The Traveling Tournament Problem with Maximum Tour Length Two: A Practical Algorithm with An Improved Approximation Bound

Jingyang Zhao and **Mingyu Xiao***
University of Electronic Science and Technology of China
1176033045@qq.com, myxiao@gmail.com

Abstract

The Traveling Tournament Problem is a well-known benchmark problem in tournament timetabling, which asks us to design a schedule of home/away games of n teams (n is even) under some feasibility requirements such that the total traveling distance of all the n teams is minimized. In this paper, we study TTP-2, the traveling tournament problem where at most two consecutive home games or away games are allowed, and give an effective algorithm for $n/2$ being odd. Experiments on the well-known benchmark sets show that we can beat previously known solutions for all instances with $n/2$ being odd by an average improvement of 2.66%. Furthermore, we improve the theoretical approximation ratio from $3/2 + O(1/n)$ to $1 + O(1/n)$ for $n/2$ being odd, answering a challenging open problem in this area.

1 Introduction

The Traveling Tournament Problem (TTP), an interesting sports scheduling problem inspired by Major League Baseball, was first systematically introduced in [Easton *et al.*, 2001], and then followed in a large amount of references [Kendall *et al.*, 2010; Rasmussen and Trick, 2008; Thielen and Westphal, 2012; Xiao and Kou, 2016]. This problem is to find a double round-robin tournament satisfying some constraints that minimizes the total distances traveled by all participant teams. In a double round-robin tournament of n teams, each team will play 2 games with each of the other $n-1$ teams, one at its home venue and one at its opponent's home venue. Furthermore, all the games should be scheduled in $2(n-1)$ consecutive days. Such all terms will play on each day and there are exactly $n/2$ games on each day. According to the definition, we know that n is always even. For TTP, we have the following two basic constraints or assumptions on the double round-robin tournament.

- *no-repeat*: A team cannot play against the same opponent in two consecutive games.
- *direct-traveling*: A team travels directly from its game venue in the ith day to its game venue in the $(i+1)$th day, where we assume that all teams are at home in the 0th day and the $(2n-1)$th day, i.e., all teams are initially at home and will come back to home after all the games.

A frequently studied version of TTP, denoted by TTP-k, is to add the following constraint on the maximum number of consecutive home games and away games.

- *bounded-by-k*: No team have a home stand or a road trip lasting more than k games.

The smaller the value of k, the more often teams have to return to their home venues. The input of TTP or TTP-k contains an $n \times n$ distance matrix D to indicate the distance between each pair of the n teams. We will use $D_{i,j}$ to denote the distance from the home of team i to the home of team j. We also assume that D satisfies the symmetry and triangle inequality properties, i.e.,$D_{i,j} = D_{j,i}$ and $D_{i,j} \leq D_{i,h} + D_{h,j}$ for all i, j, h. We let $D_{i,i} = 0$ for each i.

1.1 Related Work

TTP and TTP-k are difficult optimization problems. The NP-hardness of TTP and TTP-3 was proved in [Bhattacharyya, 2016; Thielen and Westphal, 2011]. Since the search space of TTP and TTP-k are usually very large, many instances with more than 10 teams in the online benchmark [Trick, 2021] have not been completely solved even by using high-performance machines. In the literature there is a large number of contributions on approximation algorithms [Yamaguchi *et al.*, 2011; Imahori *et al.*, 2014; Miyashiro *et al.*, 2012; Westphal and Noparlik, 2014; Hoshino and Kawarabayashi, 2013; Thielen and Westphal, 2012; Xiao and Kou, 2016] and heuristic algorithms [Easton *et al.*, 2003; Lim *et al.*, 2006; Anagnostopoulos *et al.*, 2006; Di Gaspero and Schaerf, 2007; Goerigk *et al.*, 2014].

In this paper, we will focus on TTP-2. In a sports schedule, it is generally believed that home stands and road trips should alternate as regularly as possible for each team [Campbell and Chen, 1976; Thielen and Westphal, 2012]. However, there is no feasible schedule for $k = 1$ [de Werra, 1988], and then TTP-2 becomes especially interesting. The first record of TTP-2 seems from the schedule of a basketball conference of ten teams in [Campbell and Chen, 1976]. It is easy to verify that any feasible schedule for TTP-2 is a 2-approximation solution [Thielen and Westphal, 2012].

*Contact Author

So any feasible solution will not have a very bad performance. However, we do not know any simple construction of feasible solutions. Even the scheduling algorithms for $n/2$ being even and odd may be different. Thielen and Westphal [2012] proposed two scheduling methods and proved an approximation ratio of $3/2 + O(1/n)$ for $n/2$ being odd and an approximation ratio of $1 + O(1/n)$ for $n/2$ being even. The current best approximation ratio for $n/2$ being even is $(1 + 4/n)$ by Xiao and Kou [2016]. Whether TTP-2 with $n/2$ being odd allows an approximation ratio $1 + O(1/n)$ became a challenging problem [Thielen and Westphal, 2012; Xiao and Kou, 2016].

1.2 Our Results

In this paper, we design an effective algorithm for TTP-2 with $n/2$ being odd. In theory, we prove an approximation ratio of $(1 + \frac{8}{n} + \frac{4}{n-2})$, significantly improving the previous result of $(3/2 + \frac{6}{n-4})$. This is the first $(1 + O(1/n))$-approximation algorithm known so far and it answers the open problem in [Thielen and Westphal, 2012] positively. In practice, our algorithm is easy to implement and runs very fast. Experiments show that our algorithm can solve all the 16 tested instances in the benchmark [Trick, 2021] with $n \geq 10$ and $n \equiv 2 \pmod 4$ very quickly by providing better solutions with an average improvement of 2.66%.

2 Notations

We always use n to denote the number of teams in the problem. In this paper, we assume that $n \equiv 2 \pmod 4$ and $n \geq 10$. We let $m = n/2$. Then m is also an integer. We use G to denote the complete graph on the n vertices representing the n teams, and the weight of the edge between two vertices t_i and t_j is $D_{i,j}$ the distance from the home of t_i to the home of t_j. We also use D_i to denote the weight sum of all edges incident on t_i in G, i.e., $D_i = \sum_{j=1}^{n} D_{i,j}$. The sum of all edge weights of G is denoted by D_G. We let M denote a minimum perfect matching in G. The weight sum of all edges in M is denoted by D_M. We may consider the endpoint pair of each edge in M as a *super-team*. We use H to denote the complete graph on the m vertices representing the m super-teams. The weight of the edge between two super-teams u_i and u_j, denoted by $D(u_i, u_j)$, is the sum of the weight of the four edges in G between one team in u_i and one team in u_j, i.e., $D(u_i, u_j) = \sum_{t_{i'} \in u_i \& t_{j'} \in u_j} D_{i',j'}$. We also let $D(u_i, u_i) = 0$ for any i. We give an illustration of the graphs G and H in Figure 1.

In TTP and TTP-k problems, each team t_i will visit each other team once in the tournament. For each team, a routing visiting each other team exactly once (starting at its home and coming back home last) is called an *itinerary* of the team. A *road trip* in an itinerary of team t_i is a simple cycle starting and ending at t_i. So an itinerary consists of several road trips. For TTP-2, an itinerary is *feasible* if each road trip of it is a cycle of length at most 3 (containing at most two other teams).

For two teams t_i and t_j, we may use a directed edge from t_i to t_j to denote a game between t_i and t_j taking place at the home of t_j.

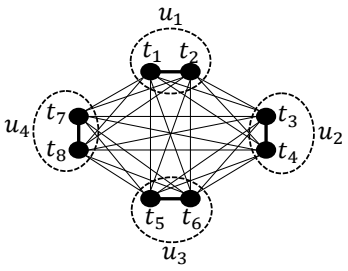


Figure 1: An illustration of graphs G and H, where the four dark lines form a minimum perfect matching M in G

3 Simple Lower and Upper Bounds

3.1 A Lower Bound

First we consider a lower bound on the total traveling distance of a single team. The bound is called the *independent lower bound* and was firstly introduced by Campbell and Chen [1976]. Here 'independent' means we only consider the traveling of the single team and do not consider the feasibility of other teams. In the itinerary of one team, if there are two away trips of a single game, then we may be able to combine the two away trips into one away trip of a pair of games without increasing the traveling distance by the triangle inequality. Note that n is even and the optimal itinerary of a team must contain exactly one away trip of a single game (other away trips contain two games). The optimal itinerary graph of a team t_i may look like the graph in Figure 2.

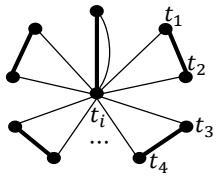


Figure 2: the itinerary graph of team t_i

Team t_i must travel to or from each other team at least once, which are denoted by light lines in Figure 2. The weight sum of all light lines is $\sum_{j \neq i} D_{i,j} = D_i$. The remaining dark lines in Figure 2 form a perfect matching of the graph G. Recall that we use M to denote a minimum perfect matching of G. Then the weight sum of all dark lines is at most D_M. The *independent lower bound* for team t_i is

$$LB_i = D_i + D_M. \tag{1}$$

The independent lower bound for TTP-2 is

$$LB = \sum_{i=1}^{n} LB_i = \sum_{i=1}^{n} (D_i + D_M) = 2D_G + nD_M. \tag{2}$$

For any team, it is possible to get its independent lower bound. The itinerary of a team is called *perfect* if the distance of it achieves the independent lower bound. However, not all teams may reach perfect itineraries synchronously in a feasible schedule [Thielen and Westphal, 2012]. So the independent lower bound for TTP-2 is not achievable.

3.2 An Upper Bound

Next, we consider a simple upper bound on the traveling distance of any feasible solution. By the triangle inequality, we can easily see that

Lemma 1. *The traveling distance of any itinerary of a team t_i is at most $2D_i$.*

Lemma 1 says that the worst itinerary consisting of only road trips of one game. Lemma 1 and (1) imply that

Lemma 2. *The traveling distance of any feasible itinerary of a team t_i is at most the traveling distance of the optimal itinerary plus D_i.*

Note that $D_i \leq LB_i$. We get that

Theorem 1. *Any feasible schedule for TTP-2 is a 2-approximation solution.*

Indeed, Theorem 1 was first proved in [Thielen and Westphal, 2012]. We repeat this result in detail because we need to use Lemma 2 in our analysis.

4 Construction of The Tournament

To get a good theory bound and good performance, we will try to design a feasible schedule such that the itinerary of each team is similar to the perfect itinerary as much as possible.

It is not easy to find a simple feasible schedule for TTP-2. Compared with known feasible schedules, our schedule is not very complicated. However, our schedule will be slightly different for $n \equiv 2 \pmod 8$ and $n \equiv 6 \pmod 8$. We will describe the algorithm for the case of $n \equiv 2 \pmod 8$. For the case of $n \equiv 6 \pmod 8$, only some edges will have different directions. These edges will be denoted by dash lines and we will explain them.

We regard each pair of teams in the minimum perfect matching M of G as a *super-team*, and first arrange games between super-teams. There are n teams and then there are m super-teams. We denote the super-teams as $\{u_1, u_2, \ldots, u_{m-1}, u_m\}$. We relabel the n teams such that $u_i = \{t_{2i-1}, t_{2i}\}$ for each i. For the sake of presentation, we will also denote $u_l = u_{m-1}$ (resp., $u_r = u_m$), and denote the two teams in u_l as $\{t_{l1}, t_{l2}\}$ (resp., the two teams in u_r as $\{t_{r1}, t_{r2}\}$).

We first arrange *super-games* between super-teams u_i and then we will extend super-games to normal games between normal teams t_i.

Each super-team will attend $m-1$ super-games in $m-1$ time slots. Each super-game in the first $m-2$ time slots will be extended to four normal games between normal teams, and each super-game in the last time slot will be extended to six normal games between normal teams. So each normal team t_i will attend $4 \times (m-2)+6 = 4m-2 = 2n-2$ games. This is the number of games each team t_i should attend in TTP-2.

We first design the super-games between super-teams in the first $m-2$ time slots, and then consider super-games in the last time slot. In each of the first $m-2$ time slots, we have $\frac{m-1}{2}$ super-games (note that m is odd), where one super-game involves three super-teams and all other super-games involve two super-teams. In the first slot, the $\frac{m-1}{2}$ super-games are arranged as shown in Figure 3. The most left super-game involving super-team u_l is called the *left super-game*, and the most right super-game involving u_r is called the *right super-game*. The right super-game is the only super-game involving three super-teams. The other $\frac{m-1}{2}-2$ super-games are called *middle super-games*. There are also directed edges in the super-games, which will be used to extend super-games to normal games.

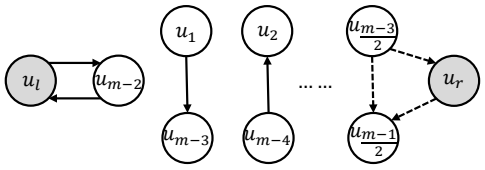


Figure 3: Schedule at the first slot

Note that the white nodes (super-teams $u_1, \ldots, u_{m-2}$) in Figure 3 form a cycle $u_1 u_2 \ldots u_{m-2}$. In the second slot, super-games are scheduled as shown in Figure 4. we change the positions of white super-teams in the cycle by moving one position in the clockwise direction, and also change the direction of each edge. The positions of u_l and u_r will always be fixed.

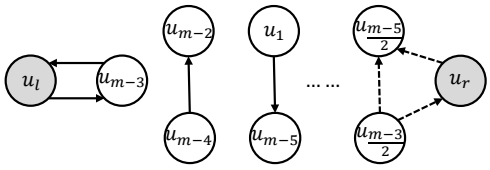


Figure 4: Schedule at the Second slot

The schedules for the first $m-2$ slots are derived analogously. Next, we explain how to extend the super-games in these slots to normal games.

Case 1: Middle super-games. We first consider middle super-games, each of which will be extended to four normal games in four days. Assume that in a middle super-game, team u_i plays against the team u_j on time slot q, $1 \leq i, j, q \leq m-2$. Recall that u_i represents normal teams $\{t_{2i-1}, t_{2i}\}$ and u_j represents normal teams $\{t_{2j-1}, t_{2j}\}$. The super-game will be extended to eight normal games in four corresponding days from $4q-3$ to $4q$, as shown in Figure 5. A directed edge from team $t_{i'}$ to team $t_{i''}$ means $t_{i'}$ plays against $t_{i''}$ at the home of $t_{i''}$. Note that if there is a directed edge from u_j to u_i, then the direction of all the edges in Figure 5 should be reversed.

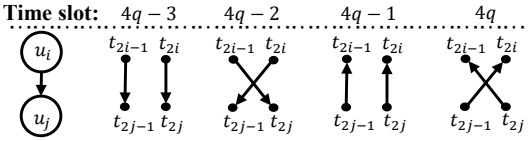


Figure 5: Middle super-games

Case 2: Left super-games. Assume that in a left super-game, team u_l plays against team u_i on time slot q, $1 \leq i, q \leq m-2$. Recall that u_l represents normal teams

$\{t_{2m-3}, t_{2m-2}\}$ and u_i represents normal teams $\{t_{2i-1}, t_{2i}\}$. The super-game will be extended to eight normal games in four corresponding days from $4q-3$ to $4q$, as shown in Figure 6 for odd time slot q. For even time slot q, the direction of edges in the figure will be reversed.

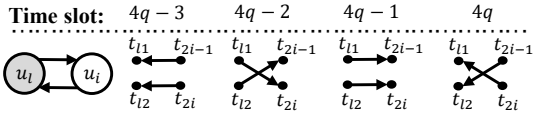


Figure 6: Left super-games

Case 3: Right super-games. Assume that in a right super-game, three teams u_i, u_{i-1} and u_r play on time slot q, $1 \leq i, q \leq m-2$ and $u_0 = u_{m-2}$. Recall that u_r represents normal teams $\{t_{2m-1}, t_{2m}\}$. The super-game will be extended to twelve normal games in four corresponding days from $4q-3$ to $4q$, as shown in Figure 7. Note that the edges between three teams are dash edges (see Figures 3 and 4). So for the case of $n \equiv 6 \pmod 8$, the direction of these three edges should be reversed. So Figure 7 describes the case of odd time slot q with $n \equiv 2 \pmod 8$ and the case of even time slot q with $n \equiv 6 \pmod 8$. For the case of odd time slot q with $n \equiv 2 \pmod 8$ and the case of even time slot q with $n \equiv 6 \pmod 8$, the direction of all edges in the figure will be reversed.

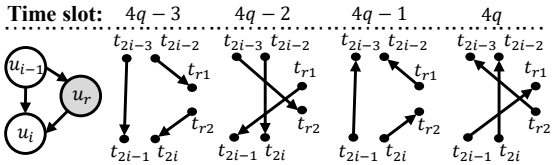


Figure 7: Right super-games

The first $m-2$ slots will be extended to $4(m-2) = 2n-8$ days according to the above rules. Each team will have six remaining games, which will be scheduled in the last time slot. Two remaining games are *self-games*, which is to play against the team in the same super-team. For teams in u_l and u_r, the other four remaining games are between the teams in u_l and u_r. For teams in the other super-teams u_i, two remaining games should be played against a team in super-team u_{i-1} and two remaining games should be played against a team in super-team u_{i+1}, such as the two missing games between t_{2i-3} and t_{2i} in Figure 7. Figure 8 shows the six remaining games for teams in u_i ($i \in \{1, 2, \ldots, m-2\}$), and Figure 9 shows the six remaining games for teams in u_l and u_r.

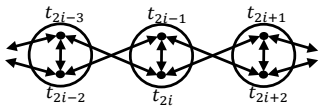


Figure 8: The six remaining games for teams in u_i ($i \neq l, r$)

The last slot. Now we are ready to design the six days of games for the last slot. We use $self$ to denote one day of self-games: for each super-team u_i, team t_{2i-1} plays against team t_{2i} at the home of t_{2i}. We will also use $\overline{self}$ to denote the day of games with the reversed direction in $self$, i.e., the partner is the same but the game site change to the other team's home. We also design two days of games A_1 and A_2 as shown in Figures 10 and 11. Then we can see that the other four days of games left are given by $A_1, A_2, \overline{A_1}, \overline{A_2}$. Note that in $A_1, A_2, \overline{A_1}$, and $\overline{A_2}$ we will have three dash edges, which will be reversed for the case of $n \equiv 6 \pmod 8$.

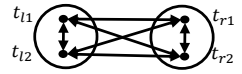


Figure 9: The six remaining games for teams in u_l and u_r

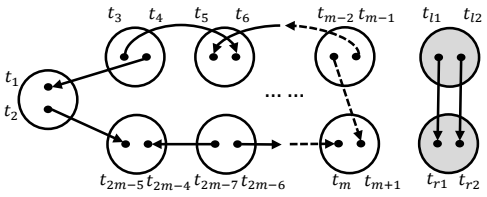


Figure 10: The day A_1

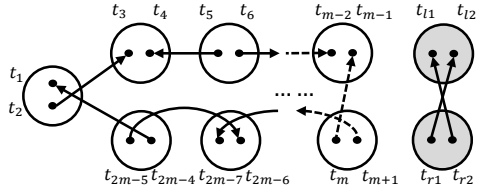


Figure 11: The day A_2

Next, we need to order the six days $\{self, \overline{self}, A_1, A_2, \overline{A_1}, \overline{A_2}\}$ to joint the previous $2n-8$ days without violating the *bounded-by-k* and *no-repeat* constraints. It will be different for the two cases of $n \equiv 2 \pmod 8$ and $n \equiv 6 \pmod 8$. For $n \equiv 2 \pmod 8$, the six days are arranged in the order: $A_1, self, A_2, \overline{self}, \overline{A_2}, \overline{A_1}$; For $n \equiv 6 \pmod 8$, the six days are arranged in the order: $\overline{A_2}, \overline{self}, \overline{A_1}, self, A_1, A_2$.

We have described the main part of the scheduling algorithm. Next, we will prove its feasibility.

Theorem 2. *For TTP-2 with n teams such that $n \geq 10$ and $n \equiv 2 \pmod 4$, the above construction can generate a feasible schedule.*

Proof. First, we show that each team plays all the required $2n-2$ games in the $2n-2$ days. According to the schedule, we can see that each team will attend one game in each of the $2n-2$ days. Furthermore, it is not hard to observe that no two teams play against each other at the same place. So each team will play the required $2n-2$ games.

Second, it is easy to see that each team will not violate the *no-repeat* property. In any time slot, no two games between the same teams are arranged in two consecutive days. For two different time slots, each team will play against different teams. Especially, $self$ and $\overline{self}$ are not arranged as two consecutive days in the last time slot.

Last, we prove that each team does not violate the *bounded-by-k* property. We will use 'H' and 'A' to denote a home game and an away game, respectively. We will also let $\overline{H} = A$ and $\overline{A} = H$.

We first look at the games in the first $2n - 8$ days. For the two teams in u_l, the 4 games in an odd time slot will be $HAAH$ (see Figure 6), and the 4 games in an even time slot will be $AHHA$. So two consecutive time slots can be jointed well. For the two teams in u_r, the 4 games will be $HAAH$ or $\overline{HAAH}$ in an odd time slot and reversed in an even time slot. Two consecutive time slots can still be jointed well. Next, we consider a team t_i in u_j ($j \in \{1, 2, \ldots, m-2\}$). In the time slots for middle super-games, the 4 games will be $AAHH$ if the direction of the edge (super-game) is toward u_j and $\overline{AAHH}$ otherwise. In the time slots for left super-games, the 4 games will be $AHHA$ or $\overline{AHHA}$. In the time slots for right super-games, the 4 games will be $AAHH$ or $\overline{AAHH}$. According to our schedule, two consecutive time slots can joint well, no matter they are two slots for middle super-games, or one slot for middle super-game and one slot for left super-game, or one slot for middle super-game and one slot for right super-game.

We have the last 6 days in the last time slot not analyzed yet. For the sake of presentation, we just list out the last 10 games in the last two time slots for each team. We will have five different cases: teams in u_l, u_r, u_1, u_0 for odd $o \in \{3, \ldots, m-2\}$, and u_e for even $e \in \{2, \ldots, m-2\}$. The reason why u_1 is different from other u_o with an odd subscript o is that in the penultimate time slot, u_1 plays with u_l, and then it has a different form of games. For the case of $n \equiv 2 \pmod 8$, the last 10 games are shown in Figure 12, and for the case of $n \equiv 6 \pmod 8$, the last 10 games are shown in Figure 13. We can see that there are no three consecutive home games or away games. So the *bounded-by-k* property holds.

$$\begin{cases} u_l \begin{cases} t_{l1}\text{: } HAAH\ AAHHAH \\ t_{l2}\text{: } HAAH\ AHHAAH \end{cases} \\ u_r \begin{cases} t_{r1}\text{: } HAAH\ HAAHHA \\ t_{r2}\text{: } AHHA\ HHAAHA \end{cases} \end{cases} \quad \begin{cases} u_1 \begin{cases} t_1\text{: } AHHA\ HAHHAA \\ t_2\text{: } AHHA\ AHAAHH \end{cases} \\ u_o \begin{cases} t_{o1}\text{: } HHAA\ HAAHHA \\ t_{o2}\text{: } HHAA\ HHAAHA \end{cases} \\ u_e \begin{cases} t_{e1}\text{: } AAHH\ AAHHAH \\ t_{e2}\text{: } AAHH\ AHHAAH \end{cases} \end{cases}$$

Figure 12: The last 10 games for the case of $n \equiv 2 \pmod 8$

$$\begin{cases} u_l \begin{cases} t_{l1}\text{: } HAAH\ AHHAAH \\ t_{l2}\text{: } HAAH\ AAHHAH \end{cases} \\ u_r \begin{cases} t_{r1}\text{: } AHHA\ HHAAHA \\ t_{r2}\text{: } HAAH\ HAAHHA \end{cases} \end{cases} \quad \begin{cases} u_1 \begin{cases} t_1\text{: } AHHA\ AHAAHH \\ t_2\text{: } AHHA\ HAHHAA \end{cases} \\ u_o \begin{cases} t_{o1}\text{: } HHAA\ HHAAHA \\ t_{o2}\text{: } HHAA\ HAAHHA \end{cases} \\ u_e \begin{cases} t_{e1}\text{: } AAHH\ AHHAAH \\ t_{e2}\text{: } AAHH\ AAHHAH \end{cases} \end{cases}$$

Figure 13: The last 10 games for the case of $n \equiv 6 \pmod 8$

Since our schedule satisfies the above three conditions, we know that our schedule is a feasible schedule for TTP-2. Note that our schedule requires at least 10 teams. So we have $n \geq 10$. □

5 Analyzing Approximation Quality

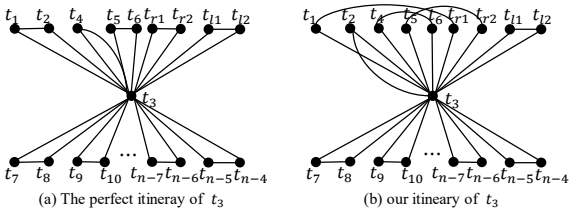
Next, we analyze the theoretical approximation bound of our schedule. We compare the itinerary of each team in our schedule with the perfect itinerary and use Δ_i to denote the difference between them for each team i. Then we only need to evaluate a bound for $\sum_{i=1}^{n} \Delta_i$.

We first introduce the framework of our proof idea. For team t_i with $1 \leq i \leq n-4$ in a white super-team u_j in Figure 3, only its trips to the nine teams in $u_l \cup u_r \cup u_{j-1} \cup u_j \cup u_{j+1} \setminus \{t_i\}$ may be different from that in the perfect itinerary. By Lemma 2, we know that the distance of this part in our schedule will be at most that in the perfect itinerary plus the nine distances from t_i to the nine teams. On average, each team t_i will use an extra distance of about $9/n$ of the independent low bound LB_i. For teams t_i ($i \in \{1, 2, \ldots, n-4\}$), the total extra distance will be bounded by about $9/n$ of LB. For the other four teams t_i ($i \in \{n-3, n-2, n-1, n\}$) in the two super-teams u_l and u_r, their itineraries may be totally different from the perfect itineraries. However, by Lemma 2, we know that the extra distance for t_i is most LB_i. We will set the four teams with the minimum value LB_i. Then the extra distance given by the four teams is at most $4/n$ of LB. Thus, we get an approximation ratio of about $(1 + 13/n)$. Next, we give a more detailed analysis and show a refined ratio of $(1 + 12/n)$ indeed.

First, we consider the four teams $\{t_{n-3}, t_{n-2}, t_{n-1}, t_n\}$ in the two super-teams u_l and u_r. By Lemma 2, we directly get that

$$\Delta_i \leq LB_i \quad \text{for each } i \in \{n-3, n-2, n-1, n\}. \tag{3}$$

Second, we consider the $n-4$ teams t_i in super-teams u_j ($j \in \{1, 2, \ldots, m-2\}$). We look at two super-teams u_{i_1} and u_{i_2}, where $1 \leq i_1, i_2 \leq m-2$ and $|i_1 - i_2| \geq 2$. The games between the two super-teams will be the form of middle super-games in Case 1. A team in u_{i_1} (resp., u_{i_2}) will visit the two teams in u_{i_2} (resp., u_{i_1}) in one road trip, which is also a road trip in the perfect itinerary. Comparing with the perfect itinerary for a team $t_i \in u_j$ ($i \in \{1, 2, \ldots, n-4\}$, and $j = \lceil i/2 \rceil$), only the trips to visit the nine teams in $u_l \cup u_r \cup u_{j-1} \cup u_j \cup u_{j+1} \setminus \{t_i\}$ may be different. See the comparison for t_3 as an example in Figure 14.



(a) The perfect itineray of t_3 (b) our itineary of t_3

Figure 14: The perfect itinerary and our itinerary of t_3: the lower parts are the same and the upper parts of the trips to the nine teams are different

By Lemma 2, we know that

Claim. *For each $t_i \in u_j$ ($i \in \{1, 2, \ldots, n-4\}$, and $j = \lceil i/2 \rceil$), Δ_i is bounded by the sum of the distances from t_i to the nine teams in $u_l \cup u_r \cup u_{j-1} \cup u_j \cup u_{j+1} \setminus \{t_i\}$.*

Recall that for two super-games u_i and u_j, we use $D(u_i, u_j)$ to denote the sum of the four distances between teams $t_p \in u_i$ and $t_q \in u_j$. By the above claim, we get that

$$\sum_{i=1}^{n-4} \Delta_i \leq \sum_{i=n-3}^{n} LB_i + 2\sum_{i=1}^{m-2} D(u_{i-1}, u_i), \tag{4}$$

where u_0 is interpreted as u_{m-2}.

Next, we reorder the teams to get good bounds for $B_1 = \sum_{i=n-3}^{n} LB_i$ and $B_2 = \sum_{i=1}^{m-2} D(u_{i-1}, u_i)$. Since $LB = \sum_{i=1}^{m}(LB_{2i-1} + LB_{2i})$, we can choose u_l and u_r such that

$$B_1 = \sum_{i=n-3}^{n} LB_i \leq \frac{2}{m} LB = \frac{4}{n} LB. \tag{5}$$

Next, we choose an order of u_i to make B_2 as small as possible. Recall that H is the complete graph on the m super-teams u_i. We decompose the complete graph H into $(m-1)/2$ edge-disjoint Hamilton cycles by using the well-known decomposition algorithm in [Alspach *et al.*, 1990] and let C be one Hamilton cycle among them having the minimum length. Then we reorder u_i such that $u_1 u_2 \dots u_{m-2}$ is a part of the Hamilton cycle C. Therefore, by the triangle inequality, we get

$$B_2 = \sum_{i=1}^{m-2} D(u_{i-1}, u_i) \leq \frac{2}{m-1} D_H \leq \frac{2}{n-2} LB. \tag{6}$$

By (3)-(6), we get that

$$\begin{aligned} \sum_{i=1}^{n} \Delta_i &\leq 2\sum_{i=n-3}^{n} LB_i + 2\sum_{i=1}^{m-2} D(u_{i-1}, u_i) \\ &= 2B_1 + 2B_2 \\ &\leq (\frac{8}{n} + \frac{4}{n-2}) LB. \end{aligned}$$

Theorem 3. *There is a polynomial-time* $(1 + \frac{8}{n} + \frac{4}{n-2})$*-approximation algorithm for TTP-2 with* n *teams, where* $n \geq 10$ *and* $n \equiv 2 \pmod 4$.

6 Experimentations

To test the experimental performance, we will also use one more local-search trick. Note that after getting a feasible schedule by the above algorithm, we can get more feasible schedules by only changing the positions of the teams. In the experiments, we use two local-search rules to check better solutions: swap between two super-teams and swap between two teams in each super-team. This will lead $O(m^3)$ loops, and then the running-time bound will increase a factor of m^3.

We implement our algorithm to solve the benchmark instances in [Trick, 2021]. The website introduces 62 instances, most of which were reported from real-world sports scheduling scenarios, such as the Super 14 Rugby League, the National Football League, and the 2003 Brazilian soccer championship. The number of teams in the instances varies from 4 to 40. We can solve all the 23 instances of n teams with $n \geq 10$ and $n \equiv 2 \pmod 4$, in which 7 instances are very special (all teams are in a cycle or all distance between pairs of teams are 1) and not tested in previous papers. For the remaining 16 instances, we compare our results with the best-known results in Table 1. In the table, the column '*Lower Bound*' indicates the independent lower bounds, '*Previous Results*' lists previous known results in [Thielen and Westphal, 2012], '*Our Results*' shows the results given by our schedule algorithm after local search, and '*Improvement Ratio*' is defined as $\frac{Previous\ Results - Our\ Results}{Previous\ Results}$.

Data Sets	Lower Bounds	Previous Results	Our Results	Improvement Ratio(%)
Galaxy38	244848	274672	262657	4.37
Galaxy34	173312	192317	185347	3.62
Galaxy30	113818	124011	122628	1.12
Galaxy26	68826	77082	75538	2.00
Galaxy22	40528	46451	45360	2.35
Galaxy18	23774	27967	27467	1.79
Galaxy14	12950	15642	15622	0.13
Galaxy10	5280	6579	6506	1.11
NFL30	951608	1081969	1038902	3.98
NFL26	669782	779895	745595	4.40
NFL22	504512	600822	574028	4.46
NFL18	361204	439152	423326	3.60
NL14	238796	296403	289323	2.39
NL10	70866	90254	87842	2.67
Super14	823778	1087749	1019941	6.23
Super10	392774	579862	589512	-1.66

Table 1: Experimental results

From this table, we can see that our scheduling improves 15 out of 16 instances. The average improvement on the 16 instances is 2.66%. Our algorithm is also fast. On a standard laptop with a 2.30GHz Intel(R) Core(TM) i5-6200 CPU and 8 GB RAM, all the 16 instances can be solved together within 0.1 seconds before applying local search and within 30 seconds including local search.

7 Conclusion

We have designed an algorithm to generate a feasible solution to TTP-2 with $n \equiv 2 \pmod 4$, which can guarantee the traveling distance at most $(1 + \frac{8}{n} + \frac{4}{n-2})$ times of the optimal. This is the first $(1 + O(\frac{1}{n}))$-approximation algorithm for TTP-2 with $n \equiv 2 \pmod 4$. The algorithm is also very practical. Experimental results show that our algorithm can beat best-known solutions for almost all instances with $n \geq 10$ and $n \equiv 2 \pmod 4$ in the well-known benchmark. Furthermore, we are able to get further improvements by using a simple local-search method. In the experiments, we only consider exchanges between two teams (or super-teams) each time. We may be able to get more improvements by considering exchanges among three or more teams. However, the running time will increase dramatically and the improvement is very limited.

Acknowledgments

The work is supported by the National Natural Science Foundation of China, under grants 61972070.

References

[Alspach *et al.*, 1990] Brian Alspach, J-C Bermond, and Dominique Sotteau. Decomposition into cycles i: Hamilton decompositions. In *Cycles and rays*, pages 9–18. Springer, 1990.

[Anagnostopoulos *et al.*, 2006] Aris Anagnostopoulos, Laurent Michel, Pascal Van Hentenryck, and Yannis Vergados. A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, 9(2):177–193, 2006.

[Bhattacharyya, 2016] Rishiraj Bhattacharyya. Complexity of the unconstrained traveling tournament problem. *Operations Research Letters*, 44(5):649–654, 2016.

[Campbell and Chen, 1976] Robert Thomas Campbell and Der-San Chen. A minimum distance basketball scheduling problem. *Management science in sports*, 4:15–26, 1976.

[de Werra, 1988] Dominique de Werra. Some models of graphs for scheduling sports competitions. *Discrete Applied Mathematics*, 21(1):47–65, 1988.

[Di Gaspero and Schaerf, 2007] Luca Di Gaspero and Andrea Schaerf. A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, 13(2):189–207, 2007.

[Easton *et al.*, 2001] Kelly Easton, George Nemhauser, and Michael Trick. The traveling tournament problem: description and benchmarks. In *7th International Conference on Principles and Practice of Constraint Programming*, pages 580–584, 2001.

[Easton *et al.*, 2003] Kelly Easton, George Nemhauser, and Michael Trick. Solving the travelling tournament problem: a combined integer programming and constraint programming approach. In *4th International Conference of Practice and Theory of Automated Timetabling IV*, pages 100–109, 2003.

[Goerigk *et al.*, 2014] Marc Goerigk, Richard Hoshino, Ken Kawarabayashi, and Stephan Westphal. Solving the traveling tournament problem by packing three-vertex paths. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 2271–2277, 2014.

[Hoshino and Kawarabayashi, 2013] Richard Hoshino and Ken-ichi Kawarabayashi. An approximation algorithm for the bipartite traveling tournament problem. *Mathematics of Operations Research*, 38(4):720–728, 2013.

[Imahori *et al.*, 2014] Shinji Imahori, Tomomi Matsui, and Ryuhei Miyashiro. A 2.75-approximation algorithm for the unconstrained traveling tournament problem. *Annals of Operations Research*, 218(1):237–247, 2014.

[Kendall *et al.*, 2010] Graham Kendall, Sigrid Knust, Celso C Ribeiro, and Sebastián Urrutia. Scheduling in sports: An annotated bibliography. *Computers & Operations Research*, 37(1):1–19, 2010.

[Lim *et al.*, 2006] Andrew Lim, Brian Rodrigues, and Xingwen Zhang. A simulated annealing and hill-climbing algorithm for the traveling tournament problem. *European Journal of Operational Research*, 174(3):1459–1478, 2006.

[Miyashiro *et al.*, 2012] Ryuhei Miyashiro, Tomomi Matsui, and Shinji Imahori. An approximation algorithm for the traveling tournament problem. *Annals of Operations Research*, 194(1):317–324, 2012.

[Rasmussen and Trick, 2008] Rasmus V Rasmussen and Michael A Trick. Round robin scheduling–a survey. *European Journal of Operational Research*, 188(3):617–636, 2008.

[Thielen and Westphal, 2011] Clemens Thielen and Stephan Westphal. Complexity of the traveling tournament problem. *Theoretical Computer Science*, 412(4):345–351, 2011.

[Thielen and Westphal, 2012] Clemens Thielen and Stephan Westphal. Approximation algorithms for TTP(2). *Mathematical Methods of Operations Research*, 76(1):1–20, 2012.

[Trick, 2021] Michael Trick. Challenge traveling tournament instances. Accessed: 2021-2-20, 2021.

[Westphal and Noparlik, 2014] Stephan Westphal and Karl Noparlik. A 5.875-approximation for the traveling tournament problem. *Annals of Operations Research*, 218(1):347–360, 2014.

[Xiao and Kou, 2016] Mingyu Xiao and Shaowei Kou. An improved approximation algorithm for the traveling tournament problem with maximum trip length two. In Piotr Faliszewski, Anca Muscholl, and Rolf Niedermeier, editors, *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*, volume 58 of *LIPIcs*, pages 89:1–89:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[Yamaguchi *et al.*, 2011] Daisuke Yamaguchi, Shinji Imahori, Ryuhei Miyashiro, and Tomomi Matsui. An improved approximation algorithm for the traveling tournament problem. *Algorithmica*, 61(4):1077–1091, 2011.