International Journal of Foundations of Computer Science

SOCCER IS HARDER THAN FOOTBALL*

JAN CHRISTENSEN
Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark
janc89@gmail.com

ANDERS NICOLAI KNUDSEN
Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark
andersnk@gmail.com

KIM S. LARSEN
Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark
kslarsen@imada.sdu.dk

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

It is known that deciding whether or not a team in a soccer tournament in progress can still win or, more generally, can obtain a certain position is NP-complete. We show that deciding whether or not a team is guaranteed a certain minimum position is coNP-complete. We also show that deciding with regards to goal difference, the standard tie-breaker for teams having the same number of points, whether or not a team can reach a certain position is NP-complete.

Keywords: Analysis of algorithms; completeness results; tournaments.

1. Introduction

We use American terminology. Thus "football" refers to American Football as organized under NFL, and "soccer" refers to the game originating in Europe, organized under FIFA. In references, "football" may refer to either game.

We are interested in soccer tournaments under the $(3, 1, 0)$-rule, i.e., a team receives 3 points for a win, each team receives 1 point for a tie, and a losing team does not receive any points. In a tournament, each team has to play against any other team; in most leagues twice during the tournament. At any point in time

*The work of the third author was partially supported by the Danish Council for Independent Research, Natural Sciences, and the Villum Foundation.

during the tournament, there is a current standing and a schedule (or list) of games still to be played. A team's position in the standing is decided primarily by the total number of points accumulated. In case of ties with regards to the total number of points, it is standard to consider the goal difference, where the goal difference for a team is the difference between the total number of goals scored by the team minus the total number of goals scored against the team.

The earliest results for placements problems were developed for tournaments using the $(1, 0)$-rule, i.e., each game has a winner. This rule applies to baseball and American football, for instance. For this rule, it was shown by Schwartz [9] that deciding if a team can still win is in P and, in fact, deciding for any constant K if a team can reach position K is in P. More precisely, one can find all teams that can still win via a single maximum flow computation, as developed by Wayne [10], or a single linear programming formulation, as developed by Adler, Erera, Hochbaum, and Olinick [1]. McCormick showed that only if K is part of the input does the problem become NP-complete [7].

Under the $(3, 1, 0)$-rule it is harder. Bernholt, Gülich, Hofmeister, and Schmitt showed that even deciding if a team can still win is NP-complete [3]. The exact characterization of this question with regards to different rules was given by Kern and Paulusma [5], where it was shown that, excluding trivial cases of getting the same number of points for two different outcomes, deciding if a team can win is possible in polynomial time if and only if there is a fixed number of points per game which are shared, e.g., it is decidable in polynomial time for the older $(2, 1, 0)$-rule for soccer. Games with more than three outcomes was also investigated by Kern and Paulusma [6]. For $(p, 1, 0)$-rules, Pálvölgyi showed that deciding, given a standing, if this can be the result of a standard tournament is NP-complete [8]. Finally, winner problems were considered by Aziz, Harrenstein, Brill, Lang, Fischer, and Seedig [2] for a variety of alternative tournament forms.

Our interest is in the $(3, 1, 0)$-rule. Other authors have primarily focused on determining possible winners of a tournament. We are interested in the question of determining a minimum guaranteed placement. Phrased as a decision problem, given some K, we are interested in whether a given team can be guaranteed a placement of at least K. Note that this problem is *not* the complement of the problem of determining if it is possible for a given team to reach a placement of at least K. Thus, the complexity status the problem of determining a minimum guaranteed placement has not been known. Furthermore, other authors have only focused on points. We are also interested settling the placement problem with regards to the standard tie-breaking rule in case of the same number of points. These considerations become natural if one wants to equip the table of the current standing with an interval for each team, listing exactly the positions where this team may end up.

To be precise, we settle the complexity status of the two problems below.

For the first problem, the outcome of a game is a pair, which is either $(3, 0)$, $(1, 1)$, or $(0, 3)$. As is standard in the literature, we make an extreme decision regarding

the selected team in case of ties in the final standing. Here, we decide that if the selected team has the same number of points as some other teams, it is placed worst in that group of teams.

> **Guaranteed Point Placement**
> *Instance*: Current standing in a $(3, 1, 0)$-rule tournament, schedule of remaining games, team t, positive integer K.
> *Question*: Is it the case that any assignment of outcomes to the remaining games will place team t at position at least K in the final standing?

For the next problem, an outcome of a game is a pair representing the number of goals scored by each team. We will refer to this as the *goal* outcome. This of course also implies the number of points awarded to each team. In the following formulation, we are discussing changing the number of goals scored without changing the number of points awarded. Thus, if we have decided that a team should win over some other team in a solution to the problem above, we must respect that decision, but can, for instance, increase the number of goals scored by the winning team, with the purpose of changing the ordering in the final standing with respect to the goal difference tie-breaker.

We assume that if the selected team ties with other teams both with regards to the total number of points and goal difference, it places best among those teams. This is the opposite of the decision we made for the previous problem. Since showing NP-hardness is a negative result, in the sense that it does not form a basis for a programming solution, the decision is not important in an application context. Thus, we have simple made the decision in each case that enabled us present the most readable proofs.

> **Possible Goal Difference Placement**
> *Instance*: Current standing in a $(3, 1, 0)$-rule tournament, assignment A of goal outcomes to the remaining games, team t, positive integer K.
> *Question*: Is there an assignment A' of goal outcomes to the remaining games where A and A' agree on point assignments for every game such that team t can reach position at least K with respect to the rule of goal difference?

We show that the Guaranteed Point Placement problem is coNP-complete. We find this to be an interesting piece of information also for practical solutions. Though we cannot substantiate our point of view with a proof, since it might be that P = NP = coNP, it seems hard to verify the for-all requirement in the Guaranteed Point Placement problem, whereas heuristic methods used in IP solvers, for instance, are good at quickly solving existential problems such as checking if a given team could still win. Though inspired by the proof of Bernholt et al. [3], we have to consider

the complement of the Guaranteed Point Placement problem and show that to be NP-complete using a construction similar to, but different from theirs.

Starting from Bernholt et al. [3], researchers have tried to determine situations where the basic problem is polynomial-time solvable. We give a simple proof showing that even if we are in that situation, it turns out to be NP-complete to solve the Possible Goal Difference Placement problem for tied teams.

To return to the title, as opposed to football (or any other game played according to the $(1, 0)$-rule), where most questions can be resolved in polynomial time, it seems that almost all the questions we ask about soccer (or any other game played according to the $(3, 1, 0)$-rule) are hard.

2. It Is Hard to Give Guarantees

In this section, we prove that the Guaranteed Point Placement problem is coNP-complete.

Containment in coNP is easy to establish by showing that a no-instance can be verified in polynomial time. Given an assignment, the standing is updated with the results of each of the remaining games, and one verifies that the position of team t is worse than K. All this can be done in polynomial time.

To establish hardness, we show that the negated problem is NP-hard, i.e., given the same instance, we must show that it is NP-hard to decide if there exists an assignment where team t ends up in a position worse than K. Note that since the $(3, 1, 0)$-rule is asymmetric, this is not the same problem as the one considered by Bernholt et al. [3].

We fix K to be one less than the number of teams, thereby considering the special case of deciding if team t can place last. We also assume that the goal difference of team t is so bad (negative) that if t ends up with the same number of points as any other team, it will be placed under that team in the standing. Even this special case is NP-hard and that is what we will prove.

2.1. *Representation*

First, we make a transformation as follows: For each game a team still needs to play, we add one point to the total number of points they have obtained so far. This can be interpreted as assuming that the future games end in ties until we decide on the result. All remaining games are then played under the $(2, 0, -1)$-rule. This is clearly equivalent, and makes it a little easier to work with, since we only have to focus on games where there is a winner. If a game is a tie, no points change.

Also for convenience, we represent a Guaranteed Point Placement problem using a graph. Each team is represented by a vertex in the graph and the vertex is labeled with the difference between the total number of points obtained by the team and the total number of points obtained by the selected team t; still under the assumption of the $(2, 0, -1)$-rule as described above. We refer to this label as the value of the vertex. Each remaining game is represented as an edge between the two vertices

representing the two teams. In principle, this could give rise to a multi-graph, but we will not need that in the results to follow.

Considering this representation, for each edge where the corresponding game ends with a victory for one of the teams, the value of the vertex representing the winner is increased by two, the value of the vertex representing the loser is decreased by one, and the edge is removed. If a game is a tie, the corresponding edge can simply be removed without any further adjustments.

2.2. *Reduction*

We establish our result by a reduction from 3SAT, as formulated by Garey and Johnson [4, p. 259]:

> **3-Satisfiability (3SAT)**
> *Instance*: Set U of variables, collection C of clauses over U such that each clause $c \in C$ has $|c| = 3$.
> *Question*: Is there a satisfying truth assignment for C?

Given such an instance, we now explain which instance of the Guaranteed Point Placement problem we create.

The following reduction will be most elegant, if the number of clauses is a power of two. Thus, we pad the formula with a number of clauses of the form $(x_1 \vee \overline{x}_1 \vee x_1)$ until we have a number of clauses that is a power of two and use C to refer to this formula. Clearly, the original formula is satisfiable if and only if the padded formula is. The padding will at most double the number of clauses. We later show that our reduction can be carried out in polynomial time in $|C|$, where $|C|$ denotes the number of clauses in C. This is then also polynomial time in $|C|/2$. We may now assume that $|C|$ is a power of two.

For each propositional variable x_j appearing in C, we create a complete binary tree T_j with $2|C|$ leaves. Note that we keep referring to these vertices as leaves even though we connect them to other vertices, ending up with a graph which is not a tree.

Additionally, we create $|C|$ vertices V_i, referred to as clause vertices. If x_j appears (positively) in the ith clause, then we connect V_i to the ith leaf in the right subtree of the root of T_j. If $\overline{x}_j$ appears in the ith clause, then we connect V_i to the ith leaf in the left subtree of the root of T_j. All the roots of the T_j trees get the value (goal difference) one. Any leaf connected to some V_i vertex gets the value -2. Any other node gets the value zero. Finally, we create an isolated vertex representing team t having no games left. This vertex is labeled with the value zero.

In Figure 1, we show the tree for the propositional variable x_j in a context where the formula has eight clauses and point to the two possible ways of creating an edge.

In Figure 2, we illustrate the complete reduction for the formula $(x_1 \vee \overline{x}_3 \vee \overline{x}_4) \wedge (x_3 \vee x_2 \vee \overline{x}_1)$, except that we do not show the isolated vertex corresponding to t.

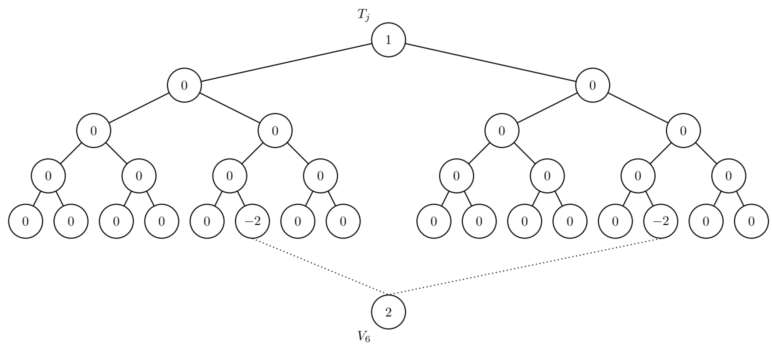

Fig. 1. A tree for $|C| = 8$. Each occurrence of a literal will give rise to exactly one edge connecting a clause vertex with a leaf. This figure illustrates the situation where the literal is x_j or $\overline{x}_j$ and it appears in the 6th clause. If the literal is $\overline{x}_j$, the connection will be to the 6th leaf counting from the left in the left subtree of the root. If the literal is x_j, the connection will be to the 6th leaf counting from the left in the right subtree of the root. Though we indicate -2 for both of these candidate leaves in the illustration, only the leaf connected in this way to V_6 will have the value -2 and the other candidate will have the value zero. Other leaves are listed as having the value zero, but these values will be -2 if they are connected to a clause vertex.

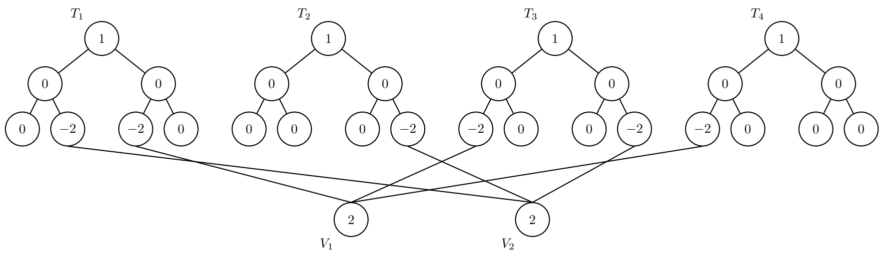

Fig. 2. An example of the entire reduction for the formula $(x_1 \vee \overline{x}_3 \vee \overline{x}_4) \wedge (x_3 \vee x_2 \vee \overline{x}_1)$.

Equivalence

Given a 3SAT formula C and the graph created from C by the reduction defined above, we now prove that C has a satisfying assignment if and only if team t can place last in the final standing. Recall that we assume that the goal difference of team t is so bad (negative) that if t ends up with the same number of points as any other team, it will be placed worse than that team in the final standing. Thus, if all other teams have at least as many points as t, then t is placed last.

The intended correspondence between a satisfying assignment for the formula and the outcomes of the games is as follows: If the propositional variable x_j is false in the satisfying assignment, then the root loses to its left child. Similarly, if x_j is true in the satisfying assignment, then the root loses to its right child.

For the first part of the equivalence, assume that C has a satisfying assignment.

To show that t can place last, we must find outcomes of the games such that all the teams end up with at least zero points, i.e., we must increase the points in vertices currently having the value -2.

Each clause vertex V_i is connected to three leaves in trees with value -2. Since we have a satisfying assignment, one of these values will be brought up to zero by the correspondence outlined above. Without loss of generality, assume that $\overline{x}_j$ is a literal that makes V_i true (so x_j is false), then we let the root of T_j lose to its left child, l, which will change the value of the root to zero and the value of l to two. Now, we let l lose to its children, which makes the value of l become zero, and the children of l will have their points increased by two. Continuing this way down to the leaves, all leaves in the left subtree of the root will have their values increased by two. In particular, the vertex in T_j that V_i is connected to has its value increased to zero. Thus, we can decide that V_i loses to the other two teams its connected to, bringing them up to zero, while V_i itself has its value decreased to zero; a decrease of one for each of these two games. By this, all leaves that V_i is connected to have non-negative points, and this construction can be carried out for all the clause vertices.

As an illustration of this proof, we continue the example from above. The example formula has the satisfying assignment

$$(x_1, x_2, x_3, x_4) = (\mathit{true}, \mathit{false}, \mathit{true}, \mathit{true}),$$

and this is equivalent to the assignment of outcomes shown in Figure 3.

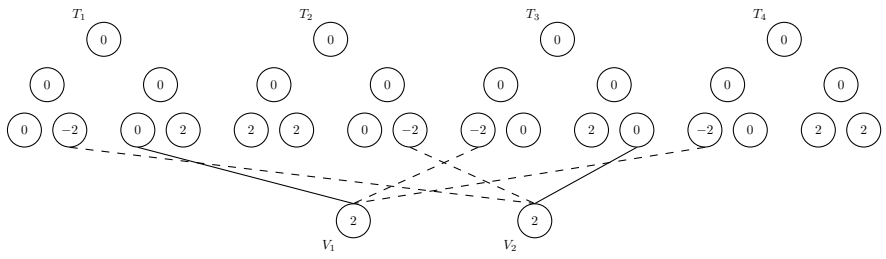

Fig. 3. Dashed edges show the games that the clause vertices should lose, and full edges are connected to leaves, the values of which are fixed by the games enabled by the satisfying assignment.

For the other direction, we must show that if outcomes can be made such that all vertices end up with non-negative values, we can find a satisfying assignment.

We first establish some facts regarding the outcomes of the games. First note that no leaf in any T_j can have its value brought up to more than zero, unless it wins over its parent, since either it has value zero and is not connected to any clause vertices, or it has value -2 and is connected to exactly one clause vertex.

By assumption, any clause vertex V_i ends up with a value of at least zero, so there is at least one game it does not lose. The only way the leaf representing the

other team in this game can have its value increased from -2 to zero is by winning over its parent p, making the value of p negative. If p wins over its other child, then that child will end up with a negative value, by the observation above, contradicting the assumption that t ends up last. Thus, p must win over its own parent, making the value negative there. Clearly this argument can be continued up to the root, where the value is changed from one to zero. Recursively, from the root down in the other subtree, this immediately implies that all games there will have to be ties to avoid that the parent in each game considered gets a negative value.

We now define x_j to be false if and only if the root of T loses to its left child. Consider some clause c_i in C. We want to argue that c_i is satisfied. Based on the discussion above, choose j such that V_i did not lose to the leaf u it is connected to in T_j. Again by the discussion above, u must be in the subtree where all vertices on the path from u to the root won over their parent. The edge from V_i to $u \in T_j$ means that either x_j or $\overline{x}_j$ appears in c_i. Without loss of generality, assume it is $\overline{x}_j$. Then, by the construction, the edge from V_i goes to the left subtree of T_j, so the root of T_j lost to its left child, meaning that x_j is assigned the value false, making $\overline{x}_j$ and therefore the clause c_i true.

We have established the equivalence and shown the following:

Theorem 1. *The Guaranteed Point Placement problem is coNP-complete.*

3. It is Hard to Resolve a Tie

In this section, we prove that the Possible Goal Difference Placement problem is NP-complete.

Containment in NP is easy to establish by verifying that the assignments A and A' agree on point assignments for every game, calculating the final standing, including goal differences, and checking if t is at position at least K, all of which can be done in polynomial time.

If t has at least one game left, its goal difference can be increased arbitrarily, so the hard problem is the one where t has finished all its games and its final placement depends on the goal outcomes of games it is not involved in.

Since we cannot change the points assigned to the different teams, but only goal differences, we assume that all teams under consideration have the same number of points, so we do not consider points any further. Without loss of generality, we assume that the goal difference of team t is zero, and that team t places best among teams with the same goal difference as t.

3.1. *Reduction*

To prove hardness, we reduce from Hitting Set, as formulated by Garey and Johnson [4, p. 222]:

> **Hitting Set**
> *Instance*: Collection C of subsets of a finite set S, positive integer

$K \leq |S|$.
Question: Is there a subset $S' \subseteq S$ with $|S'| \leq K$ such that S' contains at least one element from each subset in C?

Given such an instance, we now explain which instance of the Possible Goal Difference Placement problem we create.

We consider $|S| + |C|$ teams; one team for each element in S, referred to as element teams, and one team for each set in C, referred to as set teams. The goal difference is zero for each element team and one for each set team. The schedule of remaining games is the following: For each $c \in C$, and each $x \in c$, there is a game between the corresponding set team and element team, and in deciding points, as in the previous section, all element teams are set to win over the set teams they have scheduled games against.

3.2. *Equivalence*

We prove that there exists a subset $S' \subseteq S$ with $|S'| \leq K$ such that S' contains at least one element from each subset in C if and only if t can place at position $K+1$ or better.

First assume that there exists a subset $S' \subseteq S$ with $|S'| \leq K$ such that S' contains at least one element from each subset in C. For each $x \in S'$ and each game x is involved in, we let x win with one additional goal. This will decrease the goal differences of all set teams to zero, while increasing the goal differences of $|S'| \leq K$ element teams by one or more. Thus, at most K teams have positive goal difference, so team t places at position at least $K+1$.

For the other direction, assume that team t places at position $K+1$ or better. Thus, at most K teams have positive goal difference. If any of these teams are set teams, there exist other goal outcomes where team t places at position at least K: For any set team with positive goal difference, let any of the element teams it is playing against win sufficiently many additional goals to bring the goal difference of the set team to zero. This cannot decrease team t's final placement. Now only element teams can have positive goal difference, and we choose the hitting set S' as all the elements where the corresponding element team has a positive goal difference. By the argument above, $|S'| \leq K$. We must argue that S' is indeed a hitting set.

After the modification above, all set teams have non-positive goal difference. Thus, there must be a game, where the goal difference of some element team was increased. Element teams cannot have their goal differences decreased, so that element team ends with a positive goal difference. This includes the element into S' and this means that the set corresponding to the set team is hit.

We have established the equivalence and shown the following:

Theorem 2. *The Possible Goal Difference Placement problem is NP-complete.*

4. Concluding Remarks

We remark that the instances used in the hardness proofs can actually occur in a real tournament. It is easy, though a little space consuming, to extend the standing and schedules used to standard tournaments where each team meets any other team twice.

References

[1] I. Adler, A. L. Erera, D. S. Hochbaum and E. V. Olinick, Baseball, optimization, and the world wide web, *Interfaces* **32**(2) (2002) 12–22.

[2] H. Aziz, P. Harrenstein, M. Brill, J. Lang, F. A. Fischer and H. G. Seedig, Possible and necessary winners of partial tournaments, *Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, (IFAAMAS, 2012), pp. 585–592.

[3] T. Bernholt, A. Gülich, T. Hofmeister and N. Schmitt, Football elimination is hard to decide under the 3-point-rule, *24th International Symposium Mathematical Foundations of Computer Science (MFCS)*, *Lecture Notes in Computer Science* **1672**, (Springer, 1999), pp. 410–418.

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman and Company, 1979).

[5] W. Kern and D. Paulusma, The new FIFA rules are hard: complexity aspects of sports competitions, *Discrete Applied Mathematics* **108** (2001) 317–323.

[6] W. Kern and D. Paulusma, The computational complexity of the elimination problem in generalized sports competitions, *Discrete Optimization* **1** (2004) 205–214.

[7] S. T. McCormick, Fast algorithms for parametric scheduling come from extensions to parametric maximum flow, *Twenty-Eighth Annual ACM Symposium on the Theory of Computing (STOC)*, (ACM, 1996), pp. 319–328.

[8] D. Pálvölgyi, Deciding soccer scores and partial orientations of graphs, Tech. Rep. TR-2008-03, Egerváry Research Group, Institute of Mathematics, Eötvös Loránd University, Budapest, Hungary (2008).

[9] B. L. Schwartz, Possible winners in partially completed tournaments, *SIAM Review* **8**(3) (1966) 302–308.

[10] K. D. Wayne, A new property and a faster algorithm for baseball elimination, *SIAM Journal on Discrete Mathematics* **14**(2) (2001) 223–229.