Ann Oper Res (2019) 275:223–243
https://doi.org/10.1007/s10479-017-2595-z

PATAT 2016

The sport teams grouping problem

Túlio A. M. Toffolo[1,3] · **Jan Christiaens**[1] · **Frits C. R. Spieksma**[2] · **Greet Vanden Berghe**[1]

Published online: 27 July 2017

Abstract The sport teams grouping problem (STGP) concerns the assignment of sport teams to round-robin tournaments. The objective is to minimize the total travel distance of the participating teams while simultaneously respecting fairness constraints. The STGP is an NP-Hard combinatorial optimization problem highly relevant in practice. This paper investigates the performance of some complimentary optimization approaches to the STGP. Three integer programming formulations are presented and thoroughly analyzed: two compact formulations and another with an exponential number of variables, for which a branch-and-price algorithm is proposed. Additionally, a meta-heuristic method is applied to quickly generate feasible high-quality solutions for a set of real-world instances. By combining the different approaches' results, solutions within 1.7% of the optimum values were produced for all feasible instances. Additionally, to support further research, the considered STGP instances and corresponding solutions files were shared online.

Keywords Sport teams grouping problem · Branch-and-price · Column generation · Decomposition strategies · Integer programming · Meta-heuristic

1 Introduction

Recreational sports competitions require considerable organizational effort. In such competitions, volunteers invest a substantial amount of their time in handling many organizational issues, such as transporting players, refereeing, and setting up fields. Hence, to help volunteers as much as possible, and to ensure a good experience for all participants, it is important to efficiently organize competitions that are as fair as possible.

✉ Túlio A. M. Toffolo
tulio.toffolo@kuleuven.be

[1] Department of Computer Science, CODeS and imec, KU Leuven, Gent, Belgium

[2] Faculty of Economics and Business, ORSTAT, KU Leuven, Leuven, Belgium

[3] Department of Computing, Federal University of Ouro Preto, Ouro Preto, Brazil

A rapidly increasing amount of academic literature is devoted to the organization of sports leagues. Specific problems have been investigated such as the traveling tournament problem (Easton et al. 2001), the traveling umpire problem (Trick and Yildiz 2007, 2011; Trick et al. 2012; Xue et al. 2015; Toffolo et al. 2016), and the organization of round-robin tournaments (Nemhauser and Trick 1998; Briskorn et al. 2010; Goossens and Spieksma 2009, 2012). Such initiatives have resulted in the formation of an international community (Kendall et al. 2010) which investigates optimization problems related to sports timetabling (Ribeiro 2012), thereby generating state of the art algorithms.

Among the many investigated problems, the traveling tournament problem (TTP) has received special attention in the literature. Introduced by Easton et al. (2001), the problem and several variants have been addressed by various researchers around the globe, such as Anagnostopoulos et al. (2006), Di Gaspero and Schaerf (2007), Uthus et al. (2011), Carvalho and Lorena (2012), Januario et al. (2016). Nevertheless, according to Alarcón et al. (2017), only recently solutions produced by an optimization algorithm to the TTP were employed in practice.

Almost all the literature on sport leagues organization relates to professional leagues. Some exceptions are Schönberger et al. (2000, 2004), Knust (2010) and Goossens and Spieksma (2014), who focus on the schedule of non-commercial table-tennis leagues and amateur indoor football leagues. Several particularities are considered, such as limited access to sport facilities and limited availability of sportsmen.

The extensive study of tournament schedules is well justified. When organizing sport leagues, the main optimization challenges usually arise exactly when scheduling games. However, this is not the case for the problem addressed in this paper. Rather than focusing on the game schedule, this paper addresses the distribution of teams within leagues. In fact, differently from the TTP, the sport teams grouping problem (STGP) concerns solely the assignment of teams to round-robin tournaments, wherein the primary objective is to minimize the distance traveled by the teams while simultaneously respecting fairness constraints. Note that visitor teams always go to the game's venue and return home immediately after the match. Hence, the tournament schedule has no influence on the total distance traveled, which is fixed for a given set of teams. Consequently, we cannot build on recent algorithm achievements regarding the aforementioned well-studied sport timetabling problems. However, the STGP shares some interesting characteristics with these problems: they all are unambiguous and therefore easy to formulate.

Prior to the start of each season, clubs enroll one or more teams to participate in leagues. Each team has a location (related to its club) and is assigned a certain strength level, denoting its estimated strength. A league may contain teams of different strengths, as long as the difference in level for each pair of teams in a league is within a maximum value. Furthermore, the leagues' sizes must lie between a minimum and a maximum value. It is undesirable for teams from the same club to play in the same league, however a maximum value is defined indicating how many teams from the same club are permitted to do so. Additionally, no team may travel more than a predefined maximum distance to another team's home venue.

The STGP is a challenge faced by numerous sport associations. The Royal Belgian Football Association (RBFA), for instance, organizes youth football leagues. Within the East Flanders province alone, these leagues comprise of more than 500 youth teams playing approximately 5000 matches per year. Therefore, reducing the total travel time and distance is very desirable, both logistically and economically. It is especially noteworthy to highlight how the RBFA faces multiple instances of the STGP: there are round-robin tournaments to be constructed for each age category of each province. In addition to being highly relevant in practice due to the large number of people it affects, the STGP is also challenging from a computational

perspective. Indeed, it generalizes the clique partitioning problem, and as a consequence, is an NP-Hard problem

This paper presents three integer programming formulations for the STGP: two compact formulations and one with an exponential number of variables derived from Ji and Mitchell (2007). The first two formulations are solved by an integer programming solver (Gurobi), while the third is solved by a tailor-made branch-and-price algorithm. Gurobi is employed to solve the master problem, whereas a specialized heuristic handles the column generation pricing problem. Additionally, a meta-heuristic is employed to generate high-quality solutions for large instances.

The present paper is organized as follows. The next section offers a more detailed description of the STGP and positions it in relation to other problems in the literature. Section 3 presents and compares three integer linear programming formulations. Section 4 details the formulation with an exponential number of variables, solved by column generation, and introduces the heuristic responsible for solving the pricing problem. Section 5 explains the developed branch-and-price approach. Section 6 describes the meta-heuristic method produced. Computational experiments are presented in Sect. 7 and, finally, Sect. 8 summarizes the conclusions.

2 The sport teams grouping problem

Given a set of clubs C, a set of teams T and (symmetric) distances between each pair of teams, the STGP consists of distributing these teams over leagues while minimizing the total distance traveled by the teams in round-robin tournaments. Each team $i \in T$ is assigned a strength level s_i (where $s_i \in \{1, 2, 3\}$), and a club $c_i \in C$. Each league must contain between m^- and m^+ teams. At most c^+ teams from the same club are permitted in a single league and teams within a league must not be further than d^+ distance units apart.

Essentially, the STGP may be described by a graph $G = (V, E)$ where each vertex represents a team ($V = T$). Two vertices are connected by an edge $(i, j) \in E$ if the difference in level between the corresponding teams does not exceed 1 ($|s_i - s_j| \leq 1$) and the distance between them, given by $d_{i,j}$, is not greater than the limit d^+ ($d_{i,j} \leq d^+$). The weight of an edge (i, j) is given by the distance $d_{i,j}$ between the corresponding teams' locations. The objective is to partition the vertex set V into a set of cliques whose sizes respect both the minimum and maximum limits given by m^- and m^+, while minimizing the total weight of the cliques' edges. Note that since all edges have positive weights, cliques tend to be as small as possible. Nevertheless, m^+ is defined with $m^+ > m^-$ to reduce infeasibility and handle instances with teams geographically clustered together, where fewer larger cliques may result in less weight than more small ones.

Figure 1 shows a solution to real-world instances of the STGP faced by the RBFA. The problem depicted by the figure contains 521 teams distributed among 169 clubs and 4 different age categories. In this example, $m^- = 5$, $m^+ = 8$ and $c^+ = 2$.

The STGP is a generalization of the clique partitioning problem with minimum requirement (Labbé and Özsoy 2010; Ji and Mitchell 2007). The main differences between the STGP and the clique partitioning problem with minimum clique size requirement are (1) the presence of a maximum clique size requirement, and (2) how the STGP imposes a limit on the number of teams from the same club in a league. The latter requirement may be interpreted as associating a color with each node such that nodes of the same club receive the same color. The STGP consequently requires cliques formed to not contain more than a given number of nodes of the same color.

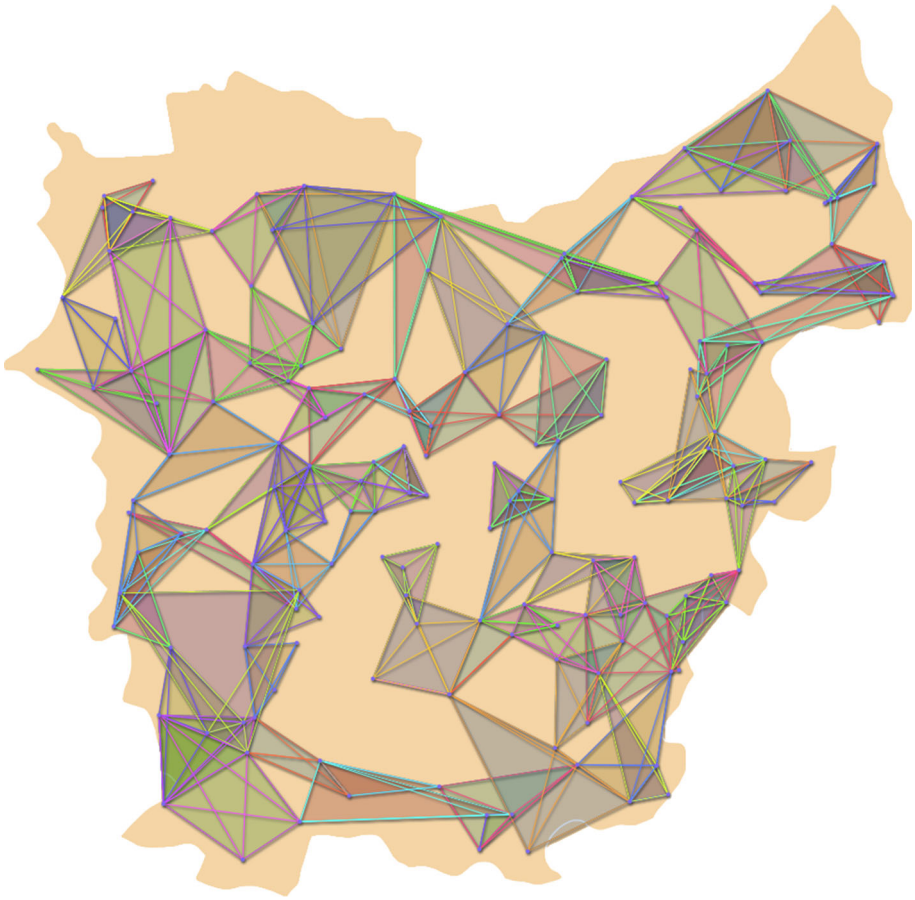

Fig. 1 Sample STGP instance solution

3 Integer programming formulations

Integer programming formulations for clustering (or grouping) problems have been studied extensively in the literature. Those seeking an extensive overview are referred to Ales et al. (2016).

This section presents three integer (binary) linear programming formulations for the STGP: two compact formulations ($\mathcal{F}_1$ and $\mathcal{F}_2$) and another with an exponential number of variables ($\mathcal{F}_3$). Before proceeding to detailed descriptions of each formulation, the input data is introduced:

C: set of clubs;
L: set of leagues; $L = \{1, 2, \ldots, |L|\}$, where $|L| = \lfloor |T|/m^- \rfloor$ is the maximum number of leagues;
T: set of teams;
T_c: set of teams from club c, $c \in C$;
c_i: club of team i, $i \in T$;
c^+: maximum number of teams from the same club per league

$d_{i,j}$: (symmetric) distance between distinct teams i and j, $i, j \in T$;
d^+: maximum permitted distance between a pair of teams in a league;
m^-: minimum number of teams in a league;
m^+: maximum number of teams in a league;
s_i: strength level of team i, $i \in T$.

Additionally, the following notation is introduced to simplify the presentation:

T^2: set of distinct team pairs, $T^2 = \{(i, j) : i \in T, j \in T, i \neq j\}$;
A^2: pairs of teams allowed within the same league, $A^2 = \{(i, j) \in T^2 : |s_i - s_j| \leq 1 \wedge d_{i,j} \leq d^+\}$;
F^2: pairs of teams forbidden within the same league, $F^2 = T^2 \backslash A^2$.

3.1 A first compact formulation

The first compact formulation ($\mathcal{F}_1$) considers three sets of binary variables:

$x_{i,\ell}$ equals 1 if and only if team i is assigned to league ℓ ($i \in T, \ell \in L$);
$y_{i,j}$ equals 1 if and only if teams i and j are assigned to the same league ($(i, j) \in A^2$);
z_ℓ equals 1 if and only if league ℓ is part of the solution ($\ell \in L$).

Next, $\mathcal{F}_1$ is presented.

$$\text{minimize} \sum_{(i,j)\in A^2} d_{i,j} y_{i,j} \tag{1}$$

$$\text{subject to} \sum_{\ell \in L} x_{i,\ell} = 1 \qquad \forall\, i \in T \tag{2}$$

$$m^- z_\ell \leq \sum_{i \in T} x_{i,\ell} \leq m^+ z_\ell \qquad \forall\, \ell \in L \tag{3}$$

$$\sum_{i \in T_c} x_{i,\ell} \leq c^+ z_\ell \qquad \forall\, \ell \in L, c \in C \tag{4}$$

$$x_{i,\ell} + x_{j,\ell} \leq y_{i,j} + 1 \qquad \forall\, (i, j) \in A^2, \ell \in L \tag{5}$$

$$x_{i,\ell} + x_{j,\ell} \leq 1 \qquad \forall\, (i, j) \in F^2, \ell \in L \tag{6}$$

$$y_{i,j} \in \{0, 1\} \qquad \forall\, (i, j) \in A^2 \tag{7}$$

$$x_{i,\ell}, z_\ell \in \{0, 1\} \qquad \forall\, i \in T, \ell \in L \tag{8}$$

The objective function (1) minimizes the total travel distance. Constraints (2) guarantee each team is assigned to exactly one league. Constraints (3) activate the z variables, while verifying the number of teams in a selected league $\ell \in L$ respects the lower and upper limits m^- and m^+, respectively. Note that these constraints also guarantee that if a league $\ell \in L$ is not part of the solution ($z_\ell = 0$), then $x_{i,\ell} = 0\ \forall i \in T$. Constraints (4) ensure the number of teams from the same club in a league does not exceed limit c^+. Constraints (5) activate the y variables, assigning to $y_{i,j}$ a value 1 if teams i and j are in the same league and 0 otherwise. Constraints (6) assert that teams $(i, j) \in F^2$ are not assigned to the same league and, finally, Constraints (7) and (8) declare that all variables are binary.

The linear relaxation of $\mathcal{F}_1$ replaces Constraints (7) and (8) by $0 \leq y_{i,j} \leq 1, 0 \leq x_{i,\ell} \leq 1$, and $0 \leq z_\ell \leq 1$, for all i, j, ℓ. This linear relaxation is weak; for all instances considered in this paper, a trivial lower bound of 0 is found. This derives from the fact that for all these

instances Constraints (5) are easily satisfied by setting $x_{i,\ell} = x_{j,\ell} = \frac{1}{2}$, hence permitting $y_{i,j} = 0$. To cut away this fractional solution, Constraints (9) are introduced:

$$m^- - 1 \leq \sum_{j:(i,j)\in A^2} y_{i,j} \leq m^+ - 1 \qquad \forall\, i \in T \tag{9}$$

These constraints certify each team has at least $m^- - 1$ and at most $m^+ - 1$ opponents in a league. This condition cuts away solutions with $y_{i,j} = 0\ \forall (i,j) \in A^2$, considerably improving the quality of the lower bound provided by the linear relaxation.

Formulation (1)–(9) may be further improved by reducing the number of symmetric solutions. Note that all leagues $\ell \in L$ are identically defined, with the subsequent permuting of leagues generating symmetric solutions. To reduce symmetry, a set $\tilde{T}$ of teams which cannot coexist in the same league is fixed to different leagues. Observe that such a set of teams corresponds to an independent set in G. While a maximal independent set is easily found, T should ideally be a maximum independent set of G. Additionally, Constraints (10) are included to further reduce symmetry for instances in which not all leagues are employed. Thus, we can extend $\mathcal{F}_1$ by adding (9)–(10).

$$z_{\ell-1} \leq z_\ell \quad \forall \ell \in L \backslash \{1\} \tag{10}$$

3.2 Another compact formulation

The second compact formulation ($\mathcal{F}_2$) for the STGP only considers variables $y_{i,j}$, indicating whether teams i and j are in the same league ($y_{i,j} = 1$) or not ($y_{i,j} = 0$), with $(i,j) \in A^2$. In addition to the input data previously presented, three sets are defined:

A_i: set of teams in T which can be present with team i in the same league, $A_i = \{j : (i,j) \in A^2\}, i \in T$;

A^3: set of triples of distinct teams allowed in the same league, $A^3 = \{(i,j,k) : (i,j) \in A^2, (i,k) \in A^2, (j,k) \in A^2, i \neq j \neq k\}$.

F^3: set of triples of distinct teams (i,j,k) such that teams i and j are allowed in the same league, while teams j and k cannot be in the same league, $F^3 = \{(i,j,k) : (i,j) \in A^2, (i,k) \in A^2, (j,k) \in F^2, i \neq j \neq k\}$.

The second compact formulation, $\mathcal{F}_2$, is given by Eqs. (11)–(16).

$$\text{minimize} \sum_{(i,j)\in A^2} d_{i,j} y_{i,j} \tag{11}$$

$$\text{subject to } y_{i,j} + y_{i,k} - y_{j,k} \leq 1 \qquad \forall\, (i,j,k) \in A^3 \tag{12}$$

$$y_{i,j} + y_{i,k} \leq 1 \qquad \forall\, (i,j,k) \in F^3 \tag{13}$$

$$m^- - 1 \leq \sum_{j\in A_i} y_{i,j} \leq m^+ - 1 \qquad \forall\, i \in T \tag{14}$$

$$\sum_{j\in A_i \cap T_{c_i}} y_{i,j} \leq c^+ - 1 \qquad \forall\, i \in T \tag{15}$$

$$y_{i,j} \in \{0,1\} \qquad \forall\, (i,j) \in A^2 \tag{16}$$

The objective function (11) minimizes the total travel distance. Constraints (12) and (13) guarantee triangle inequalities are respected. Constraints (14) verify each team is in a league

with at least $m^- - 1$ and at most $m^+ - 1$ other teams. Constraints (15) limit the number of additional teams from the same club in a league to $c^+ - 1$, guaranteeing no league has more than c^+ teams from the same club. Finally, Constraints (16) declare all variables binary.

$\mathcal{F}_2$ contains fewer variables than $\mathcal{F}_1$ and a considerably greater number of constraints. While $\mathcal{F}_1$ contains $O(|T|^2)$ constraints, $\mathcal{F}_2$ has $O(|T|^3)$ constraints.

3.3 A formulation with an exponential number of variables

The third formulation ($\mathcal{F}_3$) for the STGP is based on a variable for each feasible league (a set of teams or clique). This formulation represents a standard set partitioning formulation, where Ω is the set of possible leagues (columns) (see Ji and Mitchell 2007). The total travel distance of a league $\omega \in \Omega$ is denoted by d'_ω. Binary decision variables λ_ω are defined, being equal to 1 if and only if league $\omega \in \Omega$ is selected.

Next, $\mathcal{F}_3$ is stated:

$$\text{minimize} \sum_{\omega \in \Omega} d'_\omega \lambda_\omega \tag{17}$$

$$\text{subject to} \sum_{\omega \in \Omega : i \in \omega} \lambda_\omega = 1 \qquad \forall\, i \in T \tag{18}$$

$$\lambda_\omega \in \{0, 1\} \qquad \forall\, \omega \in \Omega \tag{19}$$

The objective function (17) minimizes the total travel distance. Constraints (18) ensure each team is assigned to exactly one league, while Constraints (19) state that variables λ are binary.

The linear relaxation of $\mathcal{F}_3$ replaces Constraints (19) by $\lambda_\omega \geq 0$ for each $\omega \in \Omega$. If the number of teams $|T|$ is not a multiple of the minimum league size, m^-, the linear relaxation of formulation (17)–(19) has a greater tendency of resulting in fractional solutions than if $|T|$ is a multiple of m^-, as observed by Ji and Mitchell (2007). Indeed, since larger leagues have a higher number of edges, they are generally associated with larger costs. These larger leagues may be replaced by a fractional combination of smaller leagues in the linear relaxation, which often yields a better objective value. Note that since m^- is the minimum league size, a solution for a problem with $|T|$ teams contains at most $\lfloor |T|/m^- \rfloor$ leagues. Constraint (20) strengthens formulation (17)–(19) by limiting the total number of selected leagues and thus reducing the replacement of large leagues by fractional smaller leagues. This constraint is included in $\mathcal{F}_3$.

$$\sum_{\omega \in \Omega} \lambda_\omega \leq \left\lfloor \frac{|T|}{m^-} \right\rfloor \tag{20}$$

The impact of Constraint (20) in $\mathcal{F}_3$ is evaluated in Sect. 7.

3.4 Strength of formulations

This section compares the strength of $\mathcal{F}_3$ against $\mathcal{F}_1$ and $\mathcal{F}_2$. The linear relaxation's optimal objective value of some formulation $\mathcal{A}$ on a given STGP instance I is denoted as $z_A(I)$. Following standard terminology, formulation $\mathcal{A}$'s linear relaxation is said to be *stronger* than that of formulation $\mathcal{B}$ if the following two conditions are fulfilled:

- for each instance I, $z_\mathcal{A}(I) \geq z_\mathcal{B}(I)$,
- there exists an instance I where $z_\mathcal{A}(I) > z_\mathcal{B}(I)$.

Theorem 1 *The linear relaxation of $\mathcal{F}_3$ is stronger than the linear relaxation of $\mathcal{F}_1$.*

Theorem 2 *The linear relaxation of $\mathcal{F}_3$ is stronger than the linear relaxation of $\mathcal{F}_2$.*

The proofs for Theorems 1 and 2 are presented in "Appendix A". The theorems justify investigating formulation $\mathcal{F}_3$. Its solution is, however, not straightforward. The algorithm employed to solve $\mathcal{F}_3$ is presented in the next sections.

4 Column generation approach

Explicitly implementing $\mathcal{F}_3$ requires an exponential number of variables, a difficulty which may be avoided by employing a column generation scheme (Dantzig and Wolfe 1960) in which variables are generated based on dual costs. For an overview of column generation, we refer the reader to Lübbecke and Desrosiers (2005) and Vanderbeck and Wolsey (2010).

Column generation is an iterative approach. A restricted master problem (RMP) is defined which initially consists of $\mathcal{F}_3$ with $\Omega = \emptyset$. Artificial variables with sufficiently high costs in the objective function are introduced to satisfy Constraints (18). The linear relaxation of the restricted master problem is subsequently solved. The pricing problem is solved at each iteration to obtain columns (leagues) with negative reduced cost considering the current dual values γ and τ corresponding to Constraints (18) and (20), respectively. A column $\omega \in \Omega$ has negative reduced cost if $d'_\omega - \sum_{i \in \omega} \gamma_i - \tau < 0$. If such columns are found, they are added to the restricted master problem, which is subsequently re-solved. The algorithm continues until no columns with negative reduced cost exist, whereupon the linear relaxation of the master problem is solved.

The implemented column generation approach resembles other procedures discussed in the literature, such as those by Mehrotra and Trick (1998) and Ji and Mitchell (2007). The difference is formed by the pricing problem, since the STGP includes additional particular constraints.

4.1 Pricing problem

The pricing problem formulated is consistent with the branching strategy employed. This strategy (see Sect. 5 for a precise description) enables teams to be *merged*. Two additional parameters are defined to keep track of the properties of *merged* teams (referred to as nodes):

μ_i number of teams merged in node i;
$\vartheta_{i,c}$ number of teams from club c in node i.

Initially, for each team $i \in T$, $\mu_i = 1$ and $\vartheta_{i,c} = 1$ if $c = c_i$ or $\vartheta_{i,c} = 0$ otherwise.

Notice how the presence of node coefficients γ_i in the objective function, together with the variable size of the clique, necessitates the use of node-variables x in an integer programming formulation of the pricing problem. Therefore, two sets of binary variables are defined:

x_i equals 1 if and only if team i is selected ($i \in T$);
$y_{i,j}$ equals 1 if and only if teams i and j are both selected ($(i, j) \in A^2$).

Recall the association of dual variables γ_i to Constraints (18) and dual variable τ to Constraint (20). Considering these associations, a formulation for the pricing problem is given by Eqs. (21)–(26).

$$\text{minimize} \sum_{(i,j)\in A^2} d'_{i,j} y_{i,j} - \sum_{i\in T} \gamma_i x_i - \tau \tag{21}$$

$$\text{subject to } m^- \leq \sum_{i\in T} \mu_i x_i \leq m^+ \tag{22}$$

$$\sum_{i\in T} \vartheta_{i,c} x_i \leq c^+ \qquad \forall\, c \in C \tag{23}$$

$$x_i + x_j \leq y_{i,j} + 1 \qquad \forall\, (i,j) \in A^2 \tag{24}$$

$$x_i + x_j \leq 1 \qquad \forall\, (i,j) \in F^2 \tag{25}$$

$$x_i, y_{i,j} \in \{0,1\} \qquad \forall\, (i,j) \in A^2 \tag{26}$$

The objective function (21) minimizes the total distance while considering dual costs γ and τ. Note that a solution for this pricing problem is a column with negative reduced cost when its objective function value is negative. Constraint (22) ensures the produced league's size is between m^- and m^+. Constraints (23) limit the number of teams from the same club in a single league to no more than c^+. Constraints (24) ensure $y_{i,j} = 1$ whenever teams i and j are both selected. Constraints (25) prevent the assignment of two incompatible teams to the same league and, finally, Constraints (26) define variables x and y as binary.

Essentially, this formulation seeks to discover a single weighted clique in a graph with both node- and edge-weights. The polyhedral structure of this problem [which does not include Constraints (23)] is studied intensively by Sørensen (2004).

Analogously to formulation (1)–(8), the linear relaxation of formulation (21)–(26) generates weak bounds since $y_{i,j} = 0\ \forall (i,j) \in A^2$ is possible. Constraint (27) is included to prevent this:

$$m^-(m^- - 1) \leq \sum_{(i,j)\in A^2} y_{i,j} \leq m^+(m^+ - 1) \tag{27}$$

Constraint (27) ensures at least $m^-(m^- - 1)$ and at most $m^+(m^+ - 1)$ games are played in a league, cutting away fractional solutions where $y_{i,j} = 0\ \forall (i,j) \in A^2$.

4.2 Solving the pricing problem

The pricing problem is either solved optimally by a general integer programming solver, or addressed heuristically without convergence proof. Although a heuristic may not find a variable with negative reduced cost even if one exists, it requires considerably less computation time than integer programming solvers.

The heuristic procedure begins by sorting nodes (or teams) by descending order with respect to dual values τ, providing a sequence for selecting the first node to add to the league. This initial league is iteratively altered by local search: the best neighboring solution of a randomly chosen neighborhood is selected. Solutions violating the maximum number of teams per club or the maximum league size are not considered. The set of local search neighborhoods includes:

1. *add nodes* adds, if possible, a node to a compatible league without exceeding the maximum league size.
2. *remove nodes* removes, if possible, a node without violating the minimum league size.
3. *swap nodes* greedily replaces the node adding the greatest value to the objective function with the node that would incur the smallest increase. Note that the contribution of a node i to the objective function is divided by its number of teams (μ_i).

The exploration of neighborhoods is randomized by rejecting neighboring solutions with a small probability β ($\beta = 20\%$), as proposed by Christiaens and Vanden Berghe (2016). While iterating over the neighborhoods, all feasible leagues with negative reduced cost are stored and ordered by increasing cost. The local search procedure ends when a certain iteration limit is reached. If no league with negative reduced cost has been generated before the iteration limit, additional computation time is granted to the heuristic, thus potentially avoiding the necessity of solving the integer programming formulation, which requires a substantial amount of runtime.

Algorithm 1 depicts the heuristic procedure. The algorithm requires four arguments: (1) a list with all nodes sorted by their dual cost, (2) a function that calculates a league's cost considering the dual values, (3) the probability of randomly rejecting neighboring solutions β and (4) a maximum number of iterations per node m. The algorithm begins by initializing the list of leagues L and the pointer p to the current node (lines 1–2). Next, the main loop begins (line 3). Note that there are two possible stopping criteria: (1) when $L \neq \emptyset$, a maximum number of iterations is considered, otherwise (2) a maximum runtime is set as the stopping criterion. The goal, as mentioned before, is to grant the heuristic additional time with the intent of avoid solving the integer programming formulation. Inside the main loop, first the current league ℓ is initialized with the node in position p (line 4). Afterwards, pointer p is updated (line 5). The algorithm performs local search using the neighborhoods presented for a limited number of iterations m, with $m = 50$ throughout computation experiments (line 6). First, a neighborhood is randomly selected (line 7). Second, a feasible neighbor is produced considering current league ℓ and acceptance prevention parameter β (line 8). If a novel league (not in L) is generated, it is accepted (lines 9–10). If the league also represents a negative reduced cost column, it is added to set L (line 11). Finally, the set of produced columns is returned (line 12).

Algorithm 1: Pricing heuristic

Let C be a list with all nodes sorted by increasing dual cost
Let $f(.)$ be a function that returns the cost of league (∞ if league is infeasible)
Let β be the probability of rejecting a neighbor
Let m be the maximum number of iterations per node
PricingSolver(C, $f(.)$, β, m)**:**

```
1  L ← ∅
2  p ← 0
3  while stopping criterion not met do
4      ℓ ← C_p
5      p ← p + 1
6      while iteration limit m not met do
7          N ← random neighborhood
8          ℓ' ← feasible solution in neighborhood N(ℓ, β)
9          if ℓ' ∉ L then
10             ℓ ← ℓ'
11             if f(ℓ) < 0 then L ← L ∪ {ℓ};
12 return L
```

Whenever the heuristic fails to provide a column with negative reduced cost within its computation time limit, formulation (21)–(26) is solved to either find a column with negative reduced cost or prove the nonexistence of such a column.

5 Branch-and-price algorithm

The column generation algorithm (Sect. 4) solves the linear relaxation of $\mathcal{F}_3$. The solutions produced may be fractional, in which case it is necessary to branch on the variables to find an integer solution. When this occurs, a branch-and-price algorithm (Barnhart et al. 1998) is applied. The algorithm is a variant of branch-and-bound where the relaxation is solved by column generation in each search tree node.

Suppose the linear relaxation of $\mathcal{F}_3$ has been solved. If the solution is integral, the procedure ends. Otherwise, the solution contains fractional λ variables, signifying that there exist λ_ω such that $0 < \lambda_\omega < 1$. Therefore, there is at least one pair of teams (i, j) such that the sum of the λ variables corresponding to leagues containing both i and j is (strictly) between 0 and 1. Clearly, in an optimal solution these two teams are either in the same league, or they are not. The branching relies on this property, and as a consequence two branches are created. The pricing problem in the branch where teams i and j are not in the same league is constructed as follows: $A^2 \leftarrow A^2 \backslash \{(i, j)\}$, and $F^2 \leftarrow F^2 \cup \{(i, j)\}$. In the other branch, where the teams i and j must be in the same league, i and j are merged. In practice, this translates into replacing teams i and j by the *merged* node, say node n, such that $T \leftarrow T \backslash \{i, j\} \cup \{n\}$. When merging the teams into node n, related parameters must be set:

$$\begin{aligned}
&\text{for each } k \in T \backslash \{i, j\}\text{: } d_{n,k} \leftarrow d_{i,k} + d_{j,k};\\
&\text{for each } c \in C\text{: } \vartheta_{n,c} \leftarrow \vartheta_{i,c} + \vartheta_{j,c};\\
&\gamma_n \leftarrow \gamma_i + \gamma_j - d_{i,j};\\
&\mu_n \leftarrow \mu_i + \mu_j.
\end{aligned}$$

Note that the distance between teams i and j is embedded in γ_n. Additionally, all sets must be updated accordingly. In particular, $A^2 \leftarrow A^2 \cup \{(n, k) : (i, k) \in A^2 \text{ and } (j, k) \in A^2\}$ (so node n is connected to all nodes in T to which teams i and j are both connected to), and next $A^2 \leftarrow A^2 \backslash \{(i, k), (j, k) : k \in T \backslash \{i, j\}\}$ (thereby removing all pairs involving either i or j).

Selecting the pair of teams i and j considered for branching is a critical component of branch-and-bound algorithms (Achterberg et al. 2005). Ideally, the pair of teams i and j incurring the largest objective function increase for both branches should be selected. However, detecting such a pair of teams may be very time consuming. The developed algorithm employs a heuristic strategy: the teams i and j farthest apart are selected. This criterion yielded better results than, for instance, selecting the pair of teams with sum of related λ variables closest to 0.5.

6 Metaheuristic approach to the STGP

A simulated annealing heuristic was developed which is capable of quickly finding good solutions for the STGP. First, an initial solution is constructed by assigning leagues to all teams via a randomized best insertion method. The procedure selects a team $t \in T$ and then iterates over all existing leagues which are compatible with t. Whenever a league yields a lower insertion cost for t, it is accepted with 80% probability. If no league is accepted, then a new one is created including only t. This constructive method does not ensure feasibility of the constructed solution since some leagues may contain fewer teams than m^-. Therefore, leagues are iteratively improved afterwards. A simple ruin-and-recreate procedure is applied to the solution. First a random number of leagues in the current solution are selected. Then q ($q \leq 2$) teams are removed from each league, with q being limited by the number of

teams that can be removed without causing infeasibility. In a small percentage of the cases, given by a parameter α, leagues are allowed to become infeasible during the ruining phase. Thereafter, new separate leagues are created for a fraction $f = 1/m^-$ of all removed teams. This construction method attempts to insert all remaining teams considering only leagues with a shortage of teams. Finally, the remaining teams are assigned to leagues by considering the full solution's objective. If no removed teams are reassigned, the neighbor solution is rejected and the original solution returned. The neighbor solution created by the ruin-and-recreate procedure may be accepted or not, depending on simulated annealing. The system is cooled down from $t_0 = 100$ to $t_1 = 1$ in 80 million iterations. The cooling rate σ is given by Eq. (28), where t_0 is the initial temperature, t_1 the final temperature and I the number of iterations.

$$\sigma = \left(\frac{t_1}{t_0}\right)^{\frac{1}{I-1}} \tag{28}$$

Algorithm 2 details the simulated annealing procedure. The procedure requires four arguments: (1) initial solution, (2) initial temperature, (3) final temperature and (4) cooling rate. Additionally, let $f(.)$ correspond to the function that returns a solution's objective value and $g(.)$ a solution's number of infeasible leagues. Initially, the best solution S^* and current temperature t are initialized (lines 1–2). While the final temperature is not reached (line 3), neighboring solutions are generated (line 4) and evaluated. Solutions with a higher number of infeasible leagues are always rejected (line 5). Otherwise, if the produced solution improves upon the previous one (lines 6–7), it is accepted (line 8). If it improves the best one, then the best solution is also updated (line 9). Worsening solutions are accepted with a certain probability (lines 11–12). The temperature is subsequently updated (line 13) and the algorithm proceeds to the next iteration. Once the final temperature is reached, the best solution generated is returned (line 14).

Algorithm 2: Simulated annealing

Let S be the current solution
Let t_0 and t_1 be the initial and final temperatures, respectively
Let σ be the cooling rate, $\sigma \in [0, 1]$
Let $f(.)$ be the function that returns a solution's objective value
Let $g(.)$ be the function that returns the solution's number of infeasible leagues
SimulatedAnnealing(S, t_0, t_1, σ)**:**

```
1  S* ← S
2  t ← t0
3  while t > t1 do
4      S' ← RuinAndRecreate(S) // creating neighboring solution
5      if g(S') ≤ g(S) then
6          Δ ← f(S') − f(S)
7          if Δ ≤ 0 then
8              S ← S'
9              if f(S') < f(S*) then S* ← S'
10         else
11             take a random x ∈ [0, 1]
12             if x < e^(−Δ/t) then S ← S' // accepting worse solution
13     t ← σ × t
14 return S*
```

Algorithm 3 describes the ruin-and-recreate procedure. The algorithm has two parameters: (1) the probability of rejecting infeasible leagues α and (2) the probability of rejecting a league β. Empirical evaluation revealed good results with $\alpha = 90\%$ and $\beta = 20\%$, and therefore these values were employed throughout computation experiments. The algorithm begins by creating a copy of the initial solution S (line 1). Next the set of removed teams is initialized (line 2) and a random number of leagues are selected to be ruined (line 3). For each league ℓ (line 4), the number of teams q to remove is calculated (lines 5–7). Note that q is updated to prevent ℓ from becoming infeasible in line 7. However, the algorithm accepts a small percentage α of leagues to become infeasible during the ruining phase. Afterwards, up to q teams are removed from league ℓ (line 8) and added to the set of removed teams (line 9). When $q \leq 0$, T' is an empty set and consequently no teams are removed. Once the ruining phase is finished, the recreation process begins with the inclusion of $\lfloor |T^s|/m^- \rfloor$ empty leagues in the solution (line 10). Next, each removed team t is processed to be assigned to a new league (line 11). A candidate list of leagues L^c is built, containing initially only infeasible leagues (line 12). Feasible leagues are included when S' contains no infeasible ones (line 13). Next one league is selected from F^c. Note that the selection of the cheapest league is randomized by rejecting certain leagues with a small probability β (Christiaens and Vanden Berghe 2016). Team t is then added to the selected league (lines 15–16). Finally, if the produced solution is feasible, it is returned (line 17); otherwise, the initial solution S is returned (line 18).

Algorithm 3: Ruin-and-recreate

Let S be the initial solution and T be the set of all teams
Let α be the probability of rejecting infeasible leagues
Let β be the probability of rejecting a league
RuinAndRecreate(S)**:**

1 $S' \leftarrow S$
2 $T^s \leftarrow \emptyset$
3 $L^s \leftarrow$ randomly selected leagues from S' `// from 1 to |S| leagues may be selected`
`// ruining solution`
4 **for** $\ell \in L^s$ **do**
5 $q \leftarrow 2$ `// up to 2 teams are removed from a league`
6 take a random number $x \in [0, 1]$
7 **if** $x > \alpha$ **then** $q \leftarrow \min(q, |\ell| - m^-)$ `// impeding ℓ from becoming infeasible`
8 $T' \leftarrow$ remove up to q random teams from ℓ
9 $T^s \leftarrow T^s \cup T'$
`// recreating solution`
10 **for** $i \leftarrow 0$ *to* $\lfloor |T^s|/m^- \rfloor$ **do** $S' \leftarrow S' \cup \{\emptyset\}$ `// adding empty leagues to S'`
11 **for** $t \in T^s$ **do**
12 $L^c \leftarrow$ infeasible leagues from S' that can accommodate t
13 **if** $L^c = \emptyset$ **then** $L^c \leftarrow$ feasible leagues from S' that can accommodate t
14 $\ell \leftarrow$ best league (with rejection probability β) from L^c
15 $\ell' \leftarrow \ell \cup \{t\}$
16 $S' \leftarrow S' \cup \{\ell'\} \backslash \{\ell\}$
17 **if** S' *is feasible* **then return** S'
18 **else return** S

7 Computational experiments

Computational experiments were conducted to evaluate the performance of the three formulations, $\mathcal{F}_1$, $\mathcal{F}_2$, and $\mathcal{F}_3$. The running time for solving each instance was limited to 3 h. The formulations and heuristics were coded in Java 1.8, and the commercial solver Gurobi 6.5 was employed. The experiments were performed on an Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz computer running Ubuntu 16.04.2 LTS.

In addition to the real-world instances gently provided by Movetex[1] (u-13, u-15, u-17 and u-21), small artificial instances were generated to further evaluate the algorithms behavior (sets *tiny* and *small*). All instances are available online[2] to encourage future research on the STGP. Note that the real-world instances were slightly altered as to hide information concerning clubs and teams.

Table 1 details the results obtained within 3 h for all presented formulations (Sect. 3) and for the meta-heuristic. Each row presents results obtained with an instance of size $|T|$ considering certain bounds for m^- and m^+. For each formulation the value of the linear relaxation (LB^0), the best lower bound obtained (LB^*), the best solution (UB), the runtime and the resulting gap are presented. Note that lower bounds are rounded up and that, for compactness, the runtime is given in different units: seconds (s), minutes (m) or hours (h). For $\mathcal{F}_3$ the lower bound obtained by the linear relaxation without Constraint (20) is also indicated (LB^a). In addition, the results obtained by simulated annealing (SA) and the gaps between the overall best solution and the best lower bound (Min. Gap) are included. Finally, best lower bounds and solutions are highlighted in bold.

Among the compact formulations ($\mathcal{F}_1$ and $\mathcal{F}_2$), $\mathcal{F}_2$ produced considerably better lower bounds. Nevertheless, both compact formulations had difficulties handling larger instances (u-13, u-15 and u-17), generating poor lower bounds and solutions. When considering branch-and-price ($\mathcal{F}_3$), Constraint (20) proved essential whenever $|T|$ was indivisible by m^-, as expected. It considerably improves the quality of the bound provided by the linear relaxation, leading to a stronger formulation, as can be seen when comparing columns LB^a and LB^0. Branch-and-price proved more adequate than the presented compact formulations for solving the STGP. While it was unable to provide feasible solutions for some of the large instances (u-13, u-15 and u-17), branch-and-price always provided the best lower bounds. Additionally, simulated annealing generated high quality solutions, obtaining optimum solutions for most solved instances. It is noticeable, however, that simulated annealing had difficulties finding feasible solutions for some tight instances (instances *small* with $m^- \geq 9$).

Overall, the resulting gaps with respect to the best solution and the best bound are very low. The worst gap among all instances is only 1.7%.

It is worth highlighting how the developed heuristic for the pricing problem (Sect. 4.2) performed considerably better than the integer programming approach. Therefore, results using only the solver for the pricing problem were omitted. Finally, it was observed that 87.5% of the branch-and-price runtime was, on average, dedicated to solving the pricing problem with the remainder used for the master problem.

[1] https://movetex.be.

[2] Instance and solution files are available at http://benchmark.gent.cs.kuleuven.be/stgp.

Table 1 Results obtained by the formulations and the simulated annealing algorithm

Inst.	$\|T\|$	m^-	m^+	$\mathcal{F}_1$					$\mathcal{F}_2$					$\mathcal{F}_3$						SA		Min. Gap (%)
				LB^0	LB*	UB	Gap (%)	Time	LB^0	LB*	UB	Gap (%)	Time	LB^a	LB^0	LB*	UB	Gap (%)	Time	UB	Time (min)	
Tiny	15	5	10	550	**724**	**724**	0.0	2.7 s	689	**724**	**724**	0.0	0.9 s	704	704	**724**	**724**	0.0	1.4 s	1438	1.7	0.0
		6	10	937	**1438**	**1438**	0.0	0.5 s	985	**1438**	**1438**	0.0	1.4 s	1019	**1438**	**1438**	**1438**	0.0	0.5 s	**1438**	1.5	0.0
		7	10	1129	**1438**	**1438**	0.0	0.4 s	1286	**1438**	**1438**	0.0	1.4 s	1306	**1438**	**1438**	**1438**	0.0	0.7 s	**1438**	1.0	0.0
Small	50	5	10	996	1044	1202	13.1	3.0 h	1140	**1196**	**1196**	0.0	66.3 s	1168	1169	**1196**	**1196**	0.0	17.8 s	**1196**	1.8	0.0
		6	10	1382	1476	1768	16.5	3.0 h	1641	**1762**	**1762**	0.0	33.7 min	1670	1741	**1762**	**1762**	0.0	22.0 s	**1762**	3.3	0.0
		7	10	1806	1914	2314	17.3	3.0 h	2186	**2312**	**2312**	0.0	24.2 min	2224	2279	**2312**	**2312**	0.0	23.6 s	**2312**	2.2	0.0
		8	10	2258	2440	3120	21.8	3.0 h	2794	2978	**3074**	3.1	3.0 h	2859	3025	**3074**	**3074**	0.0	9.6 s	**3074**	2.9	0.0
		9	10	2748	3182	**4190**	24.1	3.0 h	3430	3604	**4190**	14.0	3.0 h	3517	4180	**4190**	**4190**	0.0	3.5 s	–	1.2	0.0
		10	10	3276	3552	**4190**	15.2	3.0 h	4101	**4190**	**4190**	0.0	87.6 s	4173	4173	**4190**	**4190**	0.0	68.2 s	–	1.4	0.0
u-13	166	5	8	1841	1842	5388	65.8	3.0 h	2075	2118	2244	5.6	3.0 h	2126	2138	**2161**	–	–	3.0 h	**2190**	7.6	1.3
		6	8	2528	2528	11792	78.6	3.0 h	2886	2922	3104	5.9	3.0 h	2934	3031	**3053**	–	–	3.0 h	**3064**	8.7	0.4
		7	8	3290	3290	18786	82.5	3.0 h	3804	3842	4094	6.2	3.0 h	3860	4042	**4054**	**4054**	0.0	2.1 min	**4054**	9.4	0.0
		8	8	–	Proven infeasible			12.5 s	4827	4888	–	–	3.0 h	4938	–	Proven infeasible			8.0 s	–	4.6	–
u-15	167	5	8	1799	1800	9256	80.6	3.0 h	2035	2078	2228	6.7	3.0 h	2077	2109	**2130**	–	–	3.0 h	**2166**	8.5	1.7
		6	8	2507	2508	18984	86.8	3.0 h	2880	2922	3092	5.5	3.0 h	2935	3058	**3084**	**3084**	0.0	1.6 h	**3084**	8.9	0.0
		7	8	3268	3268	11552	71.7	3.0 h	3818	3862	4212	8.3	3.0 h	3884	4099	**4143**	**4148**	0.1	3.0 h	4156	9.1	0.1
		8	8	–	Proven infeasible			19.1 s	4845	4890	–	–	3.0 h	4936	–	Proven infeasible			8.0s	–	4.4	–
u-17	130	5	8	1637	1638	5202	68.5	3.0 h	1887	1938	2064	6.1	3.0 h	1926	1927	**1954**	–	–	3.0 h	**1988**	4.8	1.7
		6	8	2237	2238	7188	68.9	3.0 h	2621	2670	2850	6.3	3.0 h	2670	2792	**2838**	**2838**	0.0	48.1 min	2844	7.6	0.0
		7	8	2889	2890	8988	67.8	3.0 h	3411	3444	3674	6.3	3.0 h	3461	3604	**3640**	**3640**	0.0	1.4 min	**3640**	8.1	0.0
		8	8	–	Proven infeasible			5.9 s	4284	4316	–	–	3.0 h	4344	–	Proven infeasible			5.6 s	–	3.6	–

Table 1 continued

Inst.	$\|T\|$	m^-	m^+	$\mathcal{F}_1$					$\mathcal{F}_2$					$\mathcal{F}_3$						SA		Min. Gap (%)
				LB^0	LB*	UB	Gap (%)	Time	LB^0	LB*	UB	Gap (%)	Time	LB^a	LB^0	LB*	UB	Gap (%)	Time	UB	Time (min)	
u-21	58	5	8	1135	1144	1610	28.9	3.0 h	1279	1392	**1460**	4.7	3.0 h	1317	1417	**1460**	**1460**	0.0	5.8 min	**1460**	4.3	0.0
		6	8	1546	1564	2110	25.9	3.0 h	1758	1868	**1976**	5.5	3.0 h	1789	**1976**	**1976**	**1976**	0.0	1.8 s	**1976**	4.8	0.0
		7	8	2018	2134	2902	26.5	3.0 h	2315	2460	**2476**	0.6	3.0 h	2357	2475	**2476**	**2476**	0.0	6.3 s	**2476**	3.5	0.0
		8	8	–	Proven infeasible			0.5 s	2930	3034	–	–	3.0 h	2987	–	Proven infeasible			1.1 s	–	1.6	–

8 Conclusions

This paper presented the sport teams grouping problem, which constitutes a theoretically-challenging and practical combinatorial optimization problem which is internationally applicable and relevant within a variety of recreational sports environments.

Three different integer programming formulations were presented and evaluated for the STGP, two of which are compact formulations. The remaining third formulation contains an exponential number of variables and is solved using a branch-and-price algorithm. A heuristic method is employed to solve the column generation's pricing problem, accelerating branch-and-price convergence. Furthermore, a simulated annealing approach was proposed for quickly producing solutions for a set of large real-world instances.

The compact formulations were associated with much weaker results than the developed branch-and-price, which proved capable of solving almost all instances, with exception of the very largest ones. The simulated annealing approach was capable of handling both small and large instances, generally obtaining good quality solutions.

This contribution constitutes a valuable alternative to current practice. Overall, strong bounds and high-quality solutions were obtained. The largest optimality gap was only 1.7%. Nevertheless, this somewhat simplified variant of the real-world STGP proved computationally challenging. Instances and solutions were published as to facilitate further research into this interesting combinatorial optimization problem.

Future research directions include experimenting with additional instances. Moreover, it is possible to further enhance the branch-and-price algorithm. Cutting planes and techniques such as strong branching should be studied and pursued as a means for potentially improving branch-and-price convergence. Finally, other classes of algorithms should also be evaluated for solving the STGP.

Acknowledgements Work supported by the Belgian Science Policy Office (BELSPO) in the Inter-university Attraction Pole COMEX (http://comex.ulb.ac.be) and by the Leuven Mobility Research Centre (L-Mob). Editorial support provided by Luke Connolly, KU Leuven. Additionally, we would like to thank Movetex, in particular Dieter De Naeyer and Ken De Norre–De Groof, for the insightful discussions concerning the problem and for making the real-world instances available.

Appendix A

This appendix presents the proofs for Theorems 1 and 2, previously presented in Sect. 3.4.

Theorem 1 The linear relaxation of $\mathcal{F}_3$ is stronger than the linear relaxation of $\mathcal{F}_1$.

Proof Theorem 1 is proven true if conditions C1 and C2 are satisfied:

C1: for each STGP instance I, $z_{\mathcal{F}_3}(I) \geq z_{\mathcal{F}_1}(I)$,
C2: there exists a STGP instance I where $z_{\mathcal{F}_3}(I) > z_{\mathcal{F}_1}(I)$.

The validity of C2 is relatively simple to show; in fact, many of the instances considered in Sect. 7 satisfy this condition. C1 remains to be proven true. Consider λ^*_ω a feasible solution of the linear relaxation of $\mathcal{F}_3$ on some instance I while also assuming $L \leftarrow \Omega$ such that each $\omega \in \Omega$ has an equivalent $\ell \in L$, meaning $\omega = \ell$. Given this solution and the set $L = \Omega$, x-, y- and z-values can be constructed:

$$\text{for each } i \in T \text{ and } \ell \in L: \; x_{i,\ell} = x_{i,\omega} \leftarrow \lambda^*_\omega \tag{29}$$

$$\text{for each } (i,j) \in A^2: \; y_{i,j} \leftarrow \sum_{\omega:(i,j)\in\omega} \lambda^*_\omega. \tag{30}$$

$$\text{for each } \ell \in L: \; z_\ell = z_\omega \leftarrow \lambda^*_\omega \tag{31}$$

Next, it is shown how the resulting solution is a feasible solution of the linear relaxation of $\mathcal{F}_1$, indicating that it satisfies (2)–(6). Constraints (2) are always satisfied, since each team $i \in T$ is assigned to exactly one league [see Constraints (18)]:

$$\sum_{\ell\in L} x_{i,\ell} = \sum_{\omega\in\Omega} x_{i,\omega} = \sum_{\omega:i\in\omega} \lambda^*_\omega = 1 \tag{32}$$

Since the leagues (cliques) $\omega \in \Omega$ are feasible, $m^- \le |\omega| \le m^+ \; \forall \omega \in \Omega$, inequalities in (3) follow:

$$\sum_{i\in T} x_{i,\ell} = \sum_{i\in T} x_{i,\omega} = |\omega|\lambda^*_\omega = |\omega| z_\omega = |\omega| z_\ell \tag{33}$$

Given that all leagues $\omega \in \Omega$ are feasible it also follows that the number of teams from a club in any ω is less than or equal to c^+. Therefore, Constraints (4) are also satisfied since for all leagues $\ell \in L$ and clubs $c \in C$:

$$\sum_{i\in T_c} x_{i,\ell} \le c^+ \lambda^*_\omega = c^+ z_\ell \tag{34}$$

Constraints (5) also hold. If teams i and j are in the same league, they are therefore together in a league $\omega \in \Omega$. Thus, for all $(i,j) \in A^2$ and leagues $\ell \in L$ (or $\omega \in \Omega$) containing both i and j:

$$x_{i,\ell} + x_{j,\ell} = 2\lambda^*_\omega \le y_{i,j} + 1 \tag{35}$$

Observe that Constraints (5) are also satisfied when teams i and j are not in the same league ℓ since this implies $x_{i,\ell} + x_{j,\ell} \le 1$.

Finally, if teams i and j are not allowed in a league, leagues $\omega \in \Omega$ will contain at most one of them (since all leagues ω are feasible). Hence, Constraints (6) also hold for all $(i,j) \in F^2$ and $\ell \in L$:

$$x_{i,\ell} + x_{j,\ell} \le \lambda^*_\omega \le 1 \tag{36}$$

It is thus proven that every solution I of the linear relaxation of $\mathcal{F}_3$ satisfy (2)–(6). Since both $\mathcal{F}_1$ and $\mathcal{F}_3$ have the same objective function, Theorem 1 is proven. □

Theorem 2 The linear relaxation of $\mathcal{F}_3$ is stronger than the linear relaxation of $\mathcal{F}_2$.

Proof Theorem 2 is proven if conditions C3 and C4 are satisfied:

C3: for each STGP instance I, $z_{\mathcal{F}_3}(I) \ge z_{\mathcal{F}_2}(I)$,
C4: there exists a STGP instance I where $z_{\mathcal{F}_3}(I) > z_{\mathcal{F}_2}(I)$.

C4 is proven true by the experiments reported in Sect. 7. It is, however, required to mathematically prove that C3 is true. Consider again that λ^*_ω is a feasible solution of the linear relaxation of $\mathcal{F}_3$ for an instance I; y-values may be easily constructed:

$$\text{for each } (i,j) \in A^2: \; y_{i,j} \leftarrow \sum_{\omega:(i,j)\in\omega} \lambda^*_\omega. \tag{37}$$

Let $\bar{e}^k_{i,j}$ be the sum of λ^*_ω for all $\omega \in \Omega$ which include both i and j but not k, and $e_{i,j,k}$ be the sum of λ^*_ω for all $\omega \in \Omega$ which include i, j and k. More compactly:

$$\bar{e}^k_{i,j} = \sum_{\omega \in \Omega : \{i,j\} \in \omega \wedge k \notin \omega} \lambda^*_\omega \tag{38}$$

$$e_{i,j,k} = \sum_{\omega \in \Omega : \{i,j,k\} \in \omega} \lambda^*_\omega \tag{39}$$

Clearly, for each $k \in T$, $y_{i,j} = \bar{e}^k_{i,j} + e_{i,j,k}$. Equivalently, $y_{i,k} = \bar{e}^j_{i,k} + e_{i,j,k}$ and $y_{j,k} = \bar{e}^i_{j,k} + e_{i,j,k}$. Therefore, for each $(i, j, k) \in A^3$:

$$\begin{aligned} y_{i,j} + y_{i,k} - y_{j,k} = \bar{e}^k_{i,j} + e_{i,j,k} + \bar{e}^j_{i,k} + e_{i,j,k} - \bar{e}^i_{j,k} - e_{i,j,k} \leq \\ \bar{e}^k_{i,j} + \bar{e}^j_{i,k} + e_{i,j,k} = \sum_{\omega \in \Omega : i \in \omega} \lambda^*_\omega = 1. \end{aligned} \tag{40}$$

The first equality in (40) follows by definition; the first inequality from the non-negativity of all y-variables; the second equality from the fact that terms $\bar{e}^k_{i,j}$, $\bar{e}^j_{i,k}$, and $e_{i,j,k}$ correspond to disjoint sets of cliques whose union are all cliques containing i; and the final equality follows from (18). Therefore, y satisfies (12).

Furthermore, observe that $e_{i,j,k} = 0 \ \forall (i, j, k) \in F^3$. This follows from i, j and k not being permitted in the same league, and therefore there exists no league $\omega \in \Omega$ containing these three teams. Thus, modifying (40) shows that y also satisfies (13).

Consider Constraint (14). Observe that for each $i \in T$:

$$\sum_{j \in A_i} y_{i,j} = \sum_{j \in A_i} \sum_{\omega : (i,j) \in \omega} \lambda^*_\omega = \sum_{j} \sum_{\omega : (i,j) \in \omega} \lambda^*_\omega = \sum_{\omega : i \in \omega} (|\omega| - 1) \lambda^*_\omega. \tag{41}$$

Indeed, the first equality follows from (30); the second equality from how clique containing i cannot contain some team $j \notin A_i$; and the final equality results from counting how many times the value λ^*_ω is present in this term.

Next, given that each clique ω is feasible, $m^- \leq |\omega| \leq m^+$. Combining this with (41) results in:

$$\begin{aligned} \sum_{j \in A_i} y_{i,j} &= \sum_{\omega : i \in \omega} (|\omega| - 1) \lambda^*_\omega \leq \sum_{\omega : i \in \omega} (m^+ - 1) \lambda^*_\omega \\ &= (m^+ - 1) \sum_{\omega : i \in \omega} \lambda^*_\omega = m^+ - 1 \end{aligned} \tag{42}$$

$$\begin{aligned} \sum_{j \in A_i} y_{i,j} &= \sum_{\omega : i \in \omega} (|\omega| - 1) \lambda^*_\omega \leq \sum_{\omega : i \in \omega} (m^+ - 1) \lambda^*_\omega \\ &= (m^+ - 1) \sum_{\omega : i \in \omega} \lambda^*_\omega = m^+ - 1. \end{aligned} \tag{43}$$

It follows that y satisfies (14).

Finally, consider (15). Observe that for each team $i \in T$:

$$\sum_{j \in A_i \cap T_i} y_{i,j} = \sum_{j \in A_i \cap T_i} \sum_{\omega : (i,j) \in \omega} \lambda^*_\omega \leq (c^+ - 1) \sum_{\omega : i \in \omega} \lambda^*_\omega = c^+ - 1. \tag{44}$$

Again, the first equality follows from (37) and the inequality from how each clique ω contains at most c^+ nodes from the same club as team i. Hence the value λ_ω occurs at most $c^+ - 1$ times in each clique ω.

It is therefore proven that y satisfies (15).

Finally, note that both formulations have an equal objective function, and hence Theorem 2 is proven.

References

Achterberg, T., Koch, T., & Martin, A. (2005). Branching rules revisited. *Operations Research Letters*, *33*(1), 42–54.

Alarcón, F., Durán, G., Guajardo, M., Miranda, J., Muñoz, H., Ramírez, L., et al. (2017). Operations research transforms the scheduling of chilean soccer leagues and south american world cup qualifiers. *Interfaces*, *47*(1), 52–69.

Ales, Z., Knippel, A., & Pauchet, A. (2016). Polyhedral combinatorics of the k-partitioning problem with representative variables. *Discrete Applied Mathematics*, *211(c)*, 1–14.

Anagnostopoulos, A., Michel, L., Van Hentenryck, P., & Vergados, Y. (2006). A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, *9*(2), 177–193.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P., & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, *46*, 316–329.

Briskorn, D., Drexl, A., & Spieksma, F. C. R. (2010). Round robin tournaments and three index assignments. *4OR*, *8*(4), 365–374.

Carvalho, M. A. M. D., & Lorena, L. A. N. (2012). New models for the mirrored traveling tournament problem. *Computers and Industrial Engineering*, *63*(4), 1089–1095.

Christiaens, J., & Vanden Berghe, G. (2016). *A fresh ruin & recreate implementation for the capacitated vehicle routing problem*. Technical report, KU Leuven, Belgium.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, *8*(1), 101–111.

Di Gaspero, L., & Schaerf, A. (2007). A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, *13*(2), 189–207.

Easton, K., Nemhauser, G., & Trick, M. (2001). The traveling tournament problem description and benchmarks. In T. Walsh (Ed.), *Principles and Practice of Constraint Programming—CP 2001: 7th International Conference, CP 2001 Paphos, Cyprus, 2001 Proceedings* (pp. 580–584). Berlin, Heidelberg: Springer.

Goossens, D., & Spieksma, F. (2009). Scheduling the belgian soccer league. *Interfaces*, *39*(2), 109–118.

Goossens, D., & Spieksma, F. (2014). Indoor football scheduling. In E. Özcan, E. Burke, & B. McCollum (Eds.), *Proceedings of the 10th International Conference on the Practice and Theory of Automated Timetabling* (pp. 167–178), PATAT.

Goossens, D. R., & Spieksma, F. C. R. (2012). Soccer schedules in europe: An overview. *Journal of Scheduling*, *15*(5), 641–651.

Januario, T., Urrutia, S., Ribeiro, C. C., & De Werra, D. (2016). Edge coloring: A natural model for sports scheduling. *European Journal of Operational Research*, *254*(1), 1–8.

Ji, X., & Mitchell, J. E. (2007). Branch-and-price-and-cut on the clique partitioning problem with minimum clique size requirement. *Discrete Optimization*, *4*, 87–102.

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

Knust, S. (2010). Scheduling non-professional table-tennis leagues. *European Journal of Operational Research*, *200*(2), 358–367.

Labbé, M., & Özsoy, F. A. (2010). Size-constrained graph partitioning polytopes. *Discrete Mathematics*, *310*(24), 3473–3493.

Lübbecke, M. E., & Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, *53*(6), 1007–1023.

Mehrotra, A., & Trick, M. A. (1998). Cliques and clustering: A combinatorial approach. *Operations Research Letters*, *22*(1), 1–12.

Nemhauser, G. L., & Trick, M. A. (1998). Scheduling a major college basketball conference. *Operations Research*, *46*(1), 1–8.

Ribeiro, C. C. (2012). Sports scheduling: Problems and applications. *International Transactions in Operational Research*, *19*, 201–226.

Schönberger, J., Mattfeld, D., & Kopfer, H. (2000). Automated timetable generation for rounds of a table-tennis league. In *Proceedings of the 2000 Congress on Evolutionary Computation* (pp. 277–284).

Schönberger, J., Mattfeld, D., & Kopfer, H. (2004). Memetic algorithm timetabling for non-commercial sport leagues. *European Journal of Operational Research*, *153*(1), 102–116.

Sørensen, M. M. (2004). New facets and a branch-and-cut algorithm for the weighted clique problem. *European Journal of Operational Research*, *154*(1), 57–70.

Toffolo, T. A. M., Wauters, T., Van Malderen, S., & Vanden Berghe, G. (2016). Branch-and-bound with decomposition-based lower bounds for the traveling umpire problem. *European Journal of Operational Research*, *250*(3), 737–744.

Trick, M.A., & Yildiz, H. (2007). Bender's cuts guided large neighborhood search for the traveling umpire problem. In P. Van Hentenryck & L. Wolsey (Eds.), *Number 4510 in Lecture Notes in Computer Science* (pp. 332–345), Springer.

Trick, M. A., & Yildiz, H. (2011). Benders' cuts guided large neighborhood search for the traveling umpire problem. *Naval Research Logistics (NRL)*, *58*(8), 771–781.

Trick, M. A., Yildiz, H., & Yunes, T. (2012). Scheduling major league baseball umpires and the traveling umpire problem. *Interfaces*, *42*(3), 232–244.

Uthus, D. C., Riddle, P. J., & Guesgen, H. W. (2011). Solving the traveling tournament problem with iterative-deepening. *Journal of Scheduling*, *15*(5), 601–614.

Vanderbeck, F., & Wolsey, L. (2010). Reformulation and decomposition of integer. In M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, & L. A. Wolsey (Eds.), *50 years of integer programming 1958–2008* (pp. 431–502). Berlin: Springer.

Xue, L., Luo, Z., & Lim, A. (2015). Two exact algorithms for the traveling umpire problem. *European Journal of Operational Research*, *243*(3), 932–943.