The Structure and Complexity of Sports Elimination Numbers [1]

Dan Gusfield and Chip Martel

Department of Computer Science Department
University of California, Davis.

Technical Report CSE - 99 - 1
January, 1999

[1]Research partially supported by grants DBI-9723346 and CCR 94-03651 from the National Science Foundation.

The Structure and Complexity of Sports Elimination Numbers

Dan Gusfield[†] Chip Martel[‡]

February 15, 1999

Abstract

Identifying the teams that are already eliminated from contention for first place of a sports league, is a classic problem that has been widely used to illustrate the application of linear programming and network flow. In the classic setting each game is played between two teams and the first place goes to the team with the greatest total wins. Recently, two papers [Way] and [AEHO] detailed a surprising structural fact in the classic setting: At any point in the season, there is a computable threshold W such that a team is eliminated (has no chance to win or tie for first place) if and only if it cannot win W or more games. They used this threshold to speed up the identification of eliminated teams. In both papers, the proofs of the existence the threshold are tied to the computational methods used to find it.

In this paper we show that thresholds exist for a wide range of elimination problems (greatly generalizing the classical setting), via a simpler proof not connected to any particular computational method; we resolve the more refined issue (in the classic setting) of which teams have a chance to be the *strict* winner of the most games; examine these issues in the context of multi-division leagues with playoffs and wildcards; and establish NP-hardness results for certain elimination questions.

[†]Dept. of Computer Science, Univ. of California, Davis. Research partially supported by NSF grant DBI-9723346. email: gusfield@cs.ucdavis.edu

[‡]Dept. of Computer Science, Univ. of California, Davis. Research partially supported by NSF grant CCR 94-03651. email: martel@cs.ucdavis.edu

1 Introduction

Consider a sports league with n teams, where specific pairwise games are played according to some schedule, and where some of these games have already been played. The classic question asks which teams are already *eliminated* from first place. That is, under every win/loss scenario for the remaining games, which teams will necessarily win fewer total games than some other team. This problem goes back more than thirty years to Alan Hoffman and T. Rivlin [HR70], and to B. Schwartz [Sch66]; it is widely used to illustrate linear programming and network flow[1].

For team i, $w(i)$ denotes the number of games already won, and $g(i)$ is the number of remaining games to be played. In the *classic version*, no game ends in a tie. Define the quantity $W(i) = w(i) + g(i)$ for each team i. Kevin Wayne [Way] recently showed the surprising fact (previously unsuspected by us, and not suggested in the classic literature) that there is a *threshold value* W^* such that any team i is eliminated if and only if $W(i) < W^*$. This fact is all the more surprising because the classic elimination problem is very well known and widely taught in diverse courses in computer science[2], mathematics and operations research. Using the existence of the threshold, Wayne also showed how to find W^* in time proportional to a single network flow computation in a graph with n nodes. Previously, the best approaches required a separate flow computation or linear programming computation for each team i, and no relationship between the results of the computations was observed. With Wayne's result, all the eliminated teams can now be identified as quickly as one could previously test whether a single specific team is eliminated.

Adler et al. [AEHO] independently established this elimination threshold using linear and integer programming formulations for elimination problems. They show that the threshold can be computed by solving one linear program with $\Theta(n^2)$ variables and constraints (one for each pair of teams which play at least one more game). They also give a nice overview of the errors which are often made by sprots writers in determining when a team is eliminated.

In addition to the classic question, three seemingly more difficult questions about the classic setting (single division, no ties) were also previously examined in the literature:

[1]See the website http://riot.ieor.berkeley.edu, where the eliminated teams are identified as the baseball and basketball seasons unfold.

[2]We have used it yearly for over ten years in graduate algorithms classes.

1. **Q1** If team i is uneliminated, what is the *minimum* number of games i can win and still at least tie for first place in some win/loss scenario?

2. **Q2** For team i, is there a scenario in which team i is the *undisputed* winner, winning strictly more games than any other team, and if so, what is the minimum number of wins for i to be an undisputed winner in at least one scenario?

3. **Q3** For team i, is there a scenario where i can at least tie for k'th position, i.e., where at most $k-1$ teams win strictly more games than does i?

Question Q1 was shown in [GM92] to be solvable in time proportional to a single network flow, provided the number of remaining games is $O(2^{n^2})$. McCormick eliminated the later requirement in [McC96]. However, the constructive method in [Way] implies that the answer to question Q1 is W^* for *every* team i where $W(i) \geq W^*$. Hence no additional computation is needed for Q1, once the classic elimination problem has been solved.

Question Q2 can be approached by a modification of the ideas in [GM92], in time proportional to a single network flow (for a single team).

Question Q3 was shown to be NP-complete in [McC96].

Until recently, and for all three questions Q1, Q2, and Q3, no threshold-value result was suspected that would connect the separate results obtained for each team.

New Results

In this paper we do seven things. We generalize the threshold-value result of [Way] and [AEHO], showing that it holds in a wide range of problem settings, using a pure "scenario argument" not tied to linear programming or network flow formulations of any problem setting; we show how the solution to question Q1 extends to these general settings; back to the classic setting, we solve question Q2 using an algorithm that runs as fast, or faster than a single additional network flow, and which simultaneously gives the answer to Q2 for *every* team; we extend these results to problems of multiple divisions with wild-card advancement to playoffs between divisions; we show that the problem of determining if a team is eliminated from the wild-card slot is

NP-complete; we note that in some settings, even when there is a threshold-value result, the problem of computing the threshold can be NP-hard; and we establish that computing the probability that a given team can come in first is NP-hard, when a probability for the outcome of each remaining game is given.

2 A general threshold-value result

In the general setting there is, as before, a history of games already played, and a schedule of remaining games. But now, each game results in one of a finite set of payoffs for the two teams. For example, in the classic setting, the winner gets one point and the loser gets zero; if ties are allowed one might allocate one point to each team when a tie occurs, and two points to the winning team, and none to the losing team, when a win occurs[3]. In a more general setting a winner might get three points and the loser gets zero, while a tie gives one point to each team[4]. Abstractly, each game has a set of possible outcomes, each of which is associated with a specific payoff pair (x, y), where one team gets x points and the other gets y points. The league leader is the team with the most total points. The problem is again to determine which teams have been eliminated, i.e., cannot strictly win or tie for the most points, under any scenario for the remaining games. In this setting, let $w(i)$ be the points team i has won so far, and let $g(i)$ be the maximum possible points that team i could get from it's remaining games, and define $W(i) = w(i) + g(i)$.

For the following result, we use a *monotinicity* restriction on the possible payoffs:

1. Whenever two teams x and y play, there is some maximum finite payoff x^* for x and y^* for y.

2. For any permitted payoff pair, (x_1, y_1) if $x_1 < x^*$ then there is also a payoff pair (x^*, y_2) with $y_2 \leq y_1$.

[3]It is "folklore" that this setting can be reduced to the classic setting, turning each game into two games between the same teams.

[4]Thanks to G. Brinkmann and Rob Irving for bringing to our attention that this is the current practice in European soccer. We recently learned that the elimination question in this scoring system is NP-Complete [BGHS].

3. For any permitted payoff pair, (x_1, y_1) if $y_1 < y^*$ then there is also a payoff pair (x_2, y^*) with $x_2 \leq x_1$.

Stated differently, we can always improve the payoff of one team to its best possible result without increasing the payoff to other team. Clearly, the three settings mentioned above ((1,0); (2,0), (1,1); and (3,0),(1,1)) obey this restriction.

Theorem 1. In any setting where the monotonicity restriction holds, there is a threshold-value W^* such that team i is eliminated if and only if $W(i) < W^*$.

Proof. Let S denote a selection of outcomes for the remaining games (a scenario for the remaining games), and let $W(S)$ denote the maximum number of points (from games already played and from the remaining games) won by any team under scenario S. Over all possible scenarios, define S^* as the scenario S where $W(S)$ is minimum. Define W^* to be $W(S^*)$.

Consider any team i. If $W(i) < W(S^*)$, then by the definition of $W(S^*)$, team i is eliminated in all scenarios. So assume $W(i) \geq W(S^*)$. If i is not the leader (either strict or tied) in S^*, then modify scenario S^* so that i receives a total of $g(i)$ points from its remaining games (i gets the best possible outcome from each game). Thus team i now has $W(i) \geq W(S^*)$ points. By the monotonicity restriction, no other team receives more points than it received under S^*, so under the modified scenario, team i is either the undisputed leader, or ties for the lead. $\square$

Note that the above proof has no connection to network flow, cuts or linear programming, showing that the general threshold phenomenon is not inherently related to these structures, or to any other structures used to compute the thresholds. That point is a fundamental contribution of this section. (However, in the classic setting, W^* and S^* can be efficiently computed using network flow, as established in [Way] or linear programming [AEHO]. In Section 4.2 we show that there are settings where a threshold result exists, but computing the threshold is NP- hard). Note also that Theorem 1 has easy extensions to games played between more than two teams at a time, provided the natural generalization of monotonicity holds.

Theorem 1 tells us which teams are eliminated. Question Q1 asks the finer question of computing the minimum number of points needed by team i to avoid elimination. Call this value $E(i)$.

In the classic setting, $E(i)$ is exactly W^* for any uneliminated team i, because the only payoff pair is $(1, 0)$ [Way] and [AEHO]. In general settings, $E(i)$ is not always W^* (e.g. when no possible combination of points allows team i to reach an exact total of W^* points). In addition, computing $E(i)$ may be NP-hard even if W^* and S^* are known, since it can be as hard as a subset sum problem.

2.1 Problem Q2: Becoming an Undisputed Winner

We now show how to efficiently solve problem Q2 for all teams simultaneously, in the classic problem setting, i.e., where $(1, 0)$ is the only payoff pair. As noted above, W^* can be computed in time proportional to a single network flow computation in a network with $O(n)$ nodes (thus in $O(n^3)$ time).

We want to compute for each team the minimum number of games that team must win in order to be the *undisputed* winner if possible (that is, to win strictly more games than any other team). Clearly any team i such that $W(i) > W^*$, can be the strict winner with that many wins (just take a scenario where team i ties for first with W^* wins and change any single loss by i to a win). Thus the only issue to resolve is which teams can be undisputed winners with exactly W^* wins. It is clearly not true that every uneliminated team can be the undisputed winner with only W^* wins. For example, consider teams A, B and C all with 99 wins, and team D with only 98 wins. Suppose the only remaining games are A against B and C against D. One of team A or B will be the undisputed winner with 100 wins if D wins, but even if C wins, it will be in a tie for the lead with A or B.

We now show how to find the set of teams which can strictly win with only W^* wins. Let scenario S^* be as defined in the proof of Theorem 1, and recall that S^* can be computed efficiently in the classic setting. By construction, under S^*, there is at least one team whose total wins (new plus old) is exactly W^*, and no team wins more than W^* total games. Let L^* be the set of teams under scenario S^* which each win a total of W^* games.

Given a scenario S, we say that team i_r is *reachable* in S from team i_1, if there is a chain $i_1, i_2, ..., i_r$ such that i_1 beats i_2 in at least one of the new games in S, i_2 beats i_3 in one of the new games, ... , and i_{r-1} beats i_r in one of the new games. Now we try to modify scenario S^* as follows to obtain a new scenario with fewer teams that win exactly W^* games:

Set S to S^* and L to L^*

```
WHILE there is a pair of teams $i$ and $j$ such that $i$ is in $L$, $j$ is reachable in $S$ from $i$, and $j$ wins less than $W^* - 1$ games in $S$,
    BEGIN
    Reverse the outcome of each game on a chain connecting $i$ and $j$; let $S$ again denote the resulting scenario, and let $L$ be set to $L - i$, which is again the teams that win exactly $W^*$ games in the new $S$.
    {Under this new $S$, $i$ wins a total of $W^* - 1$ games, $j$ wins at most $W^* - 1$ games, and the number of games any other team wins is unchanged.}
    END
```

The above process must terminate in at most n iterations because one team is removed from L in each iteration, and none are added. It is easy to implement this process to run as fast as a single network flow computation, and faster implementations seem plausible. An alternate, explicit network flow computation is detailed in appendix A.

Let l denote the size of L at the termination of the above process. Clearly, $l \geq 1$, because otherwise a scenario has been created where all teams win less than W^* games, a contradiction to the definition of W^*. After the above process terminates, define T to be the current L unioned with all the teams that are reachable (under the current S) from at least one team in L. By construction, each team in $T - L$ wins exactly $W^* - 1$ games in the current S.

Lemma 1. In any win/loss scenario where the winning team wins exactly W^* games in total, at least l teams of T must win W^* games in total.

Proof. Let W_T be $\Sigma_{i \in T} w(i)$, the given total number of old games won by teams in T; let $G(S)_T$ be the total number of new games won by teams in T, in scenario S. By the maximality of T, every new game won by a team in T is played between two teams in T. Hence in *any* scenario, the total number of new games that will be won by teams in T must be at least $G(S)_T$, and so the total number of games won by teams in T must be at least $W_T + G(S)_T = |T|(W^* - 1) + l$. It follows that in any scenario where W^* is the maximum total number of games won by any team, at least l teams in T must win W^* games. $\square$

The following is immediate.

Corollary 1. Since $l \geq 1$, no team outside of T can be the undisputed winner (in any scenario) with exactly W^* wins. Further, if $l > 1$, then no team can be the undisputed winner with exactly W^* wins.

Lemma 2. If $l = 1$, then for every team i_r in T, there is a scenario where i_r is the undisputed winner with exactly W^* wins.

Proof. If i_r is the single team in the current L, then S is the desired scenario. Otherwise, by the definition of T and S, i_r is reachable from the single team i with W^* wins. Reversing the wins along the chain from i to i_r results in a scenario where i_r has exactly W^* wins, and all other teams have less than W^* wins. □

In summary,

Theorem 2. A team i can be the undisputed winner with exactly W^* wins, if and only if $l = 1$ and team i is in T. If i cannot be the undisputed winner with W^* wins, it can be the undisputed winner with $W^* + 1$ wins if $W(i) > W^*$. Hence question Q2 can be answered for all teams simultaneously, as fast (or faster) than the time for a single network flow computation, once W^* and S^* are known.

In the case of question Q2, there is no single threshold that holds for all uneliminated teams (it is either W^* or $W^* + 1$), but the existence of the threshold W^* for question Q1, the main result of [Way], is critical for the efficient solution to question Q2.

3 Multiple divisions, playoffs and wild-cards

Often a sports league will partition the teams into multiple *divisions*. At the end of the season, the (single) team with the best record in each division makes the playoffs. In addition to the division winners, the playoffs may include a *wildcard* team, which is a team with the best record among those teams that do not win their division. In some sports leagues more than one wildcard team is advanced to the playoffs. For example, the current baseball format has three divisions in each of two leagues. In each league, the three division winners and a wild card team advance to the playoffs. Thus, if we want to determine which teams are eliminated from playoff contention or the minimum number of wins needed for a team to make the playoffs, we need to consider not only which teams can win their respective division, but which teams can be the wildcard team. While we can extend our results to more general settings, we restrict our discussion here to the classical setting where each game is either won or lost.

To formalize the problem, assume that the n teams are partitioned into k *Divisions* which we will denote by $D_1, \ldots, D_k$. At the end of the season the team in Division D_i with the most wins is the D_i Division's winner (we assume that if there is a tie for most wins this tie will be broken by a random draw or a playoff before the wildcard team is determined). Over all the divisions, the team with the most wins which is not a division winner is the *wildcard* winner (again we assume that if there is a tie, it is broken by a random draw or playoff). A team is eliminated from playoff contention only if it cannot win its division *and* it cannot be the wildcard team.

We will show in Section 4.1, that if the number of divisions is an input to the problem, determining if a given team can be the wildcard winner is NP-Complete. However, for a fixed number of divisions, k, the problem is solvable in polynomial time. And, in practice (for existing baseball and basketball leagues) integer programming can be used to efficiently determine the minimum number of games each team needs to win in order not to be eliminated from playoff contention [AEHO]. In this section we also show there is a threshold value of wins to avoid elimination from the playoffs which applies to all teams (across all divisions) who cannot win their division. That extends the threshold result in [AEHO] which was established for each division separately. In appendix B, we show how to solve this problem using network flow techniques, using $O(n^k)$ flows in the worst case, but far fewer in typical cases.

We first examine an elimination threshold for the winner of any single division. For any division D_d, there is a number W^*_d such that any team i in division D_d can be the division winner if and only if $W(i) \geq W^*_d$. The number W^*_d can vary between divisions. This was established in [AEHO] using a linear programming formulation. A more direct way to establish this result is again by the pure scenario argument: For scenario S (for all teams in the league), let $W_d(S)$ be the maximum number of games won by any team in division D_d; let W^*_d be the minimum $W_d(S)$ over all S, and let S^*_d be the scenario associated with W^*_d. Then there is a scenario where team i in D_d can win the most games (or tie) if $W(i) \geq W^*_d$ – simply change S^*_d by letting team i win all of its remaining games.

Another important point was established in [AEHO]: there is a scenario where some team in division D_d wins exactly W_d^* games, and yet every team in division D_d loses all of its remaining games played with teams outside of D_d. Again, modify scenario S_d^* by making each team in D_d lose to any team outside of D_d. Clearly, no team in D_d wins more games than before, but by definition of W_d^*, some team in D_d still wins W_d^* games. It follows that we can compute W_d^* for division D_d in isolation of the other divisions (as is done in [AEHO]), since games between teams in D_d and teams outside D_d can all be set to wins for the outside teams. Therefore, the computation of W_d^* reduces to the case of the classical setting where all teams are in the same division. Hence we need not discuss further how W_d^* is computed.

Now we turn attention to a wild-card threshold for teams that cannot win their division (in any scenario). We are concerned with which teams among the non-division contenders still have a chance to be the wildcard team, and for each such team, we want to know the fewest games it must win to still have a chance at the wildcard position. Over all possible scenarios for playing out the remaining games, let MW be the smallest number of wins for the wildcard team (there may be a tie at MW), and let SW be a scenario where the wild card team wins MW games.

Theorem 3. Any team i which is not a division winner in scenario SW can make the playoffs if $W(i) \geq MW$.

Proof. Consider any team i which is not a division winner in SW, and $W(i) \geq MW$. Change SW to SW' by increasing $i's$ wins until i has MW wins. At that point, the only teams which could possibly have more than MW wins in SW' are the k division winners in SW. Thus at most one team per division can beat i, so i must now be a division winner or the wildcard team (possibly in a tie). $\square$

Corollary 2. Any team i which cannot win its division is eliminated from the playoffs if and only if $W(i) < MW$.

Corollary 3. For a team i which can win its division, D_d, but is not the winner of its division in scenario SW, the minimum number of games i must win to make the playoffs is MIN$\{W_d^*, MW\}$.

It is interesting to note that the Corollary does not extend to the k teams that win their respective divisions in scenario SW. For example, suppose teams A and B are in division 1, teams C and D in division 2. A, B and D have 80 wins, C has 120 wins, and A has 10 games remaining versus each of B and D and these are the only (relevant) games. Teams B and D can tie for the wildcard with 80 wins if A wins all its games. However, team A cannot make the playoffs unless it wins at least 85 games (if A loses 16 games it gets to 84 and both B and D have at least 86). In fact, $W_1^* = 85 > 80 = MW$.

In Appendix B we describe how to compute MW and also the minimum number of wins needed by the k teams who are division winners in scenario SW.

4 NP-hard elimination questions

There are many generalizations of the classic elimination question that have no known efficient solution. In this section we examine three natural questions.

4.1 Multiple divisions and wild cards

Theorem 4. When there are multiple divisions and wildcard team(s), the problem of determining whether a given team t is eliminated from the playoffs is NP-complete in the classic setting.

Proof. The reduction is from the version of 3-SAT where each clause has two or three variables and each literal appears in at most two clauses [Pap94]. Given a formula F with k distinct variables (where each variable is assumed to appear both negated and unnegated in F) and c clauses, we create an instance of a $k+1$-division baseball problem. It is convenient to use a bipartite graph $H = (A, B)$, representing formula F, to describe this problem instance. The A side contains $2k$ nodes, one for each literal that appears in F; the B side contains c nodes, one for each clause in F. There is an edge between node i in A and node j in B if and only if literal i appears in clause j (so each node in B has two or three incident edges). The corresponding baseball schedule is created by letting each node in H represent a baseball team, and each edge in H represent a game yet to be played. In this problem instance, we assume that every team in A has won z games so far, and every team q in B has won $z - d(q)$ games so far, where $d(q)$ is the degree of q in H.

The $k+1$ divisions are next described. For every variable x that appears in F, the two teams associated with literals x and $\bar{x}$ are in a single division, and no other teams represented in H are in that division. All the nodes in B are in a single division, denoted DB. Team t is also assumed be in division DB, but it has played all of its games, winning a total of z games. Since t has no games to play, it is not represented in H. We further assume that there is another team in DB that has played all its games and has already clinched the title for DB. There may be additional teams not represented in H (in order to make the divisions bigger and more realistic), but they are all assumed to be eliminated from their division title and from contention for the wild-card slot. Hence only the teams in A are competing for their respective division titles, and only the teams in H together with t are competing for the wild-card slot.

Now assume there is a satisfying assignment F_s to F; we will construct a scenario where team t is at least tied for the wild card slot i.e., at most one team from each division wins more games than t.

For every variable x in F, we let the team associated with literal x (respectively $\bar{x}$) win all of its remaining games if and only if variable x is set true (respectively false) in F_s. Hence for every literal pair $(x, \bar{x})$, one of the associated teams wins all of its remaining games, and the other looses all of its remaining games. Since the assumed truth assignment satisfies F, at least one literal in every clause is set true. Hence every team in B loses at least one of its remaining games, and no team in B wins more than z games. Thus exactly one team in each division wins more games than does team t, and hence team t is still at least tied for the wild-card slot.

Conversely, consider a scenario where t is at least tied for the wild-card slot. That means that no division has two or more teams that strictly win more games than does t. It follows that no team in B can win all of its remaining games, for then t would place behind two teams in DB. Similarly, it can't happen that both teams associated with a variable x can win one (or more) remaining games, for then both of these teams would have more total wins than does t. So we set variable x true (respectively false) if the team associated with literal x (respectively $\bar{x}$) wins one or more of its remaining games. Any unset variable can be set arbitrarily. This assignment satisfies F. □

Note that in our reduction every node in H can have degree at most 3, and thus every team has at most 3 games left to play. Thus, this problem is hard to decide even when the schedule of games is almost completed.

4.2 Thresholds are not always easy to compute

As established above, threshold results occur in a large variety of problem settings (and we will establish another one here). However, the existence of a threshold does not necessarily imply that it is easy to compute the threshold. In a sense, we have already established that point in the case of multiple divisions. However, to make the point clearer, we examine a case where there is a single, simple threshold, but the problem of computing it is NP-hard[5].

In the classic setting, consider the question of whether there is a scenario in which a given team i can come in (possibly tied for) k'th place or better, i.e., where there are no more than $k-1$ teams who win strictly more games than does i (k is a variable, given as input – the problem is solvable in polynomial time for any fixed k). This is a generalization of the classic elimination question, where $k = 1$.

It is easy to establish that there is a threshold result for this problem. For each scenario S, define $W^k(S)$ as the total points that the k'th ranked team obtains in S; define W^k as the minimum of $W^k(S)$ over all scenarios, and let S^k be the scenario that gives W^k. If $W(i) \geq W^k$, then there is a scenario in which team i comes in k'th or better –simply modify S^k so that team i wins all of its remaining games. Hence a team i can come in k'th or better if and only if $W(i) \geq W^k$. However, McCormick [McC96] has established that the problem of determining if a given team can come in k'th or better is NP complete. It immediately follows that it is NP-hard to compute the threshold W^k.

4.3 Probability of elimination

We next examine the elimination question when each remaining game (i, j) is associated with a probability $p_{i,j}$ that team i will win the game, and

[5]We recently learned that the elimination problem for the European football scoring scheme, mentioned in Section 2, has also been shown to be NP-complete [BGHS], even though (as shown in Section 2) there is a single elimination threshold for that scoring scheme.

probability $p_{j,i} = 1 - p_{i,j}$ that team j will win the game [6].

We first show that the problem of computing the number of ways (scenarios) that a given team t can avoid elimination is $\#P$-complete. The reduction is from the problem of computing the number of perfect matchings in a bipartite graph $H = (A, B)$, where $|A| = |B|$. As before, we interpret each node in H as a team and each edge as a remaining game. Each node in A is assumed to have won $z - 1$ games, each node q in B is assumed to have won $z + 1 - d(q)$ games, where $d(q)$ is the degree of q, and team t is assumed to have won z games, with no more games to play. Clearly, t is eliminated if any team in B wins all of its games, or any team in A wins more than one game. So in a scenario where t is uneliminated, every team in B looses at least one game and every team in A wins at most one game. Since $|A| = |B|$, every team in A must win exactly one game, and each team in B must lose exactly one game. Representing the outcome of a game by a directed edge from the loser to the winner, it is clear that the directed edges from B to A specify a perfect matching in H. Further, each such scenario leads to a different perfect matching. Conversely, if we interpret a perfect matching in H as a set of games where the B-team loses to the A-team (and the B-teams win all the other games), then team t is uneliminated in this scenario. Further, each perfect matching leads to a different scenario. Hence the number of perfect matchings in H equals the number of scenarios where t is not eliminated.

Summarizing, we have proved that

Theorem 5. The problem of computing the number of scenarios where team t is uneliminated is $\#P$-complete.

Turning to probabilities, if we set $p_{i,j} = 1/2$ for each remaining game and there are G total games remaining, then the probability that t will be uneliminated at the end of the season is $(1/2)^G \times$ (the number of scenarios where t is uneliminated). Hence

Theorem 6. The problem of computing the probability that team t will be eliminated is NP-hard.

[6]The results in this section were developed by Greg Sullivan and Dan Gusfield in 1983 and forgotten for many years. We thank Greg for allowing the inclusion of these results in the present paper.

5 Appendix A: Computing Undisputed Winners

To explicitly implement the algorithm in Section 2.1 as a single network flow computation, consider a "supply-demand" network with an edge for each remaining game (p, q) directed from p to q if p beats q under S^*. Each team in L^* is given one unit of supply, and each team not in L^* has demand equal to $W^* - 1$ minus the total number of games it wins under S^*, and each edge has capacity one. Then compute an integral maximum flow from the supply nodes to the demand nodes (obeying the supply-demand constraints and the edge capacities). Since the edge capacities are one, the edges with flow value one define a set of edge disjoint paths from supply to demand nodes. These paths define chains that could have been found by the algorithm in Section 2.1, and hence reversing the wins on these edges gives the needed scenario S.

For more efficiency, if two teams have more than one game to play against each other, we can combine the edges for all those games and set the capacity of the combined edge (p, q) equal to the number of remaining games where p beats q. Therefore, the time to find S is bounded by the time for a single network flow computation on a graph with n nodes.

6 Appendix B: Computing the Wildcard Threshold for Multiple Divisions

Recall there are n teams partitioned into k *Divisions* which we will denote by $D_1, \ldots, D_k$. At the end of the season the team in Division D_i with the most wins is that Division's winner (we assume that if there is a tie for most wins this tie will be broken by a random draw or a playoff before the wildcard team is determined). The team with the most wins which is not a division winner is the *wildcard* winner (again we assume that if there is a tie it is broken by a random draw or playoff). A team is eliminated from playoff contention only if it cannot win its division and it cannot be the wildcard team.

Consider a point partway through the season. For each division we can determine which teams have a chance to win that division using the techniques developed for a single division. There will be a single number W_d^* for each division D_d such that those teams who can win W_d^* games have a chance to win D_d, and this can be computed in the time for one flow. In the computation of D_d, games with teams not in D_d are ignored, as justified earlier.

For each division D_d, let $T(d)$ be the set of teams in that division which have not been eliminated as division winner, i.e., where $W(i) \geq W_d^*$. We compute MW by considering each combination of possible division winners, one from each set $T(d)$. For any such choice of division winners, we create a flow network WILD. To describe WILD, we use the convention that team d denotes the division winner for division D_d for $d = 1, ..., k$. It is easy to modify WILD for any other fixed set of division winners.

Let B be a bipartite graph containing two nodes, j and j', for each team $j > k$ (that is for each non- division winner j), one node for each side of B. For every ordered pair of nodes (i, j') in B where $i < j$, connect the i node on the left side of B to the j' node on the right side of B, and give it capacity equal to the number of remaining games between teams i and j. Also, create a directed edge from each i to i', with infinite capacity. Then create a source s connected to each node i on the left of B, and give edge (s, i) capacity equal to the number of remaining games i plays with teams numbered greater than i. Similarly connect each node j' on the right of B to a terminal t, and give that edge capacity $W - w(j)$, where W is a variable. The resulting network is called WILD. For a fixed value of W, any integral flow in WILD that saturates all the edges out of s gives a feasible scenario S for the allocation of wins in the remaining games between non-division winners. Thus, no team $j > k$ ends with more than W wins in S.

In WILD we have assumed that the division winners win all their games versus non-division winners (so those teams aren't included in the network). We use WILD to find the smallest W such that all other teams win at most W games when that happens. Recall that over all possible scenarios for playing out the remaining games, we let MW be the smallest number of wins for the resulting wildcard team (there may be a tie at MW), and we let SW be a scenario where the wild card team wins MW games.

Lemma 3. If we create the network WILD for the division winners in SW, then there will be a flow that saturates all arcs out of s when W is set to MW in WILD.

Proof. Create a flow in WILD according to SW (i.e., for each team i, if i loses r games to $j > i$, set the flow in the arc (i, j') to r, and if i wins w games versus teams lower than i, set the flow in (i, i') to w). No team i will have flow greater than $MW - w(i)$ entering node i' (by the convention that teams 1 through k win their respective divisions in SW and the wildcard has MW wins), and this will saturate all arcs out of s since all games have a winner in SW. $\square$

Lemma 4. Let WILD be the networks associated with any combination of division winners, and let W^* be the smallest value of W that allows all arcs out of s to be saturated in WILD. Then there is a scenario where some team becomes the wildcard with W^* or fewer wins.

Proof. The flow in WILD that saturates all arcs out of s with W set to W^* corresponds to a scenario S^*, by converting the flow in the natural way (and having the designated "division winners" win all their games versus other teams, and assigning wins to games between "division winners" arbitrarily). Note that teams 1 through k might not all be the division winners in S^* since we did not require them to win more than W^* games. Let A be the wildcard team in S^* (if there is a tie pick any of the tied teams). Unless A is one of teams 1 through k it can have at most W^* wins, by our construction of WILD and we are done. If A is one of teams 1 through k then some team with index greater than k must win $A's$ division. However, that new division winner has at most W^* wins, so for A to lose its division it must have W^* or fewer wins. Thus in all cases A has at most W^* wins. $\square$

Theorem 7. Let W_{min} be the minimum value of W that allows all arcs out of s to be saturated, when we construct WILD for all possible combinations of division winners. Then $W_{min} = MW$, the smallest number of wins possible for any wildcard winner.

By Lemma 3 when we set W to MW in WILD, with the division winners in SW selected, we will saturate all arcs out of s, thus $W_{min} \leq MW$.

Now suppose for contradiction that $W_{min} < MW$. Then there is some flow in WILD (for some combination of possible division winners) that saturates all arcs out of s with $W^* < MW$. This saturating flow corresponds to a scenario S^*. By Lemma 4 some team is the wildcard in S^* with W^* or fewer wins. However, this means the wildcard team has fewer than MW wins in S^* contradicting our definition of MW. Thus W_{min} cannot be smaller than MW, proving that it is equal to MW.

6.1 Minimum Wins for Special Division Winners

The division winners in SW can make the playoffs by winning W_d^* games (the minimum to win their division). However, under some other scenario, they might be able to be the wild card team with fewer wins. Consider say team A which is the winner of division D_1 in SW. If A can be the wild card with some number of wins, say W_A, with $W_A < W_1^*$ then there is clearly some best scenario S_A where A makes the playoffs with the fewest wins. In the computation described above we consider all combinations of possible division winners, so we also construct network WILD for the division winners in scenario S_A. Using the same reasoning as above, we must get the value W_A when we do this computation. Similarly, if we ever got a value smaller than W_A for a network which did not include A as a division winner, than A could reach the playoffs with fewer than W_A wins contradicting our definition of W_A.

Thus let W_A be the minimum value we get from all networks WILD where A is not selected as a division winner. Then the minimum wins for A to make the playoffs is $\text{MIN}\{W_A, MW_1\}$.

6.2 Complexity

Each network WILD (for a combination of division winners) is a linear monotone parametric flow network, so we can find the minimum value of W saturating all arcs out of s in the time for one flow [GGT89]. In the worst case there could be $(n/k)^k$ combinations, but by first computing possible division winners within each division (at a cost of the time for one flow/division), we will usually be able to greatly reduce the number of combinations.

This gives us not only MW but also the minimum wins for each division winner in SW.

Finally, once we have computed the threshold values to at least tie for a playoff spot for each team we can also determine the minimum number of wins to make the playoffs without a tie (thus to be the undisputed division winner or wildcard team). For each threshold T, there is an associated scenario where some team makes the playoffs with T wins. We can determine which teams need T versus $T+1$ wins to avoid a tie in a manner analogous to that used for the case of a single division with one winner.

References

[AEHO] I. Adler, A. Erera, D. Hochbaum, and E. Olinich. Baseball, optimization and the world wide web. unpublished manuscript 1998.

[BGHS] T. Burnholt, A. Gullich, T. Hofmeister, and N. Schmitt. Football elimination is hard to decide under the 3-point rule. unpublished manuscript, 1999.

[GGT89] G. Gallo, M. Grigoriadis, and R.E. Tarjan. A fast parametric network flow algorithm. *SIAM Journal on Computing*, 18:30–55, 1989.

[GM92] D. Gusfield and C. Martel. A fast algorithm for the generalized parametric minimum cut problem and applications. *Algorithmica*, 7:499–519, 1992.

[HR70] A. Hoffman and J. Rivlin. When is a team "mathematically" eliminated? In H.W. Kuhn, editor, *Princeton symposium on math programming (1967)*, pages 391–401. Princeton universtiy press, 1970.

[McC96] T. McCormick. Fast algorithms for parametric scheduling come from extensions to parametric maximum flow. *Proceedings of the 28th Annual ACM Symposium of Theory of Computing*, pages 394–422, 1996.

[Pap94] C. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[Sch66] B. Schwartz. Possible winners in partially completed tournaments. *SIAM Review*, 8:302–308, 1966.

[Way] K. Wayne. A new property and a faster algorithm for baseball elimination. ACM/SIAM Symposium on Discrete Algorithms, January 1999.