive closure of G_i can be computed in $O(n^2)$ time, leading to the following result.

Theorem 2.4 *All of the convex subsets of a given n-tournament $T = (V, E)$ can be computed in $O(n^3)$ time.*

Proof: By Theorem 2.3 each vertex in R_i has indegree at most two, and using the algorithm developed in [4], the transitive closure of S_i can be computed in $O(n^2)$ time. From the transitive closure of S_i the computation of the transitive closure of G_i in $O(n^2)$ time is straightforward. By Lemma 2.1 we can determine all of the convex subsets of the form $C_{i,j}$ in $O(n^2)$ time. Lemma 1.1 ensures that by repeating this process for all n vertices, we find all convex subsets in $O(n^3)$ time. ∎

Since there are tournaments with $\Omega(n^2)$ convex subsets, each with $\Omega(n)$ vertices [8] our algorithm is optimal to within a constant. For example, in the transitive tournament every pair of vertices defines a convex subset containing all the vertices between the pair in the transitive ordering of the vertices.

3 Finding Convex Subsets in Parallel

In this section we describe a straightforward parallel algorithm that finds all convex subsets in $O(\log^2 n)$ time using $O(nM(n))$ processors on a CREW-PRAM.

The obvious parallelization of the sequential algorithm given in Section 2 is to assign $O(M(n))$ processors to each vertex in the tournament. The algorithm is described in the following theorem.

Theorem 3.1 *Given a tournament $T = (V, E)$ we can compute all of its convex subsets in $O(\log^2 n)$ time using $O(nM(n)/\log n)$ CREW-PRAM processors.*

Proof: This algorithm proceeds similar to the serial algorithm. We describe the processing associated with a vertex $i \in V$. The inclusion graph G_i can be computed in $O(1)$ time using $O(n^2)$ processors, one processor for each vertex pair $x, y \in V - \{i\}$. We omit from the serial algorithm the step of finding the strongly connected component graph S_i, since the best known parallel algorithm for finding S_i has the same resource bounds as finding the transitive closure directly in G_i, namely $O(\log^2 n)$ time and $O(M(n)/\log n)$ processors [7]. ∎

4 Summary

We have proven properties about the relationships between convex subsets in tournaments and the transitive closure problem that provide a basis for sequential and parallel algorithms for finding convex subsets in tournaments. Because of these properties, our algorithm can be easily modified to an $O(n^2)$ time algorithm to determine all of the convex subsets that contain a given vertex. Our algorithm can also be modified to compute all of the convex subsets that contain any given subset in $O(n^2)$ time. Finally, note that the parallel version of our algorithm will solve both of the variations mentioned above in $O(\log^2 n)$ time using $O(M(n)/\log n)$ processors.

References

[1] A. Astie-Vidal and A. Matteo. "Non-simple tournaments: theoretical properties and a polynomial algorithm", *Proceedings of the Fifth Applied Algebra, Algebraic Algorithms and Error-Correcting Codes International Conference*, **5** (1989), pp. 1–15.

[2] A. Bouchet. "Digraph decompositions and Eulerian systems", *SIAM J. Alg. Disc. Meth.*, **8** (1987), pp. 323–337.

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. New York: McGraw-Hill, 1990.

[4] A. Goralčíková and V. Koubek. "A Reduct-and-Closure Algorithm for Graphs", in J. Bečvář, ed., *Proc. Conf. on Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science, **74** (1979), pp. 301–307.

[5] W. H. Cunningham. "Decomposition of directed graphs", *SIAM J. Alg. Disc. Meth.*, **3** (1982), pp. 214–228.

[6] D. J. Haglin and M. J. Wolf. "On Convex Subsets in Tournaments", *SIAM J. Disc. Math*, **9**, 1, (February 1996), pp. 63-70.

[7] D. S. Hirschberg. "Parallel Algorithms for the Transitive Closure and the Connected Components Problems", in *Proceedings of the Eighth Annual ACM Symposium on the Theory of Computing*, **8** (1976), pp. 55–57.

[8] J. C. Varlet. "Convexity in Tournaments", *Bulletin de la Société Royale des Sciences de Liège*, **45** (1976), pp. 570–586.