IEICE TRANSACTIONS

on Information and Systems

DOI:10.1587/transinf.2024FCP0003

Publicized:2024/08/08

A PUBLICATION OF THE INFORMATION AND SYSTEMS SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

A Bigram Based ILP Formulation for Break Minimization in Sports Scheduling Problems

Koichi FUJII[†a)], ***Nonmember and*** **Tomomi MATSUI**[†b)], ***Member***

SUMMARY

Constructing a suitable schedule for sports competitions is a crucial issue in sports scheduling. The round-robin tournament is a competition adopted in many professional sports. For most round-robin tournaments, it is considered undesirable that a team plays consecutive away or home matches; such an occurrence is called a break. Accordingly, it is preferable to reduce the number of breaks in a tournament. A common approach is to first construct a schedule and then determine a home-away assignment based on the given schedule to minimize the number of breaks (first-schedule-then-break).

In this study, we concentrate on the problem that arises at the second stage of the first-schedule-then-break approach, namely, the break minimization problem (BMP). We propose a novel integer linear programming formulation called the "bigram based formulation." The computational experiments show its effectiveness over the well-known integer linear programming formulation. We also investigate its valid inequalities, which further enhances the computational performance.

key words: *sports scheduling; tournament scheduling; round robin; graph theory; integer linear programming*

1. Introduction

Constructing a suitable schedule for sports competitions is a crucial issue in sports scheduling. A (single) *round-robin tournament* (RRT) is a simple sports competition, where each team plays against every other team once. The RRT is a competition framework adopted in many professional sports such as soccer and basketball, especially in Europe. In this study, we consider an RRT schedule with the following properties:

- Each team plays one match in each *slot*, i.e., the day when a match is held.
- Each team has a home stadium, and each match is held at the home stadium of one of the two playing teams. When a team plays a match at the home of the opponent, we state that the team plays away.
- Each team plays each other team exactly once at home or away.

It is considered undesirable that a team plays consecutive away or home matches in order to reduce player's burden; such an occurrence is called a *break*.

Accordingly, the number of breaks in a tournament should be reduced. To construct a tournament schedule with fewer breaks, the following two decomposition approaches are widely used: (i) first-break-then-schedule [1–7] and (ii) first-schedule-then-break [1, 8–12]. In the first-break-then-schedule approach, an HA-assignment is generated in the first stage, which assigns a home game or an away game to each team in each slot. The second stage finds a timetable consistent with the generated HA-assignment, if it exists. Contrarily, in the first-schedule-then-break approach, a timetable is constructed at the first stage. The second stage determines the home and away teams of each match. In this approach, the home advantage is further determined in the second stage. For detailed information, refer to [13]. In this study, we focus on a problem occurring in the second stage of the first-schedule-then-break approach. This problem is commonly called the break minimization problem, which determines an HA-assignment minimizing the number of breaks.

An integer linear programming (ILP) formulation of the break minimization problem was proposed by Trick [8]. Recent development of the general ILP solvers enables to solve larger problems dramatically [14, 15]. The enhanced presolving and cutting planes technique contributed to improve the dual bound, and the various heuristics search within branch-and-bound contributed to the primal bound. Nevertheless, the efficiency of ILP methods is significantly affected by problem formulations. The difference in formulations affects not only the computational effort required to solve LP relaxation but also the process of finding primal solutions or improving dual bound by cutting planes. Thus, investigating better formulations is crucial to improve the computational performance.

In this study, we propose a novel ILP formulation called the bigram based formulation for the break minimization problem. We demonstrate its advantage over the well-known ILP formulation proposed by Trick [8] through computational experiments. We also investigate the valid inequalities of bigram based formulation, which further improves the computational performance.

The remainder of this paper is organized as follows. Section 2 presents a literature review of the break minimization problem. Section 3 introduces our bigram based formulation and its valid inequalities. Section 4 presents the computational results of the bigram based formulation and evaluation of its valid inequalities. The Appendix provided after the main sections presents detailed information regarding Trick's ILP formulation. We also compare our formulation with the QUBO formulation, which is solved using

[†]The authors are with Department of Industrial Engineering and Economics at Tokyo Institute Technology University.
a) E-mail: fujii@msi.co.jp
b) E-mail: matsui.t.af@m.titech.ac.jp

a general mixed integer nonlinear solver and a specialized QUBO solver.

2. Literature Review

The research on the break minimization problem (BMP), introduced by the seminal work of de Werra [16], has steadily evolved. By $B_{\min}(\tau)$, we denote the minimum total number of breaks over all possible HA-assignments for a given timetable τ of an RRT with $2n$ teams. For the lower bound of $B_{\min}(\tau)$, de Werra [16] demonstrated that any schedule of a round-robin tournament has at least $2n-2$ breaks. For the upper bound, Miyashiro and Matsui [9] proposed an algorithm for finding an HA-assignment satisfying the condition that the number of breaks is less than or equal to $n(n-1)$ for a given timetable. Post and Woeginger [11] revised this analysis and improved the upper bound of the number of breaks from $n(n-1)$ to $(n-1)^2$ if the number of teams $2n$ is not a multiple of 4.

Miyashiro and Matsui [9] have proposed a polynomial time algorithm for deciding whether there exists an HA-assignment with exactly $2n-2$ breaks for a given timetable. They also demonstrated the equivalence of the break minimization and break maximization problems. Elf, Jünger, and Rinaldi [17] conjectured that the problem of finding $B_{\min}(\tau)$ is NP-hard, which have not been proven yet to the best of our knowledge.

Régin [18] proposed a constraint programming model for BMP, which was able to solve instances containing up to 20 teams. Trick [8] proposed an ILP formulation for BMP in an RRT, which was able to solve instances containing up to 22 teams. Van Hentenryck and Vergados [19] proposed a simulated annealing algorithm for BMP to rapidly obtain near-optimal solutions. Elf, Jünger, and Rinaldi [17] showed that BMP could be transformed into a maximum cut problem (MAX CUT); their proposed methods were able to solve instances of up to 26 teams. Miyashiro and Matsui [10] formulated BMP as MAX RES CUT. They also applied an approximation algorithm for MAX RES CUT based on the semi-definite programming relaxation, which is proposed by Goemans and Williamson [20]. Recently, Peng, Clark, and Dahbura [21] applied reinforcement learning to a first-schedule-then-break approach and analyzed the behavior of agents in home-away assignments. Kuramata, Katsuki, and Nakata [22] applied quantum annealing, one of the quantum techniques, in a heuristic manner to rapidly obtain sub-optimal solutions of BMP and compared their results with those obtained using the ILP formulation proposed by Trick.

3. Break Minimization Description and Novel ILP formulation

3.1 Preliminaries

In this section, we introduce some notations and formal definitions. We use the following notations and symbols throughout this paper:

- $2n$: number of teams, where $2n \geq 4$,
- $T = \{1, 2, \ldots, 2n\}$: set of teams,
- $S = \{1, 2, \ldots, 2n-1\}$: set of slots.

A schedule of an RRT is described as a pair of a timetable and an HA-assignment, which is defined below. In this study, we assume that a timetable τ of an RRT is a matrix whose rows and columns are indexed by T and S, respectively. An element $\tau(t,s)$ denotes the opponent that plays against team t at slot s. A timetable τ (of an RRT schedule) should satisfy the following conditions: (i) a row of τ indexed by team $t \in T$ is a permutation of teams in $T \setminus \{t\}$, and (ii) $\tau(\tau(t,s),s) = t$ $(\forall(t,s) \in T \times S)$. Table 1 shows a timetable for an RRT schedule of six teams.

Table 1 Timetable

Slot	**1**	**2**	**3**	**4**	**5**
team 1	3	4	2	5	6
team 2	6	3	1	4	5
team 3	1	2	5	6	4
team 4	5	1	6	2	3
team 5	4	6	3	1	2
team 6	2	5	4	3	1

Table 2 HA-assignment

Slot	**1**	**2**	**3**	**4**	**5**
team 1	0	0	1	1	1
team 2	0	0	0	1	0
team 3	1	1	0	0	0
team 4	0	1	0	0	1
team 5	1	0	1	0	0
team 6	1	1	1	1	0

A team is defined to be at *home* in slot s if the team plays a match at its home stadium in s; otherwise *away* in s. An *HA-assignment* (*home-away assignment*) is a 0-1 matrix $\mathbf{Z} = \{z_{t,s}\}$ whose rows and columns are indexed by T and S, respectively. A value of $z_{t,s}$ is equal to 1 if team t plays a match in slot s at home; otherwise, it is 0. For a given timetable τ, we say that $\mathbf{Z}$ is *consistent* with τ when $z_{t,s} = 1 - z_{\tau(t,s),s}$ holds for each $(t,s) \in T \times S$. Table 2 shows an example of an HA-assignment which is consistent with the timetable presented in Table 1.

Given an HA-assignment $\mathbf{Z}$, we say that team t has a break at slot $s \in S \setminus \{1\}$ if $z_{t,s-1} = z_{t,s}$ holds. For example, in Table 2, team 1 has a break at slot 1, as there are consecutive away matches $(z_{1,1} = z_{1,2} = 0)$. The number of breaks in a home–away assignment is defined as the total number of breaks belonging to all teams.

Following the definitions introduced above, we are now ready to present a formal definition of BMP:
Break Minimization Problem: *Given a timetable τ, the break minimization problem finds an HA-assignment consistent with τ that minimizes the number of breaks.*

3.2 A bigram based formulation

In this section, we propose a novel ILP formulation, called a *bigram based formulation*, for BMP.
Decision Variables
In our model, each team is represented as being in one of two states in each slot: "home" (H) or "away" (A). When considering consecutive slots, each team has one of the four possible sequential pairs of states: home-home (HH), home-away (HA), away-home (AH), and away-away (AA). To represent these state pairs, we introduce four binary variables;

$x^{\mathrm{HH}}_{t,s}$, $x^{\mathrm{HA}}_{t,s}$, $x^{\mathrm{AH}}_{t,s}$, and $x^{\mathrm{AA}}_{t,s}$. The value of each variable is determined based on the corresponding states of team t across two consecutive slots, namely slots s and $s+1$. Note that these variables are defined for $(t,s) \in T \times S^-$, where we define $S^- = S \setminus \{2n-1\}$. The variable $x^{\mathrm{HH}}_{t,s}$ is set to 1 if the team is in the HH state and 0 otherwise. Similarly, $x^{\mathrm{HA}}_{t,s}$, $x^{\mathrm{AH}}_{t,s}$, and $x^{\mathrm{AA}}_{t,s}$ are set to 1 if the team t is in the HA, AH, and AA states, respectively, and 0 otherwise. This binary encoding facilitates a straightforward and efficient representation of the team's state transitions. Owing to representing all pairs of states across consecutive slots, we refer to our formulation as a *bigram based formulation*. This name is inspired by the concept of a "bigram" in linguistic theory, in which it refers to a sequence of two adjacent elements. We note that Trick's ILP formulation, which also defines status transition variables, lacks two out of the four possible status transitions: AA and HH.

In the following, we describe the constraints of our bigram based formulation.

0-1 Constraint

As all the variables are binary, we have the following:

$$x^{\mathrm{HH}}_{t,s}, x^{\mathrm{HA}}_{t,s}, x^{\mathrm{AH}}_{t,s}, x^{\mathrm{AA}}_{t,s} \in \{0,1\} \quad (\forall (t,s) \in T \times S^-). \tag{1}$$

Single State Assignment Constraints

Every team is required to play in one of four distinct modes for each pair of consecutive slots: away-away (AA), home-home (HH), away-home (AH), and home-away (HA). This condition is expressed as:

$$x^{\mathrm{AA}}_{t,s} + x^{\mathrm{HH}}_{t,s} + x^{\mathrm{AH}}_{t,s} + x^{\mathrm{HA}}_{t,s} = 1 \quad (\forall (t,s) \in T \times S^-). \tag{2}$$

Sequencing Constraint

Because each team must assume exactly one state, home or away for each slot, Sequencing Constraint is modeled as:

$$x^{\mathrm{HA}}_{t,s} + x^{\mathrm{AA}}_{t,s} = x^{\mathrm{AH}}_{t,s+1} + x^{\mathrm{AA}}_{t,s+1} \quad (\forall (t,s) \in T \times S^-), \tag{3}$$

$$x^{\mathrm{AH}}_{t,s} + x^{\mathrm{HH}}_{t,s} = x^{\mathrm{HA}}_{t,s+1} + x^{\mathrm{HH}}_{t,s+1} \quad (\forall (t,s) \in T \times S^-). \tag{4}$$

However, constraint (4) can be omitted, as it is implied by constraints (2) and (3).

HA-Consistency Constraint

To obtain an HA-assignment that is consistent with a given timetable, we introduce the HA-Consistency Constraint. This constraint is expressed as follows:

$$\begin{aligned} &x^{\mathrm{AA}}_{t,s} + x^{\mathrm{AH}}_{t,s} = x^{\mathrm{HA}}_{\tau(t,s),s} + x^{\mathrm{HH}}_{\tau(t,s),s} (\forall (t,s) \in T \times S^-),\\ &x^{\mathrm{AA}}_{t,2n-2} + x^{\mathrm{HA}}_{t,2n-2} = x^{\mathrm{AH}}_{\tau(t,2n-1),2n-2} + x^{\mathrm{HH}}_{\tau(t,2n-1),2n-2}\\ &\hfill (\forall t \in T). \end{aligned} \tag{5}$$

Symmetry Breaking Constraint

Reversing all the home and away assignments in an optimal solution should also yield an optimal solution. Therefore, we can impose the restriction that one of the teams must play at home in slot 1. This constraint is expressed as follows:

$$x^{\mathrm{HH}}_{1,1} + x^{\mathrm{HA}}_{1,1} = 1$$

Now we have a bigram based formulation (BBF) for BMP:

$$\begin{aligned} \text{BBF: minimize } &\sum_{t\in T}\sum_{s\in S^-}(x^{\mathrm{HH}}_{t,s} + x^{\mathrm{AA}}_{t,s})\\ \text{subject to } &(1),(2),(3),(5), \end{aligned}$$

which minimizes the number of breaks.

3.3 Valid Inequalities

In this section, we propose a valid inequality for our BBF. For any pair of different teams $t_1, t_2 \in T$, $s(t_1,t_2)$ indicates the slot in which teams t_1 and t_2 have a match. Clearly, we have $s(t_1,t_2) = s(t_2,t_1)$. We define the set Φ as follows:

$$\Phi := \{(t_1,t_2,t_3) \in T^3 \mid s(t_1,t_2) < s(t_2,t_3) < s(t_1,t_3)\}.$$

In the above definition, we assume that each triplet $(t_1,t_2,t_3) \in \Phi$ ensures that these three teams are mutually different.

Theorem 3.1. *For any $(t_1,t_2,t_3) \in \Phi$, the following inequality:*

$$\begin{aligned} &\left(-x^{\mathrm{HH}}_{t_1,s_1} + \sum_{s=s_1+1}^{s_3-1} x^{\mathrm{HA}}_{t_1,s}\right) + \left(-x^{\mathrm{HH}}_{t_2,s_1} + \sum_{s=s_1+1}^{s_2-1} x^{\mathrm{HA}}_{t_2,s}\right)\\ &\quad + \left(-x^{\mathrm{HH}}_{t_3,s_2} + \sum_{s=s_2+1}^{s_3-1} x^{\mathrm{HA}}_{t_3,s}\right) \le s_3 - s_1 - 2 \end{aligned} \tag{6}$$

is a valid inequality of the BBF where $s_1 = s(t_1,t_2), s_2 = s(t_2,t_3), s_3 = s(t_1,t_3)$.

First, we present the following two lemmas.

Lemma 3.2. *For any $t \in T$ and $\forall (s', s'') \in S^- \times S$ satisfying that $s'' - s'$ is an even positive, any feasible solution $\boldsymbol{x}$ of the BBF satisfies*

$$-x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-1} x^{\mathrm{HA}}_{t,s} \le \frac{s''-s'}{2}. \tag{7}$$

When the equality is satisfied, team t plays away games at both slots s' and s''.

Proof of Lemma 3.2: We will prove this by induction. First we assume that $s'' - s' = 2$. The inequality (7) holds since $-x^{\mathrm{HH}}_{t,s'} + x^{\mathrm{HA}}_{t,s'+1} \le 1$. When the equality satisfied, then $x^{\mathrm{HH}}_{t,s'} = 0$ and $x^{\mathrm{HA}}_{t,s'+1} = 1$ hold. This indicates $x^{\mathrm{AH}}_{t,s'} = 1$, which proves the lemma for this case. Next, we assume that the lemma holds for $s'' - s' = 2k$ where $k > 0$. We will prove that the lemma holds for $s'' - s' = 2k+2$ under this assumption. The inequality (7) holds since

$$\begin{aligned} &-x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-1} x^{\mathrm{HA}}_{t,s} = -x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-3} x^{\mathrm{HA}}_{t,s} + \sum_{s=s''-2}^{s''-1} x^{\mathrm{HA}}_{t,s}\\ &\le k + x^{\mathrm{HA}}_{t,s''-2} + x^{\mathrm{HA}}_{t,s''-1} \le k+1. \end{aligned}$$

When the equality is satisfied, then the equalities $-x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-3} x^{\mathrm{HA}}_{t,s} = k$ and $\sum_{s=s''-2}^{s''-1} x^{\mathrm{HA}}_{t,s} = 1$ hold since $\sum_{s=s''-2}^{s''-1} x^{\mathrm{HA}}_{t,s} \leq 1$. Then, team t plays away games at slot s' and slot $s''-2$ by the assumption. The equality $x^{\mathrm{HA}}_{t,s''-1} = 1$ holds and therefore team t plays away game at slot s''. This proves the lemma. ■

Lemma 3.3. *For any $t \in T$ and $\forall (s', s'') \in S^- \times S$ satisfying that $s'' - s'$ is an odd positive, any feasible solution $\boldsymbol{x}$ of the BBF satisfies*

$$-x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-1} x^{\mathrm{HA}}_{t,s} \leq \frac{s''-s'-1}{2}. \tag{8}$$

When the equality is satisfied, team t plays (at least one) away game at either slot s' or slot s''.

<u>Proof of Lemma 3.3:</u> First we assume that $s'' - s' = 1$. The inequality (8) holds since $-x^{\mathrm{HH}}_{t,s'} \leq 0$. When the equality satisfied, then $x^{\mathrm{HH}}_{t,s'} = 0$ holds. This indicates team t plays at least one away game at either slot s' or slot s''. Next, we assume that the lemma holds for $s''-s' = 2k+1$ where $k > 0$. We will prove that the lemma holds for $s''-s' = 2k+3$ under this assumption. The inequality (8) holds by the assumption since

$$\begin{aligned} &- x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-1} x^{\mathrm{HA}}_{t,s} = -x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-3} x^{\mathrm{HA}}_{t,s} + \sum_{s=s''-2}^{s''-1} x^{\mathrm{HA}}_{t,s} \\ &\leq k + x^{\mathrm{HA}}_{t,s''-2} + x^{\mathrm{HA}}_{t,s''-1} \leq k+1. \end{aligned}$$

When the equality is satisfied, then the equalities $-x^{\mathrm{HH}}_{t,s'} + \sum_{s=s'+1}^{s''-3} x^{\mathrm{HA}}_{t,s} = k$ and $\sum_{s=s''-2}^{s''-1} x^{\mathrm{HA}}_{t,s} = 1$ hold since $\sum_{s=s''-2}^{s''-1} x^{\mathrm{HA}}_{t,s} \leq 1$. Then, team t plays away games at slot s' or slot $s''-2$ by the assumption. If team t does not plays at slot s', then the equality $x^{\mathrm{HA}}_{t,s''-1} = 1$ holds and therefore team t plays away game at slot s''. This proves the lemma. ■

Now we prove the validity of the inequality (6).

<u>Proof of Theorem 3.1:</u> Because $(s_3 - s_1) + (s_2 - s_1) + (s_3 - s_2) = 2s_3 - 2s_1$ is an even integer, $\{s_3 - s_1, s_2 - s_1, s_3 - s_2\}$ consists of three even numbers or includes exactly two odd numbers.

<u>Case 1:</u> The set $\{s_3 - s_1, s_2 - s_1, s_3 - s_2\}$ consists of three even numbers. Lemma 3.2 implies that the left-hand side of the inequality (6) is less than or equal to $((s_3 - s_1) + (s_2 - s_1) + (s_3 - s_2))/2 = s_3 - s_1$. Assume, on the contrary, that the left hand side of (6) is strictly greater than $s_3 - s_1 - 2$, i.e., either $s_3 - s_1$ or $s_3 - s_1 - 1$. Lemma 3.2 states that at least two teams play two away games in the three matches among teams $\{t_1, t_2, t_3\}$. Thus, there are at least four away games in the three matches among $\{t_1, t_2, t_3\}$. This is a contradiction.

<u>Case 2:</u> The set $\{s_3 - s_1, s_2 - s_1, s_3 - s_2\}$ includes exactly two odd numbers. Lemmas 3.2 and 3.3 imply that the left-hand side of the inequality (6) is less than or equal to $((s_3 - s_1) + (s_2 - s_1) + (s_3 - s_2) - 2)/2 = s_3 - s_1 - 1$. Assume, on the contrary, that the left hand side of (6) is equal to $s_3 - s_1 - 1$. Lemmas 3.2 and 3.3 indicate that in the three matches among teams $\{t_1, t_2, t_3\}$, every team plays at least one away game and at least one team plays two away games. Thus, there are at least four away games in the three matches among $\{t_1, t_2, t_3\}$. This is a contradiction. ■

From Theorem 3.1, we can derive the following corollary.

Corollary 3.4. *When (s_1, s_2, s_3) is a sequence of consecutive three integers, we obtain the following inequality:*

$$x^{\mathrm{HA}}_{t_1,s_1+1} \leq x^{\mathrm{HH}}_{t_1,s_1} + x^{\mathrm{HH}}_{t_2,s_1} + x^{\mathrm{HH}}_{t_3,s_2} \tag{9}$$

is a valid inequality of the BBF.

We present an example of the valid inequality (9) using the timetable depicted in Table 1. For $(1,2,5) \in T^3$, where $(5,1,2) \in \Phi$ and $(s_1, s_2, s_3) = (2,3,4)$, the corresponding valid inequality can be expressed as $x^{\mathrm{HA}}_{5,3} \leq x^{\mathrm{HH}}_{5,2} + x^{\mathrm{HH}}_{1,2} + x^{\mathrm{HH}}_{2,3}$.

4. Numerical Results

In this section, we describe our assessment of the bigram based formulation for BMP. Through computational experiments, we will demonstrate its effectiveness compared with the well-known ILP formulation.

SCIP [23] is a non-commercial solver, which works as a general framework based on branching for constraint integer and mixed integer programming using branch-cut-and-price. The methods for processing various constraints can be implemented through constraint handlers, and advanced methods like primal heuristics, branching rules, and cutting plane separators can be integrated as plugins. In addition to plugins supplied as part of the SCIP distribution, new plugins can be created by users. This functionality is particularly advantageous in cases where users wish to manually incorporate cutting planes, as demonstrated in this study. We used the Python interface PySCIPOpt with SCIP 8.0.3 in our experiments for evaluating the performance of different formulations.

The experiments were performed on a Linux machine with AMD Ryzen 9 5900X 12-Core Processor at 3.7 GHz, 64GB of RAM, Ubuntu 22.04 LTS.

4.1 Test set

Although there is a long history of applying optimization techniques to sports scheduling problems, there has been a lack of standardized benchmarks. This has made it difficult for researchers to fairly compare new techniques for solving problems. From the perspective of practitioners, evaluating proposed methods has also been challenging. Van Bulck, Goossens, Schönberger, and Guajardo [24] have made significant efforts to address this issue. They developed a project called RobinX, aimed to create a standardized benchmark by gathering test sets and making them publicly available on a website. For details, please refer to [24]. It includes 122 instances of BMP, named TC_BM instances. We selected all of them for the test set.

4.2 Comparison of Trick's formulation and bigram based formulation

In this section, we compare the results of Trick's formulation [8] (TF) and bigram based formulation (BBF). Both formulations were implemented via PySCIPOpt. We set the time limit to 3,600 seconds.

Table 3 presents the comparison results. In this table, we compare two Integer Linear Programming (ILP) formulations, namely "TF" and "BBF," in terms of their performance on TC_BM instances.

The first column indicates the number of the team. The second and third columns, under the header "#variables," display the average number of variables in the presolved model. The average numbers of variables in the original model are shown in parentheses. The fourth and fifth columns, under the header "#constraints," display the average number of constraints in the presolved model. The average numbers of columns in the original model are also shown in parentheses. ILP solver typically simplifies the problem by identifying and eliminating redundant variables and constraints. We have presented them because the size of the reduced model indicates the intrinsic size. The sixth and seventh columns, labeled "#solved," show the number of problems successfully solved out of the total attempted problems for each formulation. For example, "4/5" means that four out of five instances were solved within the time limit. The eighth and ninth columns, labeled "TF" and "BBF," respectively, under the header "#nodes," display the average number of nodes explored in the branch-and-bound algorithm. The tenth and eleventh columns, under the header "run time [s]," indicate the average computational time (seconds) required to solve the problems. If the solving process reaches the time limit, the reported number of nodes and computational time represent the values at the end of the interrupted solving. The twelfth and thirteenth columns, under the header "gap(%)," indicate the average gap at the end of solving. If the instance is solved within the time limit, then the gap is 0.00%.

This table indicates the problem size and performance differences between the two ILP formulations across various problem sizes. The number of (reduced) constraints in the bigram based formulation is smaller than that in TF, while the number of (reduced) variables is larger. The performance differences indicate that BBF outperforms TF regarding success rates when solving instances and the average runtime. For example, for $2n = 30$, bigram based formulation successfully solved three out of five instances, while TF failed to solve any instances. Although no instances could be solved within the time limit for $2n \geq 34$, BBF still outperforms TF regarding the average gap.

There are two reasons for the effectiveness of BBF. First, the size of the tree generated during the branch-and-bound process is much smaller, as seen under the '#nodes' header of Table 3. Second reason is its advantage in solving LP relaxation. To observe the difference in solving LP relaxations, we conducted experiments with two settings: disabling all cutting plane separators and imposing a node limit of 1,000. Disabling all cutting plane separators means that the solver is imposed not to generate valid inequalities during the solution process. Differences in LP solving can arise from variations in two key aspects: the additional valid inequalities and the size of the branch and bound tree. Therefore, we restricted them so that we can observe the differences without any side effects. Figure 1 shows the comparison of LP iterations per node. The graph presents a comparative analysis of the number of iterations required by the simplex method in two different formulations. The x-axis denotes the size of the instances, $2n$. The y-axis indicates the average number of simplex method iterations per node. The average is calculated across several instances. The result indicates that BBF requires fewer LP iterations when solving nodes. The relaxation problem of BBF has advantages in the perspective of solving node relaxation via simplex method, despite having more decision variables.

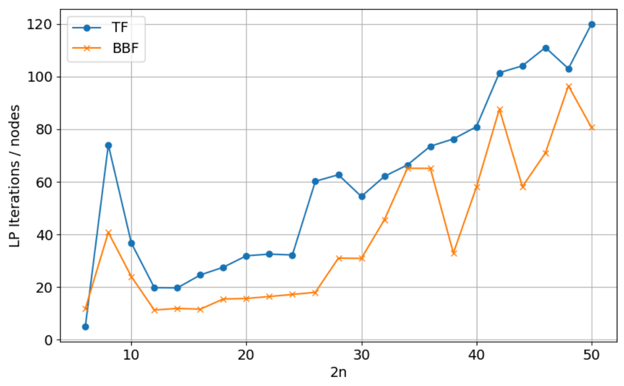

Fig. 1 LP iteration per node

4.3 Assessment of Valid Inequality

In this section, we investigate the effectiveness of the valid inequality (9) for the proposed BBF. The valid inequalities were implemented via separator plugins of PySCIPTOpt. This callback is executed during the subproblem processing. The user can control the frequency of these calls. We set the frequency to 1 so that this callback should be called every time in the branch-and-bound loop. Inside the callback, we add the valid inequalities that violate the LP solution by more than a certain amount.

Generally, the valid inequalities help to increase the relaxation value. However, they also increase the computational effort required to solve each node. In particular, adding dense inequalities can significantly slow down the simplex method. We conducted computational experiments by varying the density of valid inequalities. We found that adding only most sparse valid inequality (9) achieved the best performance. Consequently, we compared the results of BBF with them. For this experiment, we set the time limit to 7,200 seconds.

Table 3 Trick's formulation vs. the proposed bigram based formulation

$2n$	#variables		#constraints		#solved		#nodes		run time [s]		gap(%)	
	TF	BBF	TF	BBF	TF	BBF	TF	BBF	TF	BBF		
4	17(40)	24(59)	0(32)	0(19)	5/5	5/5	1	0	0.00	0.00	0.00	0.00
6	58(96)	88(166)	61(96)	42(58)	5/5	5/5	1	1	0.01	0.01	0.00	0.00
8	119(176)	184(325)	143(192)	101(117)	5/5	5/5	1	1	0.06	0.03	0.00	0.00
10	200(280)	312(536)	255(320)	180(196)	5/5	5/5	1	1	0.26	0.10	0.00	0.00
12	301(408)	472(799)	393(480)	279(295)	5/5	5/5	2	2	0.53	0.28	0.00	0.00
14	422(560)	664(1,114)	560(672)	398(414)	5/5	5/5	9	12	2.52	0.95	0.00	0.00
16	563(736)	888(1,481)	756(896)	537(553)	6/6	6/6	27	17	4.90	2.25	0.00	0.00
18	724(936)	1,144(1,900)	979(1,152)	696(712)	6/6	6/6	224	377	11.34	6.70	0.00	0.00
20	905(1,160)	1,432(2,371)	1,230(1,440)	875(891)	5/5	5/5	632	562	33.65	13.65	0.00	0.00
22	1,106(1,408)	1,752(2,894)	1,509(1,760)	1,074(1,090)	5/5	5/5	1,629	1,460	113.58	40.13	0.00	0.00
24	1,327(1,680)	2,104(3,469)	1,816(2,112)	1,293(1,309)	5/5	5/5	4,399	1,668	426.58	75.56	0.00	0.00
26	1,568(1,976)	2,488(4,096)	2,151(2,496)	1,532(1,548)	4/5	5/5	*13,093	3,156	1,551.12	257.36	1.21	0.00
28	1,829(2,296)	2,904(4,775)	2,515(2,912)	1,793(1,807)	0/5	5/5	*24,102	9,326	3,600.00	1,088.89	6.72	0.00
30	2,110(2,640)	3,352(5,506)	2,906(3,360)	2,072(2,086)	0/5	3/5	*17,665	*14,939	3,600.00	2,670.18	14.35	2.26
32	2,411(3,008)	3,832(6,289)	3,325(3,840)	2,371(2,385)	0/5	1/5	*10,816	*16,147	3,600.00	3,265.70	17.83	5.27
34	2,732(3,400)	4,344(7,124)	3,773(4,352)	2,691(2,704)	0/5	0/5	*6,483	*12,919	3,600.00	3,600.00	30.61	11.27
36	3,073(3,816)	4,888(8,011)	4,249(4,896)	3,031(3,043)	0/5	0/5	*3,930	*10,094	3,600.00	3,600.00	33.62	13.90
38	3,434(4,256)	5,464(8,950)	4,752(5,472)	3,391(3,402)	0/5	0/5	*3,834	*5,873	3,600.00	3,600.00	51.49	23.75
40	3,815(4,720)	6,072(9,941)	5,285(6,080)	3,770(3,781)	0/5	0/5	*3,532	*4,718	3,600.00	3,600.00	65.84	29.40
42	4,216(5,208)	6,712(10,984)	5,843(6,720)	4,169(4,180)	0/5	0/5	*2,498	*5,074	3,600.00	3,600.00	73.48	44.91
44	4,637(5,720)	7,384(12,079)	6,431(7,392)	4,588(4,599)	0/5	0/5	*2,156	*3,476	3,600.00	3,600.00	95.80	57.06
46	5,078(6,256)	8,088(13,226)	7,047(8,096)	5,028(5,038)	0/5	0/5	*1,498	*3,313	3,600.00	3,600.00	101.70	55.68
48	5,539(6,816)	8,824(14,425)	7,690(8,832)	5,487(5,497)	0/5	0/5	*1,156	*2,312	3,600.00	3,600.00	103.34	69.40
50	6,020(7,400)	9,592(15,676)	8,361(9,600)	5,966(5,976)	0/5	0/5	*1,178	*2,112	3,600.00	3,600.00	121.55	79.37

* Underestimated as some instances could not be solved within the time limit.

Table 4 reports the comparison results. The table headers represent the same categories of data as in Table 3. For instance, "#solved" indicates the number of problems that were successfully solved out of the given instances. The tenth and eleventh columns, under the header "root gap(%)," indicate the average gap at the root node. We have compared the dual bound obtained at the root node by normalizing the optimal value or known best value. It is defined by $\mathrm{db}(\tau)/B_{\min}(\tau)$, where $\mathrm{db}(\tau)$ denotes the dual bound at root node and $B_{\min}(\tau)$ denotes the optimal value or best known value. A larger value indicates that we obtained a tighter lower bound at the root node. In the table, "BBF" corresponds to the results of BBF without valid inequalities, while "BBF-V" represents the results of BBF with valid inequalities.

Table 4 indicates that the valid inequality (9) improves the result on some instances. For example, when $2n = 30$, the BBF with valid inequalities solved 24% faster on average than the BBF without valid inequalities Though "root gap(%)" is certainly improved when valid inequalities are added, we can conclude that the effect of the valid inequalities is modest. We guess that this result comes from the duplication of valid inequalities automatically generated by the solver and those we generated.

5. Conclusion

In this study, we developed a novel ILP formulation for the break minimization problem of the (single) round-robin tournament, which we call the bigram based formulation. Through computational experiments, we have demonstrated that our formulation outperforms a well-known existing formulation. The experiments revealed two advantages of BBF; smaller tree size and increased efficiency in solving LP relaxation.

We also presented valid inequalities of this formulation. We confirmed that their addition further improves the results, especially for larger instances. By carefully choosing sparse ones, we achieve better root gaps while avoiding an increase in the computational effort required to solve each node.

The break minimization problem we applied is the most simple one; in practice, there are many constraints. One of the most common restrictions is an upper bound of consecutive home games or away games [25]. It appears as tournament problem of Danish soccer league [13]. We believe that our methods hold significant potential even for such practical problems. Future research will focus on verifying the effectiveness of our approach by imposing practical constraints on real-world applications.

6. Acknowledgments

This research was supported by JSPS KAKENHI Grant Numbers JP20K04973.

References

[1] G.L. Nemhauser and M.A. Trick, "Scheduling a major college basketball conference," Operations Research, vol.46, no.1, pp.1–8, 1998.

[2] D. Briskorn, "Feasibility of home–away-pattern sets for round robin tournaments," Operations Research Letters, vol.36, no.3, pp.283–284, 2008.

[3] R. Miyashiro, H. Iwasaki, and T. Matsui, "Characterizing feasible pattern sets with a minimum number of breaks," in PATAT 2002, Lecture Notes in Computer Science, vol.2740, pp.78–99, Springer,

Table 4 Bigram based formulation with valid inequalities

$2n$	#solved		#nodes		run time [s]		gap (%)		root gap (%)	
	BBF	BBF+V	BBF	BBF+V	BBF	BBF+V	BBF	BBF+V	BBF	BBF+V
4	5/5	5/5	1	1	0.01	0.01	0.00	0.00	100.00	100.00
6	5/5	5/5	1	1	0.01	0.01	0.00	0.00	100.00	100.00
8	5/5	5/5	1	1	0.03	0.03	0.00	0.00	100.00	100.00
10	5/5	5/5	1	1	0.10	0.11	0.00	0.00	100.00	100.00
12	5/5	5/5	2	2	0.28	0.28	0.00	0.00	97.17	98.22
14	5/5	5/5	12	9	0.95	0.53	0.00	0.00	93.09	94.72
16	6/6	6/6	17	15	2.25	1.89	0.00	0.00	93.00	93.91
18	6/6	6/6	377	325	6.70	6.79	0.00	0.00	88.07	90.13
20	5/5	5/5	562	634	13.65	15.74	0.00	0.00	83.38	86.53
22	5/5	5/5	1460	810	40.13	32.58	0.00	0.00	81.87	83.30
24	5/5	5/5	1668	1594	75.56	80.35	0.00	0.00	79.59	81.88
26	5/5	5/5	3156	3215	257.36	268.67	0.00	0.00	78.84	80.07
28	5/5	5/5	9326	6588	1092.06	888.93	0.00	0.00	75.65	78.48
30	5/5	5/5	17127	13120	3335.12	2682.59	0.00	0.00	75.91	76.48
32	3/5	2/5	*26982	*23971	5941.70	5353.68	1.80	3.09	74.43	75.03
34	0/5	0/5	*24618	*24831	7200.00	7200.00	7.75	9.36	71.76	71.36
36	0/5	0/5	*22341	*20602	7200.00	7200.00	8.76	9.88	70.76	71.69
38	0/5	0/5	*11703	*13298	7200.00	7200.00	16.27	19.09	66.96	67.56
40	0/5	0/5	*9458	*7126	7200.00	7200.00	22.40	19.95	65.69	66.25
42	0/5	0/5	*9290	*7524	7200.00	7200.00	35.83	30.06	61.74	62.09
44	0/5	0/5	*6177	*5867	7200.00	7200.00	49.34	53.67	57.44	58.00
46	0/5	0/5	*6088	*6652	7200.00	7200.00	43.39	51.38	58.48	59.39
48	0/5	0/5	*4269	*4230	7200.00	7200.00	56.16	52.29	55.12	55.92
50	0/5	0/5	*4250	*4145	7200.00	7200.00	71.77	69.28	52.49	52.84

* Underestimated as some instances could not be solved within the time limit.

2002.

[4] M. Henz, "Scheduling a major college basketball conference—revisited," Operations Research, vol.49, no.1, pp.163–168, 2001.

[5] D. Van Bulck and D. Goossens, "Optimizing rest times and differences in games played: an iterative two-phase approach," Journal of Scheduling, vol.25, no.3, pp.261–271, 2022.

[6] L. Zeng and S. Mizuno, "On the separation in 2-period double round robin tournaments with minimum breaks," Computers & Operations Research, vol.39, no.7, pp.1692–1700, 2012.

[7] D. Van Bulck and D. Goossens, "On the complexity of pattern feasibility problems in time-relaxed sports timetabling," Operations Research Letters, vol.48, no.4, pp.452–459, 2020.

[8] M.A. Trick, "A schedule-then-break approach to sports timetabling," in PATAT 2000, Lecture Notes in Computer Science, vol.2079, pp.242–253, Springer, 2000.

[9] R. Miyashiro and T. Matsui, "A polynomial-time algorithm to find an equitable home–away assignment," Operations Research Letters, vol.33, no.3, pp.235–241, 2005.

[10] R. Miyashiro and T. Matsui, "Semidefinite programming based approaches to the break minimization problem," Computers & Operations Research, vol.33, no.7, pp.1975–1982, 2006.

[11] G. Post and G.J. Woeginger, "Sports tournaments, home–away assignments, and the break minimization problem," Discrete Optimization, vol.3, no.2, pp.165–173, 2006.

[12] R.V. Rasmussen and M.A. Trick, "The timetable constrained distance minimization problem," in CPAIOR 2006, Lecture Notes in Computer Science, vol.3990, pp.167–181, Springer, 2006.

[13] R.V. Rasmussen, "Scheduling a triple round robin tournament for the best danish soccer league," European Journal of Operational Research, vol.185, no.2, pp.795–810, 2008.

[14] T. Achterberg and R. Wunderling, "Mixed integer programming: Analyzing 12 years of progress," in Facets of combinatorial optimization: Festschrift for Martin Grötschel, pp.449–481, Springer, 2013.

[15] T. Koch, T. Berthold, J. Pedersen, and C. Vanaret, "Progress in mathematical programming solvers from 2001 to 2020," EURO Journal on Computational Optimization, vol.10, p.100031, 2022.

[16] D. de Werra, "Scheduling in sports," Studies on Graphs and Discrete Programming, vol.11, pp.381–395, 1981.

[17] M. Elf, M. Jünger, and G. Rinaldi, "Minimizing breaks by maximizing cuts," Operations Research Letters, vol.31, no.5, pp.343–349, 2003.

[18] J.C. Régin, "Minimization of the number of breaks in sports scheduling problems using constraint programming," DIMACS workshop on Constraint Programming and Large Scale Discrete Optimization, pp.115–130, 2000.

[19] P. Van Hentenryck and Y. Vergados, "Minimizing breaks in sport scheduling with local search," ICAPS, pp.22–29, 2005.

[20] M.X. Goemans and D.P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," Journal of the ACM, vol.42, no.6, pp.1115–1145, 1995.

[21] J. Peng, A.D. Clark, and A. Dahbura, "Introducing human corrective multi-team SRR sports scheduling via reinforcement learning," 2021 IEEE Symposium Series on Computational Intelligence (SSCI), pp.1–7, IEEE, 2021.

[22] M. Kuramata, R. Katsuki, and K. Nakata, "Solving large break minimization problems in a mirrored double round-robin tournament using quantum annealing," Plos one, vol.17, no.4, p.e0266846, 2022.

[23] K. Bestuzheva, M. Besançon, W.K. Chen, A. Chmiela, T. Donkiewicz, J. van Doornmalen, L. Eifler, O. Gaul, G. Gamrath, A. Gleixner, *et al.*, "Enabling research through the SCIP optimization suite 8.0," ACM Transactions on Mathematical Software, vol.49, no.2, pp.1–21, 2023.

[24] D. Van Bulck, D. Goossens, J. Schönberger, and M. Guajardo, "RobinX: A three-field classification and unified data format for round-robin sports timetabling," European Journal of Operational Research, vol.280, no.2, pp.568–580, 2020.

[25] K. Easton, G. Nemhauser, and M. Trick, "Solving the travelling tournament problem: A combined integer programming and constraint programming approach," in PATAT 2002, Lecture Notes in Computer Science, vol.2740, Springer, 2002.

[26] H.L. Urdaneta, J. Yuan, and A.S. Siqueira, "Alternative integer linear and quadratic programming formulations for HA-Assignment problems," Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, vol.6, no.1, 2018.

[27] S. Vigerske and A. Gleixner, "Scip: Global optimization of mixed-

integer nonlinear programs in a branch-and-cut framework," Optimization Methods and Software, vol.33, no.3, pp.563–593, 2018.

[28] D. Rehfeldt, T. Koch, and Y. Shinano, "Faster exact solution of sparse MaxCut and QUBO problems," Mathematical Programming Computation, pp.1–26, 2023.

Appendix A: Trick's ILP formulation for BMP

In this appendix, we introduce the ILP formulation proposed by Trick [8] (TF).

Decision Variables

The variable $start_t$ indicates if team t starts at "home" in the initial time slot. This variable is applicable to each team t, which ranges from 1 to $2n$. A value of $start_t = 1$ suggests that team t starts at "home" and $start_t = 0$ starts at "away."

The variable $to\_home_{t,s}$ indicates the transition of status; whether team t goes home or not after slot s. If $to\_home_{t,s} = 1$, team t plays at away in slot s and at home in slot $s + 1$. Similarly, $to\_away_{t,s}$ indicates whether team t goes away or not after slot s. If $to\_away_{t,s} = 1$, team t plays at home in slot s and at away in slot $s + 1$. The variables $to\_home_{t,s}$ and $to\_away_{t,s}$ are defined for $(t, s) \in T \times S^-$, where we define $S^- = S \setminus \{2n-1\}$. The variables $to\_home_{t,s}$ and $to\_away_{t,s}$ correspond to $x^{\mathrm{AH}}_{t,s}$ and $x^{\mathrm{HA}}_{t,s}$ in bigram based formulation, respectively.

The variable $at\_home_{t,s}$ indicates whether team t is playing at home in time slot s. A value of 1 for $at\_home_{t,s}$ suggests that team t plays at home for that slot, and a value of 0 suggests it plays at away. Based on the definition of the variables $to\_home_{t,s}$ and $to\_away_{t,s}$, the sum of those will be equal to the variable $at\_home_{t,s}$:

$$at\_home_{t,s} = start_t + \sum_{s'=1}^{s-1} (to\_home_{t,s'} - to\_away_{t,s'}) \quad (\forall (t, s) \in T \times S). \tag{A· 1}$$

Though it is mentioned in [8] that the variable $at\_home_{t,s}$ can be omitted, we decided to define it and impose constraints (A· 1) owing to the improved computational performance.

0-1 Constraints As all the variables are binary, we have the following:

$$\begin{aligned} start_t\,(\forall t \in T),\ at\_home_{t,s} \in \{0, 1\} \quad (\forall (t, s) \in T \times S), \\ to\_home_{t,s}, to\_away_{t,s} \in \{0, 1\} \quad (\forall (t, s) \in T \times S^-). \end{aligned} \tag{A· 2}$$

Home-Away Consistency Constraint This constraint ensures that a home-away assignment is consistent with a given timetable. This can be expressed as follows:

$$at\_home_{t,s} + at\_home_{\tau(t,s),s} = 1 \quad (\forall (t, s) \in T \times S). \tag{A· 3}$$

Single Status Constraint For every team t and slot s, the team cannot go both home and away, which is expressed as follows:

$$to\_home_{t,s} + to\_away_{t,s} \leq 1 \quad (\forall (t, s) \in T \times S^-). \tag{A· 4}$$

Status Transition Constraint This constraint ensures that a team cannot make transition to away unless it is currently at home. Similarly, a team cannot make transition to home unless it is currently at away. This constraint is expressed as follows:

$$\begin{aligned} at\_home_{t,s} - to\_away_{t,s} \geq 0 \quad (\forall (t, s) \in T \times S^-), \\ at\_home_{t,s} + to\_home_{t,s} \leq 1 \quad (\forall (t, s) \in T \times S^-). \end{aligned} \tag{A· 5}$$

The number of non-breaks can be expressed by sum of transition to home and away. Therefore, the number of breaks can be expressed by subtracting it from the total number of slot intervals. Accordingly, we have TF for BMP:

$$\begin{aligned} \text{TF: minimize} \quad & 4n^2 - 4n \\ & - \sum_{t \in T} \sum_{s \in S^-} (to\_home_{t,s} + to\_away_{t,s}) \\ \text{subject to} \quad & (\text{A· 1}), (\text{A· 2}), (\text{A· 3}), (\text{A· 4}), (\text{A· 5}). \end{aligned}$$

Appendix B: Comparision with QUBO formulation and QUBO solver

The BMP was formulated as quadratic unconstrained binary optimization problem (QUBO) [17, 26] and hence can be solved by general mixed-integer nonlinear (MINLP) solvers. SCIP is one of the solvers which has a capability to solve MINLP, implementing a spatial branch-and-bound algorithm based on linear outer approximation. See [27] for the details. Several solvers specifically designed for QUBO have also been developed. QuBowl is one of the state-of-the-art QUBO solvers developed by Rehfeldt, Koch, and Shinano [28]. It converts QUBO problems to MaxCut problems and solve them using a branch-and-cut method.

We compared our proposed bigram based formulation with QUBO formulation. For QUBO formulation, we solved it with two solvers, SCIP 8.0.3 and QuBowl. For this experiment, We set the time limit to 7,200 seconds.

Table A· 1 shows the result. In this table, we compare the bigram based formulation and QUBO formulation solved by SCIP and QuBowl, namely "BBF," "QUBO," "QuBowl."

The first column indicates the size of the instances. The second, third, and fourth columns, labeled "#solved," show the number of problems successfully solved out of the total attempted problems for each formulation. The fifth, sixth, and seventh columns, under the header "#nodes," display the average number of nodes explored in the branch-and-bound algorithm. The eighth, ninth, and tenth columns, under the header "run time [s]," indicate the average computational time (seconds) required to solve the problems. If the solving process reaches the time limit, the reported number of nodes and computational time represent the values at the end of the interrupted solving. The eleventh, twelfth, and thirteenth columns, under the header "gap(%)," indicate the average gap at the end of solving.

As seen in the result, bigram based formulation is less effective than QUBO formulation nor applying QUBO solver. However, this formulation needs MINLP solver or QUBO solver. Thus, our bigram based formulation should be attractive for those who use ILP solver.

Koichi Fujii received his M.S. degree in Mathematics from Tokyo University in 2007. He is currently a PhD student in the Department of Industrial Engineering and Economics at the Tokyo Institute of Technology. His research interests include integer programming and computational mathematical programming.

Tomomi Matsui received his B.E., M.E., and Dr. of Science degree from Tokyo Institute of Technology. He is now a professor at Tokyo Institute of Technology. His research interests include combinatorial optimization and graph theory.

Table A· 1 Comparison with QUBO formulation and QuBowl

$2n$	#solved			#nodes			run time [s]			gap (%)		
	BBF	QUBO	QuBowl	BBF	QUBO	QuBowl	BBF	QUBO	QuBowl	BBF	QUBO	QuBowl
4	5/5	5/5	5/5	0	1	1	0.00	0.01	0.02	0.00	0.00	0.00
6	5/5	5/5	5/5	1	1	1	0.01	0.01	0.01	0.00	0.00	0.00
8	5/5	5/5	5/5	1	1	1	0.03	0.17	0.01	0.00	0.00	0.00
10	5/5	5/5	5/5	1	1	1	0.11	0.13	0.01	0.00	0.00	0.00
12	5/5	5/5	5/5	2	5	1	0.28	0.40	0.02	0.00	0.00	0.00
14	5/5	5/5	5/5	9	9	1	0.53	0.56	0.04	0.00	0.00	0.00
16	6/6	6/6	6/6	15	31	1	1.89	1.32	0.06	0.00	0.00	0.00
18	6/6	6/6	6/6	325	255	4	6.79	4.29	0.17	0.00	0.00	0.00
20	5/5	5/5	5/5	634	678	2	15.74	11.80	0.26	0.00	0.00	0.00
22	5/5	5/5	5/5	810	1,097	3	32.58	23.81	0.42	0.00	0.00	0.00
24	5/5	5/5	5/5	1,594	2,499	4	80.35	67.41	0.77	0.00	0.00	0.00
26	5/5	5/5	5/5	3,215	5,862	22	268.67	236.52	1.69	0.00	0.00	0.00
28	5/5	5/5	5/5	6,588	18,471	208	888.93	923.25	15.26	0.00	0.00	0.00
30	5/5	5/5	5/5	13,120	35,927	186	2,682.59	2,097.16	18.18	0.00	0.00	0.00
32	2/5	3/5	5/5	*23,971	*71,363	415	5,353.68	4,923.53	46.77	3.09	1.04	0.00
34	0/5	0/5	5/5	*24,831	*75,440	3,595	7,200.00	7,200.00	509.36	9.36	5.98	0.00
36	0/5	0/5	5/5	*20,602	*52,718	6,191	7,200.00	7,200.00	1,156.00	9.88	7.17	0.00
38	0/5	0/5	4/5	*13,298	*37,551	*8,986	7,200.00	7,200.00	2,038.93	19.09	11.00	0.53
40	0/5	0/5	2/5	*7,126	*28,877	*18,027	7,200.00	7,200.00	5,114.12	19.95	14.64	2.17
42	0/5	0/5	1/5	*7,524	*18,405	*16,252	7,200.00	7,200.00	5,924.13	30.06	19.45	3.27
44	0/5	0/5	0/5	*5,867	*16,683	*15,189	7,200.00	7,200.00	7,200.00	53.67	26.00	4.75
46	0/5	0/5	0/5	*6,652	*13,999	*12,679	7,200.00	7,200.00	7,200.00	51.38	24.92	3.31
48	0/5	0/5	0/5	*4,230	*11,422	*10,380	7,200.00	7,200.00	7,200.00	52.29	25.21	3.37
50	0/5	0/5	0/5	*4,145	*9,116	*7,890	7,200.00	7,200.00	7,200.00	69.28	33.43	6.47

* Underestimated as some instances could not be solved within the time limit.