# The Multi-Round Balanced Traveling Tournament Problem

**Richard Hoshino and Ken-ichi Kawarabayashi**
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan

## Abstract

Given an $n$-team sports league, the Traveling Tournament Problem (TTP) seeks to determine an optimal double round-robin schedule minimizing the sum total of distances traveled by the $n$ teams as they move from city to city. In the TTP, the number of "rounds" is fixed at $r = 2$. In this paper, we propose the Multi-Round Balanced Traveling Tournament Problem (mb-TTP), inspired by the actual league structure of Japanese professional baseball, where $n = 6$ teams play 120 intra-league games over $r = 8$ rounds, subject to various constraints that ensure competitive balance. These additional balancing constraints enable us to reformulate the $2k$-round mb-TTP as a shortest path problem on a directed graph, for all $k \geq 1$. We apply our theoretical algorithm to the 6-team Nippon (Japanese) Professional Baseball Central League, creating a distance-optimal schedule with 57836 kilometres of total travel, a 26.8% reduction compared to the 79067 kilometres traveled by these six teams during the 2010 regular season.

## Motivation

The Traveling Tournament Problem (TTP) is a challenging topic within the field of sports scheduling that involves techniques in computer science and discrete optimization. The TTP is a theoretical problem involving an $n$-team sports league whose output is a league schedule minimizing the total distance traveled by the $n$ teams. Since its introduction (Easton, Nemhauser, and Trick 2001), the TTP has emerged as a popular area of study within the operations research community due to its incredible complexity, where challenging benchmark problems remain unsolved.

In certain sports leagues, such as college basketball and Latin American soccer, each pair of teams plays two games, with one game scheduled at each team's home stadium. As teams must travel long distances to play all of their regular-season games, finding a schedule that reduces total travel distance is essential, for both economic and environmental reasons. The TTP is the correct framework for these leagues, as the desired output is a distance-optimal double round-robin schedule.



However, for professional baseball leagues, the TTP is not the correct framework, as teams play against each other multiple times during the course of a season. Furthermore, certain sports leagues impose scheduling constraints that go beyond the feasibility requirements of the TTP. In light of this, we propose a generalization of the TTP, the *Multi-Round Balanced Traveling Tournament Problem* (mb-TTP). In addition to extending the TTP to multiple rounds, we present additional "balancing" constraints that increase the fairness of a schedule, motivated by the actual structure of the Nippon Professional Baseball (NPB) league in Japan, where two leagues of $n = 6$ teams play 120 intra-league games over 8 rounds against 5 teams. We show that the NPB league schedule can be modeled by the mb-TTP.

In this paper, we present an algorithm that solves the $2k$-round mb-TTP for any $k \geq 1$, by reformulating it as a shortest path problem on a digraph. We then apply our graph-theoretic algorithm to the NPB Central League to produce a distance-optimal schedule that reduces the total distance traveled by 26.8% compared to the 2010 season schedule. We conclude the paper with directions for future research.

## The Multi-Round Balanced TTP

Let there be $n$ teams in a sports league, where $n$ is even. Let $D$ be the $n \times n$ *distance matrix*, where entry $D_{i,j}$ is the distance between the home stadiums of teams $i$ and $j$. By definition, $D_{i,j} = D_{j,i}$ for all $1 \leq i, j \leq n$, and all diagonal entries $D_{i,i}$ are zero.

| **Team** | C | T | D | B | G | S |
|---|---|---|---|---|---|---|
| C | 0 | 323 | 488 | 808 | 827 | 829 |
| T | 323 | 0 | 195 | 515 | 534 | 536 |
| D | 488 | 195 | 0 | 334 | 353 | 355 |
| B | 808 | 515 | 334 | 0 | 37 | 35 |
| G | 827 | 534 | 353 | 37 | 0 | 7 |
| S | 829 | 536 | 355 | 35 | 7 | 0 |

Table 1: Distance matrix for the NPB Central League (in km)

For example, the distance matrix for the NPB Central League is given in Table 1. Each of the six teams (Hiroshima Carp, Hanshin Tigers, Chunichi Dragons, Yokohama Baystars, Yomiuri Giants, Tokyo Swallows) is represented by the first letter of their team name.

In the TTP, a double round-robin schedule is sought, where each pair of teams plays twice during a tournament lasting $2(n-1)$ days, with each team having one game scheduled per day. As the context for our paper is baseball, we will now use *sets* rather than *days* to refer to the length of a tournament. Unlike other sports (e.g. football, soccer, hockey, basketball) where a team visits another city to play a single match, baseball leagues always involve a team visiting another city to play multiple games. To avoid any confusion, we will now re-define the TTP to the scheduling of $2(n-1)$ sets, where each set consists of a fixed number of games played on consecutive days.

The objective is to minimize the total distance traveled by the $n$ teams, with the requirement that each team begins the tournament at home, and returns home after having played their last away set. When a team is scheduled for a road trip consisting of multiple away sets, the team doesn't return to their home city but rather proceeds directly to their next away venue. In many ways, the TTP is a variant of the well-known Traveling Salesman Problem, asking for an optimal schedule linking venues that are close to one another.

In the TTP, a double round-robin schedule must satisfy the following conditions:

(a) The *each-venue* condition: Each pair of teams must play two sets, once in each other's home venue.

(b) The *at-most-three* condition: No team may have a home stand or road trip lasting more than three sets.

(c) The *no-repeat* condition: A team cannot play against the same opponent in two consecutive sets.

To illustrate, Table 2 lists the first ten sets of the NPB Central League for the 2010 season, where home teams are marked in bold. We see that this is a double round-robin schedule satisfying all of the above conditions.

| **Team** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Carp | D | **T** | **G** | S | B | **S** | **D** | T | G | **B** |
| Tigers | **B** | C | D | **G** | **S** | G | B | **C** | **D** | S |
| Dragons | **C** | S | **T** | B | G | **B** | C | **S** | T | **G** |
| Baystars | T | **G** | S | **D** | **C** | D | **T** | G | **S** | C |
| Giants | **S** | B | C | T | **D** | **T** | S | **B** | **C** | D |
| Swallows | G | **D** | **B** | **C** | T | C | **G** | D | B | **T** |

Table 2: First ten sets of the 2010 Central League Schedule.

Given a feasible solution of the TTP, for each team we define a *trip* to be a pair of consecutive sets not occurring in the same city (i.e., any situation where that team doesn't play at home in sets $s$ and $s+1$, and therefore has to travel from one venue to another.) For the case $n = 6$, each team plays five home sets and five away sets. Given that a home stand is at most three sets, it is easy to see that each team must make at least seven trips. Thus, in any feasible solution of the TTP, at least $7 \times 6 = 42$ total trips are required by the six teams. For the $n = 6$ case, this lower bound is known to be 43 (Rasmussen and Trick 2007). Intuitively, the solution of the TTP that minimizes the objective function of total travel distance will have around 43 trips. Table 2 has 51 trips, and is far from optimal!

The TTP involves both *integer programming* to prevent excessive travel, as well as *constraint programming* to create a schedule of home and away games that meet all the feasibility requirements. While each problem is simple to solve on its own, its combination has proven to be extremely challenging, even for small cases such as $n = 6$ and $n = 8$ (Easton, Nemhauser, and Trick 2001). The TTP has attracted much research in recent years, with sophisticated heuristics being developed from various areas of discrete optimization. For details, we refer the reader to two comprehensive survey articles on sports scheduling (Kendall et al. 2010) and the TTP (Rasmussen and Trick 2008).

In all of these papers, various algorithms have improved the upper and lower bounds for the $n$-team TTP. While there are several benchmark sets, the most common ones are based on the teams from MLB's National League (Easton, Nemhauser, and Trick 2001). In addition to the trivial NL4 case, there has been much analysis conducted on NL6, NL8, NL10, NL12, NL14, and NL16. We remark that in almost all of these papers, a technique was introduced to improve upon the known bounds rather than to prove optimality, due to the computational complexity of the problem for $n > 6$. The exceptions are two recently-published papers that successfully determined the optimal solution of NL8 using a novel branch-and-price approach (Irnich 2010) and a depth-first search with upper bounding that stored expensive heuristic estimates in memory to significantly reduce running time (Uthus, Riddle, and Guesgen 2009). Recently, it was shown that the TTP is NP-complete (Thielen and Westphal 2010), thus resolving a long-standing conjecture.

Define a *block* to be a feasible solution of the TTP, i.e., a tournament lasting $2(n-1)$ sets. We say that a block consists of two *rounds*, with the first round being the first $n-1$ sets and the second round being the last $n-1$ sets. In our multi-round extension, a tournament will have $k$ blocks, which is equivalent to having $2k$ rounds, or $2k(n-1)$ sets.

While much research has been conducted on the Traveling Tournament Problem, there has been no consideration given to tournaments lasting longer than two rounds. Conceivably, this is due to the computational complexity of analyzing a tournament lasting beyond $2(n-1)$ days. However, a multi-round tournament is the correct framework for professional baseball leagues, and so the design of an optimal schedule for NPB must take this into account. In addition to this multi-round extension (from $r = 2$ rounds to $r = 2k$ rounds for any $k \geq 1$), we propose two additional "balancing" constraints inspired by the actual league structure of professional baseball in Japan.

First, the NPB schedule requires each pair of teams to play exactly one set during each $(n-1)$-set round, which is stronger than the *each-venue* condition that only requires the two sets to be played sometime during a two-round block. As in the *each-venue* condition, these two sets must be played at different locations, with one set held at each team's home venue. We note that this condition is similar but not identical to the more stringent *mirrored* condition of Latin American soccer leagues (Ribeiro and Urrutia 2004), which has the rule that if team $i$ hosts team $j$ in set $s$ (where $1 \leq s \leq n-1$), then team $j$ hosts team $i$ in set $s+n-1$.

Secondly, the NPB schedule requires a balance in the number of home and away sets played by each team at any point in the season. More formally, for each ordered pair $(i, s)$ with $1 \leq i \leq n$ and $1 \leq s \leq 2k(n-1)$, define $H_{i,s}$ and $R_{i,s}$ to be the number of home and away sets played by team $i$ within the first $s$ sets. By definition, $|H_{i,s}+R_{i,s}| = s$. In the NPB, we require that $|H_{i,s} - R_{i,s}| \leq 2$ for all pairs $(i, s)$. For example, under this requirement a team cannot start or end a season with three consecutive home sets, ensuring that no team gains a momentum-increasing advantage at a key point in the season.

These two conditions are summarized below.

(d) The *each-round* condition: Each pair of teams must play exactly once per round, with their matches in rounds $2t-1$ and $2t$ taking place at different venues (for all $1 \leq t \leq k$).

(e) The *diff-two* condition: $|H_{i,s} - R_{i,s}| \leq 2$ for all $(i, s)$ with $1 \leq i \leq n$ and $1 \leq s \leq 2k(n-1)$.

In the following section, we present an algorithm to solve the the $2k$-round mb-TTP satisfying conditions (a)-(e). As mentioned above, condition (d) automatically implies (a). It is straightforward to check that Table 2 satisfies all five conditions of the mb-TTP for $k = 1$. When we apply our algorithm to the NPB Central League, we will set $k = 4$ and $n = 6$, since the six teams each play $2k(n-1) = 40$ sets of three intra-league games over $r = 2k = 8$ rounds.

The *each-round* condition allows us to apply the theory of perfect matchings, since each five-set round represents a one-factorization of the complete graph $K_6$. And as we will demonstrate in the following section, the *diff-two* condition allows us to define a simple "concatenation matrix" with just four columns to quickly verify the *no-repeat* and *at-most-three* conditions whenever two ten-set blocks are concatenated. Therefore, adding these two balancing constraints actually simplifies our analysis and makes the algorithm computationally feasible for the case $n = 6$.

We now present our algorithm for solving the $2k$-round mb-TTP, for any fixed $k \geq 1$, by reformulating it as a shortest path problem on a directed graph. We will create a source node and a sink node and link them to numerous vertices in a graph whose (weighted) edges represent the possible blocks that can appear in an optimal schedule. We then apply Dijkstra's Algorithm (Dijkstra 1959) to find the path of minimum weight between the source and the sink, which is an $O(|V| \log |V| + |E|)$ graph search algorithm that can be applied to any graph or digraph with non-negative edge weights.

## Shortest Path Algorithm for the mb-TTP

The mb-TTP can be formulated as an IP problem with $2k(n-1)$ time slots with each slot representing a set. But such an IP would have $k$ times as many variables as the original IP formulation for the 2-round TTP, and even for the $n = 6$ case, may not be computationally feasible for large $k$. As a result, we propose a graph-theoretic approach to solving the $2k$-round mb-TTP by reformulating the optimization problem into a shortest path problem on a digraph. While this methodology is more computationally laborious if $k$ is small, its utility will be demonstrated for large values of $k$.

To solve the mb-TTP, we first compute the set of blocks that *can* appear in a distance-optimal tournament. We then introduce a simple "concatenation matrix" to check whether two pre-computed blocks can be joined together to form a multi-block schedule, without violating the *at-most-three* and *no-repeat* conditions. As we will explain, to determine whether blocks $B_1$ and $B_2$ can be concatenated, it suffices to check just the last two columns of $B_1$ and the first two columns of $B_2$.

By definition, a block is a two-round tournament satisfying the conditions of the mb-TTP, with each of the $n$ teams playing $2(n-1)$ sets of games. Each column of a block represents a set consisting of $\frac{n}{2}$ different *matches*, with each match specifying the two teams as well as the stadium/venue. Thus, a match identifies the home team and away team, not just each team's opponent.

For any column in a block, there are $\binom{n}{n/2}$ ways to select the home teams. Also there are $\binom{n}{n/2} \cdot \left(\frac{n}{2}\right)!$ ways to specify the matches of any column, since there are $\left(\frac{n}{2}\right)!$ ways to map any choice of the $\frac{n}{2}$ home teams to the unselected $\frac{n}{2}$ away teams to decide the set of $\frac{n}{2}$ matches. Hence, there are $m = \binom{n}{n/2}^2 \cdot \left(\frac{n}{2}\right)!$ different ways we can specify the home teams of one column *and* the matches of another column. For $n = 6$, we have $m = \binom{6}{3}^2 \times 3! = 2400$.

There are $m$ ways that the first two columns of a block can be chosen as described above, with the first column listing matches and the second column listing home teams. Now use any method, such as a lexicographic ordering, to index these $m$ options with the integers from 1 to $m$. By symmetry, there are $m$ different ways we can specify the last two columns of a block, with the last column listing matches and the second-last column listing home teams. Thus, we use the same scheme to index these $m$ options. To avoid confusion, we write the home teams column in binary form, with $\mathbf{1}$ representing a home game and 0 representing an away game.

As an example, consider Table 2. For the 6-team NPB Central League, there exists some integer $p$ (with $1 \leq p \leq 2400$) that is the index of the instance where the matches column is $(\mathrm{D}, \mathbf{B}, \mathbf{C}, \mathrm{T}, \mathbf{S}, \mathrm{G})^T$ and the home teams column is $(\mathbf{1}, 0, 0, \mathbf{1}, 0, \mathbf{1})^T$. Similarly, there exists some $q$ (with $1 \leq q \leq 2400$) that is the index of the instance where the two columns are $(\mathbf{B}, \mathrm{S}, \mathbf{G}, \mathrm{C}, \mathrm{D}, \mathbf{T})^T$ and $(0, \mathbf{1}, 0, \mathbf{1}, \mathbf{1}, 0)^T$. Table 2 is a block for which the first two columns have index $p$ and the last two columns have index $q$.

For each pair $(u_1, u_2)$, with $1 \leq u_1, u_2 \leq m$, define $C_{u_2,u_1}$ to be the $n \times 4$ *concatenation matrix* where the first two columns list the home teams and matches with index $u_2$, and the next two columns list the matches and home teams with index $u_1$. For the indices $p$ and $q$ from the previous paragraph, we have

$$C_{q,p} = \begin{bmatrix} 0 & \mathbf{B} & \mathrm{D} & \mathbf{1} \\ \mathbf{1} & \mathrm{S} & \mathbf{B} & 0 \\ 0 & \mathbf{G} & \mathbf{C} & 0 \\ \mathbf{1} & \mathrm{C} & \mathrm{T} & \mathbf{1} \\ \mathbf{1} & \mathrm{D} & \mathbf{S} & 0 \\ 0 & \mathbf{T} & \mathrm{G} & \mathbf{1} \end{bmatrix}.$$

We now explain the role of $m$ and $C_{u_2,u_1}$ in the construction of our directed graph. Let $G$ consist of a source vertex $v_{start}$, a sink vertex $v_{end}$, and vertices $x_{t,u}$ and $y_{t,u}$ defined for each $1 \leq t \leq k$ and $1 \leq u \leq m$.

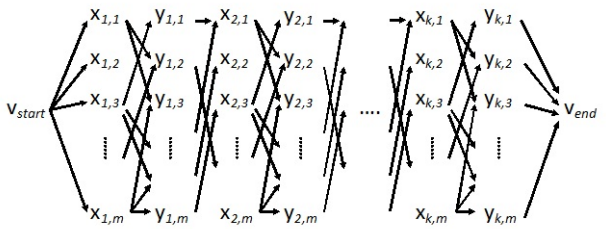


Figure 1: Reformulation of mb-TTP as a shortest path problem.

We now describe how these edges are connected, with a pictorial representation of $G$ in Figure 1. For notational simplicity, denote $v_1 \rightarrow v_2$ as the directed edge from $v_1$ to $v_2$.

(a) For each $1 \leq u \leq m$, add the edge $v_{start} \rightarrow x_{1,u}$.

(b) For each $1 \leq u \leq m$, add the edge $y_{k,u} \rightarrow v_{end}$.

(c) For each $1 \leq t \leq k$, and for each $1 \leq u_1, u_2 \leq m$, add the edge $x_{t,u_1} \rightarrow y_{t,u_2}$ iff there exists a (feasible) block for which the first two columns have index $u_1$ and the last two columns have index $u_2$.

(d) For each $1 \leq t \leq k-1$, and for each $1 \leq u_1, u_2 \leq m$, add the edge $y_{t,u_2} \rightarrow x_{t+1,u_1}$ iff the concatenation matrix $C_{u_2,u_1}$ has no row with four home sets, no row with four away sets, and no row with the same opponent appearing in Columns 2 and 3.

The following theorem shows that the mb-TTP can be reformulated in a graph-theoretic context, for any $k \geq 1$.

**Theorem 1** *Every feasible solution of the mb-TTP can be described by a path from $v_{start}$ to $v_{end}$ in graph $G$. Conversely, any path from $v_{start}$ to $v_{end}$ in $G$ corresponds to a feasible solution of the mb-TTP.*

**Proof** By the definition of $G$, any path from $v_{start}$ to $v_{end}$ has length $2k+1$, and is of the form $P = v_{start} \rightarrow x_{1,p_1} \rightarrow y_{1,q_1} \rightarrow x_{2,p_2} \rightarrow y_{2,q_2} \rightarrow \ldots \rightarrow x_{k,p_k} \rightarrow y_{k,q_k} \rightarrow v_{end}$.

We first show that $P$ corresponds to a schedule that is a feasible solution of the mb-TTP. For each $1 \leq t \leq k$, let $B_t$ be any block for which the first two columns have index $p_t$ and the last two columns have index $q_t$. By part (c) of our construction, since $x_{t,p_t} \rightarrow y_{t,q_t}$ is an edge of $G$, such a block $B_t$ must necessarily exist. Since each $B_t$ is a (feasible) block, all of the balancing constraints of the mb-TTP hold within that block: each pair of teams plays exactly once per round with one match at each venue, no team has a home stand or road trip lasting more than three sets, a team does not play the same opponent in consecutive sets, and $|H_{i,s} - R_{i,s}| \leq 2$ for all $1 \leq i \leq n$ and $1 \leq s \leq 2(n-1)$.

We claim that $B_1, B_2, \ldots, B_k$, the concatenation of these $k$ blocks, is a feasible solution of the mb-TTP. Clearly each team plays $2k(n-1)$ sets of games, and each pair of teams plays once per round with their matches in rounds $2t-1$ and $2t$ taking place at different venues (for all $1 \leq t \leq k$). Since each team plays $n-1$ home sets and $n-1$ away sets in every block, we have $H_{i,2t(n-1)} = R_{i,2t(n-1)}$ for all $1 \leq t \leq k$. This implies that $|H_{i,s} - R_{i,s}| \leq 2$ for all $1 \leq i \leq n$ and $1 \leq s \leq 2k(n-1)$, since this difference function $|H_{i,s} - R_{i,s}|$ resets to 0 at the end of every block. Therefore, the *each-round* and *diff-two* conditions are satisfied.

To complete our claim, we must justify that the *at-most-three* and *no-repeat* conditions are not violated. Based on the previous paragraph, a violation can only be created in the concatenation of two blocks, specifically in one of the following four situations:

(1) The last column of block $B_t$ matches up with the first column of block $B_{t+1}$ in at least one row, i.e., there exists some team that plays the same opponent in sets $2t(n-1)$ and $2t(n-1)+1$, thereby violating the *no-repeat* condition.

(2) Some row of $B_t$ ends with two home (away) sets and the same row of $B_{t+1}$ begins with two home (away) sets, thereby violating the *at-most-three* condition from set $2t(n-1)-1$ to $2t(n-1)+2$.

(3) Some row of $B_t$ ends with three home (away) sets and the same row of $B_{t+1}$ begins with a home (away) set, thereby violating the *at-most-three* condition from set $2t(n-1)-2$ to $2t(n-1)+1$.

(4) Some row of $B_t$ ends with a home (away) set and the same row of $B_{t+1}$ begins with three home (away) sets, thereby violating the *at-most-three* condition from set $2t(n-1)$ to $2t(n-1)+3$.

Consider the home teams in the second-last column of $B_t$, the matches in the last column of $B_t$, the matches in the first column of $B_{t+1}$, and the home teams in the second column of $B_{t+1}$. These four columns, representing the sets from $2t(n-1)-1$ to $2t(n-1)+2$, are precisely the four columns of the concatenation matrix $C_{q_t,p_{t+1}}$. Since the path $P$ includes the edge $y_{t,q_t} \rightarrow x_{t+1,p_{t+1}}$, the matrix $C_{q_t,p_{t+1}}$ has no row with four home sets, no row with four away sets, and no row with the same opponent appearing in Columns 2 and 3. Hence, no team has a four-set home stand or road trip from set $2t(n-1)-1$ to $2t(n-1)+2$, and no team plays the same opponent in sets $2t(n-1)$ and $2t(n-1)+1$. This proves that neither situations (1) or (2) can occur.

As noted before, $H_{i,2t(n-1)} = R_{i,2t(n-1)}$, for all $t$. If row $i$ of $B_t$ ends with three consecutive home or away sets, then $|H_{i,2t(n-1)-3} - R_{i,2t(n-1)-3}| = 3 > 2$, a contradiction. Similarly, if row $i$ of $B_{t+1}$ begins with three consecutive home or away sets, then $|H_{i,2t(n-1)+3} - R_{i,2t(n-1)+3}| = 3 > 2$, a contradiction. This proves that neither situations (3) or (4) can occur.

We have shown that the concatenation of these $k$ blocks, namely $B_1, B_2, \ldots, B_k$, is a feasible solution of the mb-TTP. To conclude the proof, we establish the converse.

Note that any feasible solution of the mb-TTP can be partitioned into $k$ non-overlapping blocks. Let $B_t$ be the $t^{\text{th}}$ block of this feasible solution. For this block $B_t$, let $p_t$ be the index of the first two columns and let $q_t$ be the index of the last two columns. Since $B_t$ is a (feasible) block, it follows that each $x_{t,p_t} \rightarrow y_{t,q_t}$ is an edge in $G$. Since pairwise

blocks can be concatenated without violating the constraints of the mb-TTP, $y_{t,q_t} \rightarrow x_{t+1,p_{t+1}}$ in an edge in $G$ for all $1 \leq t \leq k-1$. We conclude that $P = v_{start} \rightarrow x_{1,p_1} \rightarrow y_{1,q_1} \rightarrow x_{2,p_2} \rightarrow y_{2,q_2} \rightarrow \ldots \rightarrow x_{k,p_k} \rightarrow y_{k,q_k} \rightarrow v_{end}$ is a path in $G$ connecting $v_{start}$ to $v_{end}$. ∎

Having constructed our digraph, we now assign a weight to each edge using the distance matrix $D$ so that the shortest path (i.e., path of minimum total weight) from $v_{start}$ to $v_{end}$ corresponds to the desired solution of the mb-TTP that minimizes the total distance traveled by the $n$ teams.

For any block, we define its *in-distance* to be the total distance traveled by the $n$ teams within that block, i.e., starting from set 1 and ending at set $2(n-1)$. Note that the in-distance does not include the distance traveled by the teams heading *to* the venue of set 1 or *from* the venue of set $2(n-1)$. We will use this definition in part (c) below.

We are now ready to assign the edge weights of $G$.

(a) For each $1 \leq u \leq m$, the weight of $v_{start} \rightarrow x_{1,u}$ is the distance traveled by the $\frac{n}{2}$ teams making the trip from their home city to the venue of their opponent in set 1.

(b) For each $1 \leq u \leq m$, the weight of $y_{k,u} \rightarrow v_{end}$ is the distance traveled by the $\frac{n}{2}$ teams making the trip from the venue of their opponent in set $2k(n-1)$ back to their home city.

(c) For each $1 \leq t \leq k$, and for each $1 \leq u_1, u_2 \leq m$, the weight of $x_{t,u_1} \rightarrow y_{t,u_2}$ is the *minimum* in-distance of a block, selected among all blocks for which the first two columns have index $u_1$ and the last two columns have index $u_2$.

(d) For each $1 \leq t \leq k-1$, and for each $1 \leq u_1, u_2 \leq m$, the weight of $y_{t,u_2} \rightarrow x_{t+1,u_1}$ is the distance traveled by all the teams that make a trip as they travel from their match in set $2t(n-1)$ to their match in set $2t(n-1)+1$, where the last two columns of the $t^{\text{th}}$ block have index $u_2$ and the first two columns of the $(t+1)^{\text{th}}$ block have index $u_1$.

To illustrate (d), consider the two-block schedule produced by concatenating two copies of Table 2. Then this is a feasible solution of the mb-TTP for $k = 2$, with path $P = v_{start} \rightarrow x_{1,p} \rightarrow y_{1,q} \rightarrow x_{2,p} \rightarrow y_{2,q} \rightarrow v_{end}$, having total weight $1010+15895+1707+15895+1697 = 36204$. The term 1707, representing the weight of edge $y_{1,q} \rightarrow x_{2,p}$, is the distance traveled by the teams from their matches in set 10 to their matches in set 11. From Table 1, this total is $D_{C,D} + D_{S,T} + D_{C,T} + D_{D,G} + D_{S,G} = 1707$.

By this construction, we have produced a weighted digraph. For step (c), suppose there exist two blocks $B$ and $B'$ for which the first two columns have index $u_1$ and the last two columns have index $u_2$. If the in-distance of $B$ is less than the in-distance of $B'$, then block $B'$ cannot be a block in an optimal solution, since we can just replace $B'$ by $B$ to create a feasible solution with a lower objective value. This trivial observation, based on Bellman's Principle of Optimality, allows us to assign the *minimum* in-distance as the weight of edge $x_{t,u_1} \rightarrow y_{t,u_2}$, for all $1 \leq u_1, u_2 \leq m$. As a result, we have a digraph $G$ on $2mk+2$ vertices and at most $2m + (2k-1)m^2$ edges, with a unique weight for each edge. Combined with the previous theorem, we have established the following.

**Theorem 2** *Let $P = v_{start} \rightarrow x_{1,p_1} \rightarrow y_{1,q_1} \rightarrow x_{2,p_2} \rightarrow y_{2,q_2} \rightarrow \ldots \rightarrow x_{k,p_k} \rightarrow y_{k,q_k} \rightarrow v_{end}$ be a shortest path in $G$ from $v_{start}$ to $v_{end}$, i.e., a path that minimizes the total weight. For each $1 \leq t \leq k$, let $B_t$ be the block of minimum in-distance selected among all blocks for which the first two columns have index $p_t$ and the last two columns have index $q_t$. Then the multi-block schedule $S = B_1, B_2, \ldots, B_k$, created by concatenating these $k$ blocks, is an optimal solution of the mb-TTP.*

## Constructing the Digraph

We have shown that the mb-TTP is isomorphic to a finding the shortest weighted path in the directed graph $G$. We now build $G$ for 6-team NPB Central League, applying the distance matrix from Table 1.

First, note that for steps (c) and (d), these two constructions are independent of $t$. Thus, to construct the edges $x_{t,u_1} \rightarrow y_{t,u_2}$, it suffices to determine all the edges from $x_{1,u_1}$ to $y_{1,u_2}$ with the corresponding weights and replicate that $k$ times. To construct the edges $y_{t,u_2} \rightarrow x_{t+1,u_1}$, it suffices to determine all the edges from $y_{1,u_2}$ to $y_{2,u_1}$ with the corresponding weights and replicate that $k-1$ times. Regardless of $k$, we only need to compute steps (c) and (d) once, and we will store all of the necessary information in $m \times m$ matrices, which for the $n = 6$ case would be precomputed matrices with dimensions $2400 \times 2400$.

To perform step (c) of the construction, we must ensure that the weight of each edge $x_{1,u_1} \rightarrow y_{1,u_2}$ is the *minimum* in-distance of a block for which the first two columns have index $u_1$ and the last two columns have index $u_2$. To accomplish this, we adopt a three-phase approach that is a common heuristic for solving the two-round TTP (Rasmussen and Trick 2007). Phase one generates double round-robin home-away pattern (HAP) sets in the form of an $n$ by $2(n-1)$ matrix, phase two converts these HAP sets into timetables which are assignments of matches to time slots, and phase three converts timetables into feasible schedules (i.e., blocks) by assigning each team a unique row in the matrix. Once we have enumerated all possible blocks, we will be able to assign the proper weight to each edge $x_{1,u_1} \rightarrow y_{1,u_2}$ by applying the distance matrix $D$. To illustrate, Table 3 is a HAP set satisfying the mb-TTP conditions, with a corresponding timetable in Table 4.

| **Team** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **H** | A | A | **H** | **H** | **H** | A | A | A | **H** |
| 2 | A | **H** | **H** | **H** | A | A | A | **H** | **H** | A |
| 3 | A | A | **H** | **H** | **H** | A | A | A | **H** | **H** |
| 4 | **H** | **H** | A | A | A | **H** | **H** | **H** | A | A |
| 5 | **H** | A | A | A | **H** | **H** | **H** | A | A | **H** |
| 6 | A | **H** | **H** | A | A | A | **H** | **H** | **H** | A |

Table 3: A balanced HAP set.

To turn a timetable into a block, one simply maps the six Central League teams to any of the 6! permutations of $\{1, 2, 3, 4, 5, 6\}$. We note that any timetable can be turned

| **Team** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **2** | 6 | 3 | **5** | **4** | **3** | 5 | 4 | 2 | **6** |
| 2 | 1 | **3** | **4** | **6** | 5 | 4 | 6 | **5** | **1** | 3 |
| 3 | 5 | 2 | **1** | **4** | **6** | 1 | 4 | 6 | **5** | **2** |
| 4 | **6** | **5** | 2 | 3 | 1 | **2** | **3** | **1** | 6 | 5 |
| 5 | **3** | 4 | 6 | 1 | **2** | **6** | **1** | 2 | 3 | **4** |
| 6 | 4 | **1** | **5** | 2 | 3 | 5 | **2** | **3** | **4** | 1 |

Table 4: A timetable corresponding to this HAP set.

into a block, but a HAP set does not necessarily produce a feasible timetable. For example, consider a HAP set with two identical rows, say in rows $i$ and $j$. Then this HAP set will not produce a feasible timetable as teams $i$ and $j$ cannot play one another since there is no time slot where one team is home while the other is on the road. Using the theory of one-factorizations, we show that there are 627944 possible HAP sets, which generate 169728 feasible timetables (Hoshino and Kawarabayashi 2010).

From these 169728 timetables, we calculate the in-distance of all $169728 \times 6!$ blocks that can be generated and determine the indices of their first and last two columns. From this we determine the $2400 \times 2400$ matrix $M$, where $M[u_1, u_2]$ is the minimum in-distance of a block, selected among all blocks for which the first two columns have index $u_1$ and the last two columns have index $u_2$. We determine that 2618520 of the $2400^2 = 5760000$ possible entries of $M$ are defined, all of whose values lie in the range $[13075, 22189]$.

For the other $2400^2 - 2618520 = 3141480$ cases, there does not exist a feasible block with indices $u_1$ and $u_2$, and so we define $M[u_1, u_2] := 10^5$ in these cases to ensure that the matrix $M$ is well-defined. In a shortest path algorithm, a non-edge can be replaced by a fake edge with massive weight (such as $10^5$) that dominates the weight of all actual edges. As a result, all the information on the edges and edge weights of $x_{1,u_1} \to y_{1,u_2}$ can be stored in this single matrix $M$, and when we apply Dijkstra's Algorithm to find the path that minimizes the total weight, an edge of weight $10^5$ will of course not appear in the output.

Finally, we proceed with step (d) of our construction. We define a $2400 \times 2400$ matrix $N$ which will store the weights of all edges $y_{1,u_2} \to x_{2,u_1}$. By definition, $y_{1,u_2} \to x_{2,u_1}$ is an edge in $G$ iff the concatenation matrix $C_{u_2,u_1}$ has no row with four home sets, no row with four away sets, and no row with the same opponent appearing in columns 2 and 3. We find that 1486320 of the $2400^2 = 5760000$ choices for $C_{u_2,u_1}$ satisfy the requirements for $y_{1,u_2} \to x_{2,u_1}$ to be an edge of $G$. In all these cases, the edge weight $N[u_2, u_1]$ is determined from $C_{u_2,u_1}$ by calculating the distance traveled by all the teams that make a trip from their match in column 2 to their match in column 3. We find that the 1468320 values of $N[u_2, u_1]$ lie in the range $[79, 3394]$. In the other $2400^2 - 1486320 = 4273680$ cases, set $N[u_2, u_1] := 10^5$, as we did before to ensure a well-defined matrix.

Clearly steps (a) and (b) in our construction each add 2400 edges to $G$. From above, step (c) adds $2618520k$ edges and step (d) adds $1486320(k-1)$ edges. Therefore, graph $G$ consists of $2mk + 2 = 4800k + 2$ vertices and $2400 + 2400 + 2618520k + 1486320(k-1) = 4104840k - 1481520$ edges. We now apply Dijkstra's shortest path algorithm to obtain the $k$-block $2k$-round schedule that minimizes the total travel distance.

Dijkstra's Algorithm constructs a shortest path tree from the initial vertex to every other vertex in the graph. For each vertex $v$ in $G$, let $w(v)$ be the weight of the shortest path from $v_{start}$ to that vertex. By definition, $w(v_{start}) = 0$. For each $1 \le u \le m$, $w(x_{1,u})$ is the total distance traveled by the $\frac{n}{2}$ teams making the trip from their home city to the venue of their opponent in set 1.

For each $1 \le u \le m$ and $1 \le t \le k$, we have

$$w(y_{t,u}) = \min_{1 \le u^* \le m} \{w(x_{t,u^*}) + M[u^*, u]\}.$$

And for each $1 \le u \le m$ and $1 \le t \le k-1$, we have

$$w(x_{t+1,u}) = \min_{1 \le u^* \le m} \{w(y_{t,u^*}) + N[u^*, u]\}.$$

The last step of this double-recursion generates $w(y_{k,u})$, the weight of the shortest path from $v_{start}$ to each $y_{k,u}$. For each $u$, define $last(u)$ to be the total distance traveled by the $\frac{n}{2}$ teams making the trip from the venue of their opponent in set $2k(n-1)$ back to their home city. By symmetry, it is clear that $last(u) = w(x_{1,u})$. Therefore, we have

$$w(v_{end}) = \min_{1 \le u^* \le m} \{w(y_{k,u^*}) + w(x_{1,u^*})\}.$$

The value of $w(v_{end})$ is the distance of the optimal solution to the $k$-block mb-TTP, corresponding to some path $P = v_{start} \to x_{1,p_1} \to y_{1,q_1} \to x_{2,p_2} \to y_{2,q_2} \to \ldots \to x_{k,p_k} \to y_{k,q_k} \to v_{end}$. For each $1 \le t \le k$, let $B_t$ be the block of minimum in-distance selected among all blocks for which the first two columns have index $p_t$ and the last two columns have index $q_t$. By Theorem 2, the desired distance-optimal schedule is the concatenation of $B_1, B_2, \ldots, B_k$.

## Running Time

All of our code was written in Maple and compiled using Maplesoft 13 using a single Toshiba laptop under Windows with a single 2.10 GHz processor and 2.75 GB RAM. Various methods, such as exploiting symmetry, were used to reduce the computation time.

Maple required 13 hours to generate the 627944 feasible HAP sets, and required an additional 67 hours of computation time to turn these HAP sets into the 169728 feasible timetables. While this process takes over three days of computation time, this procedure needs to be only run once, and then the 169728 timetables can be stored in a file and applied to any distance matrix $D$. For the interested reader, the authors would be happy to provide the files containing these 169728 timetables, as well as all of the Maple code used in the production of this paper.

It takes Maple less than 0.1 seconds to complete step (a) of the digraph construction, assigning a weight to each edge $v_{start} \to x_{1,u}$. Step (b), which assigns a weight to each edge $y_{k,u} \to v_{end}$, also takes 0.1 seconds. Step (d), which determines the weights of all edges $y_{1,u_2} \to x_{2,u_1}$, and stores this

information in the $2400 \times 2400$ matrix $N$, takes just 53 seconds. Finally, step (c) takes almost 5 hours (17346 seconds) of computation time. This is by far the most expensive calculation in the algorithm as we must consider all blocks that can be generated from our 169728 feasible timetables. For each block, we determine the indices $u_1$ and $u_2$, calculate the in-distance, and check whether we need to update the values of $M[u_1, u_2]$, $M[u_2, u_1]$, $B[u_1, u_2]$, and $B[u_2, u_1]$. This information is summarized in the table below.

| Process | Edge Construction | Edges Created (#) | Time (s) |
|---|---|---|---|
| Step (a) | $v_{start} \rightarrow x_{1,u}$ | 2400 | 0.1 |
| Step (b) | $y_{k,u} \rightarrow v_{end}$ | 2400 | 0.1 |
| Step (c) | $x_{1,u_1} \rightarrow y_{1,u_2}$ | 2,618,520 | 17346 |
| Step (d) | $y_{1,u_2} \rightarrow x_{2,u_1}$ | 1,486,320 | 53 |

Table 5: Running time for each step of the mb-TTP algorithm.

The weakness of our graph-theoretic approach is the running time of step (c). One possible time-saving technique is to store expensive heuristic estimates in memory, so that the same calculations do not need to be repeated multiple times. In one recent paper (Uthus, Riddle, and Guesgen 2009), the authors use this memory-storage technique to solve the 2-round TTP for the 8-team NL8 benchmark set, reducing the running time of their depth-first-search algorithm from 26 hours to just 7 minutes on a single processor. Perhaps the same technique can be applied to reduce the running time of step (c) of our algorithm from five hours to several minutes.

Finally we apply Dijkstra's Algorithm. For each $k$, applying the above recursive procedure to find the shortest path from $v_{start}$ to $v_{end}$ takes approximately $25k$ seconds. Hence, incrementing $k$ by one adds only 25 seconds to the total computation time. Once we have determined the matrices $M$ and $N$, Dijkstra's Algorithm enables us to determine the optimal 100-round schedule ($k = 50$) in just twenty minutes. While our proposed graph-theoretic approach may not be computationally efficient for small values of $k$, its utility is clearly evident for large values of $k$.

## Application to the Japanese NPB League

We apply our shortest path algorithm to optimize the scheduling for Nippon Professional Baseball (NPB). The NPB consists of the six-team Central League and the six-team Pacific League. All teams play 144 games during a season, with 120 intra-league games and 24 inter-league games. Specifically, an NPB team plays twelve home games and twelve away games against each of the other five teams in its league ($24 \times 5$) in addition to two home games and two away games against all six teams in the other league ($4 \times 6$). All twenty-four inter-league games take place during a common five-week stretch beginning in mid-May, right near the start of the season.

Our algorithm for the mb-TTP will only apply to the 120 intra-league games; the remaining 24 inter-league games will not be examined in this paper as it is a separate optimization problem based on the theory of bipartite tournaments (Hoshino and Kawarabayashi 2011). For the purposes of this paper, we will assume that the NPB schedule consists of 120 intra-league games, ignoring the 24 inter-league games and the annual July break for the all-star game. When we demonstrate that the NPB Central League schedule can be improved by 26.8% with respect to total travel distance, we are strictly referring to the scheduling of these 120 intra-league games.

As in Major League Baseball, nearly all NPB games occur in sets of three games. Thus, we will adopt the same structure when building our schedule, so that each of the $n = 6$ teams plays 40 sets of three games. Hence, we require a schedule with $k = 4$ blocks and $r = 2k = 8$ rounds, producing $r(n-1) = 40$ sets per team.

Applying our shortest path algorithm to graph $G$, we determine that $w(v_{end}) = 57836$ and that the shortest path is $P = v_{start} \rightarrow x_{1,p_1} \rightarrow y_{1,q_1} \rightarrow x_{2,p_2} \rightarrow y_{2,q_2} \rightarrow x_{3,p_3} \rightarrow y_{3,q_3} \rightarrow x_{4,p_4} \rightarrow y_{4,q_4} \rightarrow v_{end}$, with the edge weights $664, 13131, 729, 13665, 1100, 13075, 1686, 13075, 711$, respectively. In this shortest path, we have $p_3 = p_4$ and $q_3 = q_4$, which explains why the last two blocks in Table 6 are identical. We remark that the 40-set schedule can be written backwards to produce another optimal schedule with the same objective value of 57836.

| **Team** | R1 | R2 | R3 | R4 |
|---|---|---|---|---|
| Carp | **TG**BSD | **BSD**TG | D**TGS**B | GS**BD**T |
| Tigers | C**DG**BS | G**BSC**D | BC**DGS** | DGS**BC** |
| Dragons | **B**TSG**C** | **SG**CB**T** | **C**STB**G** | **TBGCS** |
| Baystars | D**SCT**G | CT**GDS** | **TGSDC** | **S**DCT**G** |
| Giants | **SCTDB** | **T**DBS**C** | **SB**CTD | **CTDS**B |
| Swallows | GB**DCT** | DCT**GB** | G**DB**CT | B**CTG**D |

| **Team** | R5 | R6 | R7 | R8 |
|---|---|---|---|---|
| Carp | BT**DSG** | DSG**BT** | BT**DSG** | DSG**BT** |
| Tigers | D**CSG**B | SG**BDC** | D**CSG**B | SG**BDC** |
| Dragons | **TGCBS** | **C**BST**G** | **TGCBS** | **C**BST**G** |
| Baystars | **C**SGD**T** | **GDTCS** | **C**SGD**T** | **GDTCS** |
| Giants | S**DB**TC | B**TCS**D | S**DB**TC | B**TCS**D |
| Swallows | **GB**TCD | **TCD**GB | **GB**TCD | **TCD**GB |

Table 6: Optimal schedule for the Central League, $r = 8$ rounds.

During the 2010 NPB season, every Central League team made at least 33 trips. Under our optimal schedule, no team would make any more than 29 trips. We compare these two schedules with respect to the total distance traveled by all six teams, using Table 1. As seen in Table 7, our optimal schedule achieves a 26.8% reduction in total travel compared to the existing schedule, in addition to a 14.6% reduction in total trips taken.

Despite the impressive 26.8% reduction in total distance traveled, we note that one team (Chunichi Dragons) achieves a paltry 1.1% reduction under our optimal schedule. This discrepancy can be corrected by proposing a non-optimal feasible schedule with an objective value just slightly higher than 57836 kilometres where the team-by-team percentage improvement is more equitably distributed.

We repeat the same algorithm for the Pacific League (Hoshino and Kawarabayashi 2010), producing a distance-optimal schedule that achieves a 25.8% reduction in total

| | Distance (Existing) | Distance (Optimal) | % Reduction in Distance |
|---|---|---|---|
| Carp | 17850 | 11741 | 34.2 % |
| Tigers | 14304 | 8712 | 39.1 % |
| Dragons | 11790 | 11665 | 1.1 % |
| Baystars | 13104 | 8929 | 31.9 % |
| Giants | 11469 | 9020 | 21.4 % |
| Swallows | 10550 | 7769 | 26.4 % |
| Total | 79067 | 57836 | 26.8 % |

| | Trips (Existing) | Trips (Optimal) | % Reduction in Trips |
|---|---|---|---|
| Carp | 33 | 27 | 18.2 % |
| Tigers | 33 | 29 | 12.1 % |
| Dragons | 33 | 28 | 15.2 % |
| Baystars | 34 | 29 | 14.7 % |
| Giants | 33 | 28 | 15.2 % |
| Swallows | 33 | 29 | 12.1 % |
| Total | 199 | 170 | 14.6 % |

Table 7: Comparison of existing and optimal schedules in terms of total distance traveled and total trips taken.

distance traveled and a $18.8\%$ reduction in total trips taken, compared to the 2010 NPB schedule.

Naturally, there are additional factors involved with the actual scheduling of NPB games at these home stadiums. For example, one of the ballparks hosts a three-day concert each August, and thus must play away games on those particular dates. Sometimes a league has two outstanding teams that are bitter rivals, hence officials will deliberately schedule a match between them to conclude the season, to add drama and boost TV ratings. These constraints must be taken into account when producing an optimal schedule that can be implemented by NPB, to ensure no conflicts occur, and that the schedule is best possible for all parties involved.

## Future Research

We have developed a rigorous algorithm to solve the Multi-Round Balanced Traveling Tournament Problem (mb-TTP) for the case $n = 6$ and applied it to develop optimal intra-league schedules for the two six-team leagues in Nippon Professional Baseball, showing how both leagues can reduce their total traveling distance by over $25\%$. While our algorithm can be applied to any six-team league, our approach is unlikely to be computationally feasible for $n \geq 8$. Even for the $n = 8$ case, we have $m = \binom{n}{n/2}^2 \cdot (\frac{n}{2})! = 117600$, requiring us to store the edge weights of digraph $G$ in two $117600 \times 117600$ matrices. Hence, we require more sophisticated techniques for solving the mb-TTP for $n \geq 8$. Since the $8$-team benchmark set NL8 was recently solved using a depth-first-search with upper bounding (Uthus, Riddle, and Guesgen 2009) as well as a branch-and-price approach (Irnich 2010), perhaps these powerful methods could be applied in conjunction with our Dijkstra formulation to solve the mb-TTP for $n = 8$.

Also a natural question is to extend the standard TTP to multiple rounds, eliminating the *each-round* and *diff-two* conditions of our balanced framework. However, this problem may be quite difficult as the *each-round* condition enabled us to apply the theory of perfect matchings and one-factorizations to enumerate our feasible timetables, and the *diff-two* condition allowed us to reduce our concatenation matrix to just four columns. Without the *diff-two* condition, we would require six columns rather than four, since we need to check cases when one team ends a block with three consecutive home (away) games or starts the next block with three consecutive home (away) games, to validate the *at-most-three* condition. To apply the same shortest path algorithm for the multi-round standard TTP would require us to set $m = \binom{n}{n/2}^3 \cdot (\frac{n}{2})!$, which requires $m = 48000$ for $n = 6$. Once again, this approach is not likely to be computationally feasible given the size of the $m \times m$ matrices, and therefore more sophisticated techniques must be developed.

## References


Dijkstra, E. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1:269–271.

Easton, K.; Nemhauser, G.; and Trick, M. 2001. The traveling tournament problem: description and benchmarks. *Proceedings of the 7th International Conference on Principles and Practice of Constraint Programming* 580–584.

Hoshino, R., and Kawarabayashi, K. 2010. A multi-round generalization of the traveling tournament problem and its application to Japanese baseball. *European Journal of Operational Research* (submitted).

Hoshino, R., and Kawarabayashi, K. 2011. The inter-league extension of the traveling tournament problem and its application to sports scheduling. *AAAI Conference on Artificial Intelligence* (submitted).

Irnich, S. 2010. A new branch-and-price algorithm for the traveling tournament problem. *European Journal of Operational Research* 204:218–228.

Kendall, G.; Knust, S.; Ribeiro, C.; and Urrutia, S. 2010. Scheduling in sports: An annotated bibliography. *Computers and Operations Research* 37:1–19.

Rasmussen, P., and Trick, M. 2007. A Benders approach for the constrained minimum break problem. *European Journal of Operational Research* 177:198–213.

Rasmussen, P., and Trick, M. 2008. Round robin scheduling - a survey. *European Journal of Operational Research* 188:617–636.

Ribeiro, C., and Urrutia, S. 2004. Heuristics for the mirrored traveling tournament problem. *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling* 323–342.

Thielen, C., and Westphal, S. 2010. Complexity of the traveling tournament problem. *Theoretical Computer Science* DOI:10.1016/j.tcs.2010.10.001.

Uthus, D.; Riddle, P.; and Guesgen, H. 2009. DFS* and the traveling tournament problem. *Proceedings of the 6th International Conference on the Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems CPAIOR 2009* 279–293.