A NEW PROPERTY AND A FASTER ALGORITHM FOR BASEBALL ELIMINATION*

KEVIN D. WAYNE†

Abstract. In the baseball elimination problem, there is a league consisting of n teams. At some point during the season, team i has w_i wins and g_{ij} games left to play against team j. A team is eliminated if it cannot possibly finish the season in first place or tied for first place. The goal is to determine exactly which teams are eliminated. The problem is not as easy as many sports writers would have you believe, in part because the answer depends not only on the number of games won and left to play, but also on the schedule of remaining games. In the 1960's, Schwartz showed how to determine whether *one particular team* is eliminated using a maximum flow computation.

This paper indicates that the problem is not as difficult as many mathematicians would have you believe. For each team i, let g_i denote the number of games remaining. We prove that there exists a value W^* such that team i is eliminated if and only if $w_i + g_i < W^*$. Using this surprising fact, we can determine *all* eliminated teams in time proportional to a single maximum flow computation in a graph with n nodes; this improves upon the previous best known complexity bound by a factor of n.

Key words. network flow, combinatorial optimization

AMS subject classifications. 05C85, 68R10, 68Q25, 90B10, 90C27, 90C90

1. Introduction. In the *baseball elimination problem*, there is a league consisting of n teams which we denote by the set T. At some point during the season, each team has played some number of games. Team $i \in T$ has w_i wins, g_{ij} remaining games against team $j \in T$, and $g_i = \sum_{j \in T} g_{ij}$ total remaining games. Table 1.1 gives the input data for a sample league. The goal of a team is to finish the season with the most wins. We say that a team is *eliminated* if it cannot finish in first place, (i.e., with the most wins or tied for the most wins) for *any* possible outcome of the remaining games. We assume there are no ties (i.e., each game has a winner and loser), and no rain-outs (i.e., all remaining games are played). Without loss of generality, we assume all of the remaining games are against other teams in the same league. This classical problem was first popularized by Alan Hoffman in the 1960's as a nice application of optimization and network flow. The reader is referred to [2, 4] for textbook treatments of the problem.

Schwartz [15] proposed a method to determine whether a *single* team is eliminated using a maximum flow computation. Hoffman and Rivlin [11] generalized the result of [15], providing a characterization of when a team is eliminated from finishing in t-th place. Robinson [14] gave a linear programming based model that finds the maximum lead a team can have at the end of the season. Gusfield and Martel [10] and McCormick [12] determined the *elimination number*, i.e., the minimum number of remaining games a team must win in order to have any chance of finishing in first place. Their methods use different extensions of the parametric maximum flow techniques of Gallo, Grigoriadis, and Tarjan [5]. McCormick [12] also showed that it is NP-complete to determine whether a team is eliminated from finishing the season in t-th place or better. Adler, Erera, Hochbaum, and Olinick [1] proposed an integer programming formulation to determine which teams are eliminated from finishing

*A preliminary version of this paper has been accepted to the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms and will appear in the conference proceedings.

†Computer Science Department, Princeton University, Princeton, NJ 08544. Research supported while the author was at Cornell University by ONR through grant AASERT N00014-97-1-0681. Email: wayne@cs.princeton.edu.

team	wins	to play	schedule			
i	w_i	g_i	Atl	Phi	NY	Mon
Atlanta	83	8	–	1	6	1
Philadelphia	79	4	1	–	0	3
New York	78	7	6	0	–	1
Montreal	76	5	1	3	1	–

TABLE 1.1
Team standings and remaining schedule

the season in first place or as a wildcard playoff team and corresponding elimination numbers. Their Web site `www.riot.ieor.berkeley.edu/~baseball` maintains these statistics on-line for Major League Baseball.

In this paper we introduce a new structural property for the baseball elimination problem. Specifically, we order the teams according to their total number of wins possible (current wins + remaining games). We show that if a team is eliminated, then so are all teams below it in the ordering. For example, this implies that if two teams have the same number of wins and remaining games, then they are either both eliminated or both not eliminated, regardless of their remaining opponents. Using our new ordering and binary search, we can find *all* eliminated teams with $\log n$ maximum flow computations. Using the parametric maximum flow techniques of Gallo, Grigoriadis, and Tarjan [5], we show how to determine all eliminated teams in the same complexity as a single maximum flow computation. It is also straightforward to determine all of the elimination numbers from our computation.

We note that the new structural property was independently proved by Adler, Erera, Hochbaum, and Olinick [1] using linear programming techniques. They also describe how to compute all eliminated teams by solving a single linear program. Our proof is based on flows and cuts, and as a result, leads to a faster algorithm.

2. Preliminaries. In this section we review the necessary and sufficient conditions for a team to be eliminated. Also, we show how to determine whether a single team is eliminated using a maximum flow computation.

Let x_{ij} be a variable representing the number of games that team $i \in T$ wins among games remaining to be played against team $j \in T$. Team k is *not eliminated* if there is some assignment of nonnegative integer values $\{x_{ij} : i, j \in T\}$ such that:

$$\forall i, j \in T: \qquad x_{ij} + x_{ji} = g_{ij} = g_{ji} \tag{2.1}$$

$$\forall j \in T: \qquad w_k + \sum_{j \in T} x_{kj} \geq w_i + \sum_{j \in T} x_{ij}. \tag{2.2}$$

Equations (2.1) imply that all remaining games are played; inequalities (2.2) imply that no team finishes the season with more wins than team k.

Consider Table 1.1 above. Montreal is eliminated since it can finish with at most 81 wins, but Atlanta already has 83 wins. This is the simplest reason for elimination. However, there can be more complicated reasons. For example, Philadelphia is also eliminated. It can finish the season with at most 83 wins. But either Atlanta will win more than 83 games, or it will lose all 6 of its remaining games against New York, in which case New York will finish with at least 84 wins.

For any subset of teams $R \subseteq N$, let $w(R) = \sum_{i \in R} w_i$ denote the total number of games won already by teams in R, and let $g(R) = \sum_{\{i,j\} \subseteq R} g_{ij}$ denote the number of

games remaining to be played by teams both in R. We define $a(R) = \frac{w(R)+g(R)}{|R|}$, and note that $a(R)$ gives a lower bound on the average number of games (including games already won) that must be won by teams in R: the teams in R have already won $w(R)$ games, and some team in R must win each of the $g(R)$ games played between teams both in R.

LEMMA 2.1. *Let $i \in T$ and $R \subseteq T - \{i\}$. If $a(R) > w_i + g_i$ then team i is eliminated.*

Proof. If team i wins all of its remaining games, it will finish the season with $w_i + g_i$ wins. On average, the teams in R win at least $a(R) > w_i + g_i$ games. Thus (at least) one team in R will finish with more wins than team i. □

In this case, we say that R *eliminates* i, since it provides a certificate of elimination for team i. Surprisingly, if a team is eliminated, there is always such a simple certificate of elimination, as stated in Theorem 2.3 below. First, we review how to determine whether or not a single team is eliminated. The following theorem is due to Schwartz [15].

THEOREM 2.2. *Using a single s-t minimum cut computation, we can determine whether one particular team k is eliminated.*

Proof. Clearly, the best possible scenario for team k is if it wins all of its remaining games, in which case it will end up with $W := w_k + g_k$ wins. If $W < w_i$ for any $i \in T$, then $\{i\}$ trivially eliminates k.

Now, we check for more complicated reasons for elimination. We construct a bipartite network in which feasible integral flows correspond to outcomes of the remaining schedule. The following network flow formulation is due to Schwartz [15]; Gusfield and Martel [10] give an alternate construction. There are nodes corresponding to teams and to remaining games. Intuitively, each unit of flow in the network corresponds to a remaining game. As it flows through the network, it passes from a game node, say between teams i and j, then through one of the team nodes i or j, classifying this game as being won by that team.

The flow network for the baseball elimination problem is shown in Figure 2.1. Formally, let $N := T - \{k\}$ denote the set of teams other than team k. Let $P := \{\{i, j\} \subseteq N : g_{ij} > 0\}$ denote the set of pairs of teams (that don't involve team k) with remaining games to be played. Let $V := P \cup N \cup \{s, t\}$ denote the set of nodes in the network. For each $\{i, j\} \in P$ we include an arc $(s, \{i, j\})$ with capacity g_{ij}. For each team $i \in N$ we include an arc (i, t) with capacity $W - w_i$. Finally, for each $\{i, j\} \in P$ we include arcs $(\{i, j\}, i)$ and $(\{i, j\}, j)$ with infinite capacity. The flow on arc $(\{i, j\}, i)$ represents the total number of remaining games in which i beats j. The flow on arc (i, t) represents the total number of remaining games won by i.

It is easy to see that integral feasible flows of value $g(N)$ in the resulting network are in one-to-one correspondence with possible outcomes of the remaining games in which team i is not eliminated, i.e., they satisfy (2.1) and (2.2). It follows that we can determine whether i is eliminated with a single maximum integer flow (or minimum s-t cut) computation in the above bipartite network. □

The previous theorem says that we can determine whether any single team k is eliminated using a maximum flow computation in an appropriate bipartite network. In fact, if team k is eliminated, then the minimum s-t cut (in the same bipartite network) indicates a subset of teams that eliminates k. The next theorem is due to Hoffman and Rivlin [11]. We include its proof only for completeness.

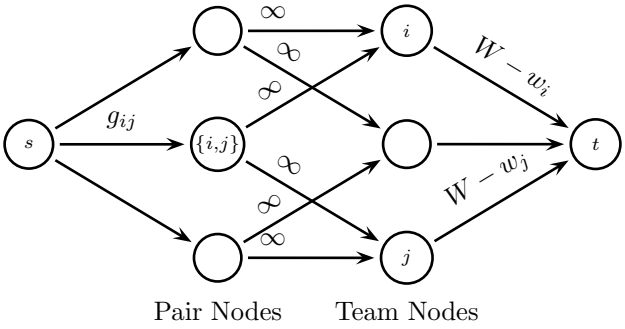

FIG. 2.1. *Flow network for baseball elimination*

THEOREM 2.3. *Suppose team $k \in T$ is eliminated. Then there exists $R \subseteq T - \{k\}$ that eliminates k. Moreover, we can find such a subset R with a single s-t minimum cut computation.*

Proof. Consider the maximum flow network described in Theorem 2.2. Let S denote the source side of a minimum s-t cut and let $R = N \cap S$ denote the team nodes on the source side of the cut. For example, for the four team league considered above with k = Philadelphia, it turns out that the minimum cut is $S = \{s, \{\text{Atl, NY}\}, \text{Atl, NY}\}$ and $R = \{\text{Atl, NY}\}$. We note that R eliminates Philadelphia since $a(R) = 167/2 > 83 = W = w_k + g_k$.

In general, if team k is eliminated, we show $R = N \cap S$ eliminates k. Since k is eliminated, the maximum flow in the network is less than $g(N)$. Hence, by the Max-Flow Min-Cut Theorem, the capacity of the minimum cut S is also less than $g(N)$; it is the sum of the capacities of some arcs leaving the source and some arcs entering the sink:

$$\begin{aligned}\sum_{\{i,j\}\in P} g_{ij} = g(N) &> cap(S) \\ &= \sum_{\{i,j\}\in P\setminus S} g_{ij} + \sum_{i\in R}(W - w_i) \\ &= \sum_{\{i,j\}\in P\setminus S} g_{ij} + W|R| - w(R). \qquad (2.3)\end{aligned}$$

Let $\{i, j\} \in P \cap S$. Then $i \in R$ and $j \in R$, since otherwise the cut would have infinite capacity. Thus

$$\sum_{\{i,j\}\in P\cap S} g_{ij} \le \sum_{\{i,j\}\subseteq R} g_{ij} = g(R). \qquad (2.4)$$

Combining (2.3) and (2.4) we obtain:

$$\begin{aligned}W|R| - w(R) &< \sum_{\{i,j\}\in P} g_{ij} - \sum_{\{i,j\}\in P\setminus S} g_{ij} \\ &= \sum_{\{i,j\}\in P\cap S} g_{ij} \\ &\le g(R).\end{aligned}$$

In other words, R eliminates team k. □

3. Problem Structure. We now provide a new structural property for the baseball elimination problem. We use the total order $i \preceq j$ to indicate $w_i+g_i \leq w_j+g_j$. The following theorem indicates that if a team is eliminated, then so are all lower ordered teams.

THEOREM 3.1. *Suppose team $k \in T$ is eliminated. If $i \preceq k$ then team i is also eliminated.*

Proof. Since k is eliminated, by Theorem 2.3 there exists $R \subseteq T - \{k\}$ that eliminates k. That is

$$a(R) > w_k + g_k \geq w_i + g_i.$$

If $i \notin R$ then R also eliminates i. Now suppose $i \in R$. Clearly $R \neq \{i\}$. Then, $R \setminus \{i\}$ eliminates i since

$$\begin{aligned} a(R \setminus \{j\}) &= \frac{g(R-\{i\}) + w(R-\{i\})}{|R|-1} \\ &\geq \frac{g(R) - g_i + w(R) - w_i}{|R|-1} \\ &> \frac{g(R) + w(R) - a(R)}{|R|-1} \\ &= a(R) \\ &> w_i + g_i. \end{aligned}$$

□

The following corollary was also derived independently by Adler, Erera, Hochbaum, and Olinick [1] using linear programming techniques instead of flows of cuts.

COROLLARY 3.2. *There exists a team $i^* \in T$ such that all teams $i \preceq i^*$ are eliminated and all teams $i \succ i^*$ are not eliminated.*

Proof. Choose i^* to be the eliminated team with the largest value of $w_i + g_i$. □

COROLLARY 3.3. *There exists a single subset of teams $R^* \subseteq T$ that eliminates every eliminated team.*

Proof. Choose R^* to be a nonempty subset of teams that maximizes $a(R)$. First we observe that if team k is eliminated then $k \notin R^*$. This follows from our choice of R^* because the proof of Theorem 3.1 would then imply $a(R^* - \{k\}) > a(R^*)$. By Theorem 2.3, if team k is eliminated then there exists a subset R such that $a(R) > w_k + g_k$. Now $a(R^*) \geq a(R)$ and $k \notin R^*$ so R^* also eliminates k. □

4. Determining All Eliminated Teams. In this section we show how to find all eliminated teams efficiently. It suffices to find the i^* guaranteed by Corollary 3.2. We can order the n teams according to their $w_i + g_i$ values and use binary search to find i^*. This requires $\log n$ minimum cut computations.

Now, we give an even faster method to find all eliminated teams. It suffices to find the R^* guaranteed by Corollary 3.3. We introduce an artificial team 0 which has no remaining games and a variable number of wins W. Let R^* be a nonempty subset that maximizes $a(R)$ and let $W^* = a(R^*)$. Note that team 0 is eliminated if and only if $W < W^*$. Also the elimination number for team i is easily seen to be $\lceil W^* \rceil - w_i$.

Now, we show how to find W^* and R^* efficiently. We construct a bipartite maximum flow network as in Figure 2.1, but now $k = 0$ and $N = T$. Also for each $i \in N$, the capacity of arc (i, t) is $W - w_i$ where W is a parameter. Note that all of the "parametric arcs" enter the sink and are increasing linear functions

of the parameter W. So, we are in a position to apply the parametric maximum flow technique of Gallo, Grigoriadis, and Tarjan [5] which computes all minimum cut values parametrically in terms of W in the same complexity as a single preflow-push maximum flow computation. Thus, we can compute W^* efficiently. The team nodes on the source side of the minimum cut gives R^*. The following theorem summarizes this discussion.

THEOREM 4.1. *Let $G = (V, E)$ be an undirected graph with arc weights g_{ij} and node weights w_i. We can find a nonempty subset of nodes R that maximizes $a(R) := \frac{g(R)+w(R)}{|R|}$ using a single monotone parametric maximum flow computation.*

If the undirected network G has n nodes and m arcs, then the bipartite network we construct has $\mathcal{O}(m)$ nodes and $\mathcal{O}(m)$ arcs. However, the smaller side of the bipartition has only $n_1 = \mathcal{O}(n)$ nodes. For a network with n nodes and m arcs, the Goldberg-Tarjan [9] preflow-push algorithm solves the maximum flow problem in $\mathcal{O}(mn \log(m/n^2))$ time, and the Goldberg-Rao [8] algorithm requires $\mathcal{O}(\min(n^{2/3}, m^{1/2})m \log(n^2/m) \log U)$ time if the capacities are integers between 1 and U. In our problem $U \leq \max_{i \in T}(w_i + g_i)$. Using the bipartite maximum flow techniques of Ahuja, Orlin, Stein, and Tarjan [3, 7], the running times remain valid for bipartite networks when the number of nodes n is replaced by the number of nodes on the smaller side of the bipartition n_1.

COROLLARY 4.2. *All eliminated teams can be determined in time proportional to one preflow-push maximum flow computation in a network with n nodes.*

The problem considered in Theorem 4.1 generalizes the *maximum density subgraph* problem considered by Goldberg [6]. In the maximum density subgraph problem, the goal is to find a subset of nodes that maximizes the ratio of the number of internal arcs to the number of nodes. This is the special case of our problem when the arc weights are uniform and the node weights are zero. Picard and Queyranne [13] and Gallo, Grigoriadis, and Tarjan [5] considered a different generalization of the maximum density subgraph problem that maximizes $g(R)/w(R)$.

Acknowledgments. The author is thankful to Éva Tardos for helpful discussions. The author also thanks the referees for providing useful comments that aided in the presentation of this paper.

REFERENCES

[1] I. Adler, A. L. Erera, D. S. Hochbaum, and E. V. Olinick. The RIOT baseball project: using the Internet to broadcast optimization-based playoff race statistics, 1998.
[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1993.
[3] R. K. Ahuja, J. B. Orlin, C. Stein, and R. E. Tarjan. Improved algorithms for bipartite network flow. *SIAM Journal on Computing*, 23-5:906–933, 1994.
[4] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley & Sons, Inc., New York, 1998.
[5] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18:30–55, 1989.
[6] A. V. Goldberg. Finding a maximum density subgraph. Technical Report UCB CSD 84/171, University of California, Berkeley, 1984.
[7] A. V. Goldberg. Personal communication, 1998.
[8] A. V. Goldberg and S. Rao. Length functions for flow computations. Technical Report 97-055, NEC Research Institute, 1997.
[9] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988.
[10] D. Gusfield and C. Martel. A fast algorithm for the generalized parametric minimum cut problem and applications. *Algorithmica*, 7:499–519, 1992.

[11] A. Hoffman and T. Rivlin. When is a team "mathematically" eliminated? In *Princeton Symposium on Mathematical Programming (1967)*, pages 391–401, 1970.

[12] S. T. McCormick. Fast algorithms for parametric scheduling come from extensions to parametric maximum flow. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pages 319–328, 1996.

[13] J. C. Picard and M. Queyranne. Selected applications of minimum cuts in networks. *Information Systems and Operations Research*, 20:394–422, 1982.

[14] L. W. Robinson. Baseball playoff eliminations: an application of linear programming. *Operations Research Letters*, 10:67–74, 1991.

[15] B. Schwartz. Possible winners in partially completed tournaments. *SIAM Review*, 8:302–308, 1966.