

Centralized Round-Robin Tournament Scheduling with a Flexible Number of Matches per Round

Hikaru Anai

Shizuoka University

Yunjian Cao

Seikei University

Wei Wu

`goi@shizuoka.ac.jp`

Shizuoka University

Research Article

Keywords: Sports scheduling, round-robin tournament, venue assignment, edge coloring, mixed-integer programming, iterated local search

Posted Date: October 15th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10470160/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Centralized Round-Robin Tournament Scheduling with a Flexible Number of Matches per Round

Hikaru Anai^{1†}, Yunjian Cao^{2†}, Wei Wu^{1*}

^{1*}Graduate School of Integrated Science and Technology, Shizuoka University, 3-5-1 Johoku Chuo-ku, Hamamatsu, 432-8561, Shizuoka, Japan.

² Faculty of Science and Engineering, Seikei University, 3-3-1 Kichijoji-Kitamachi, Musashino, 180-8633, Tokyo, Japan.

*Corresponding author(s). E-mail(s): goi@shizuoka.ac.jp;

Contributing authors: hikaru.anai.22@shizuoka.ac.jp;

yunjian-cao@st.seikei.ac.jp;

[†]These authors contributed equally to this work.

Abstract

We study a centralized single round-robin tournament scheduling problem in which the number of matches per round is flexible. A common venue is selected for each round, and each team plays at most once per round. The goal is to minimize the total travel distance. We prove that the decision version is NP-complete in the strong sense. We also propose mixed-integer programming (MIP) models with symmetry-breaking constraints and identify polynomially solvable cases, including an exact optimality test for the two-venue case based on edge coloring of threshold graphs. These structural results characterize conditions under which an independent-match lower bound is attainable and provide polynomial-time constructions for several parameter regimes. For larger instances, we propose an iterated local search (ILS) heuristic featuring constructive edge coloring, merge-and-split, ejection chain, and eject-and-inject neighborhoods. Computational experiments on 64 benchmark instances demonstrate that the ILS heuristic matches proven optima on small instances and substantially outperforms the MIP approach on larger ones.

Keywords: Sports scheduling, round-robin tournament, venue assignment, edge coloring, mixed-integer programming, iterated local search

1 Introduction

Round-robin tournament scheduling has been studied extensively in operations research, with applications ranging from professional baseball and basketball to football leagues [1, 2]. A common assumption in this literature is that the round structure is fixed: Each round consists of a maximum matching of teams, and the number of matches per round is therefore determined by the tournament format. Under this assumption, the main decisions concern the ordering of rounds [3, 4], home-away patterns [3], venue assignments [5], travel minimization [1, 6], break minimization [3], carry-over effects [7], and other fairness or operational requirements [2, 4]. The traveling tournament problem is a prominent example in which travel and home-away sequences are optimized jointly for professional leagues [6, 8].

In this paper, we consider a different scheduling environment motivated by regional amateur sports and school club leagues in Japan. In these settings, all matches in a round are often held at a single venue, possibly with multiple grounds, and matches may start at different times on the same day. This single-venue format simplifies facility reservation and match-day management, including communication with teams, schedule control, and responses to unexpected changes. It can also create an event-like atmosphere in which players and parents from participating teams interact beyond their own matches, and it provides young players with opportunities to watch other matches at the same venue.

These practical features motivate a centralized single round-robin tournament in which a common venue is selected for each round, every pair of teams plays exactly once, and each team plays at most once per round. Unlike many existing round-robin and centralized tournament scheduling studies, in which each round corresponds to a maximum matching [4, 9], we allow a round to contain any number of non-overlapping matches. To the best of our knowledge, the number of matches assigned to a round has received limited attention as an explicit decision variable in the round-robin tournament scheduling literature.

This flexibility reflects situations in amateur and school sports leagues where organizers may prefer to hold only a subset of matches on a given day because of travel burden or other operational constraints. Consequently, the number of matches assigned to each round becomes a decision variable rather than a fixed consequence of the tournament format. We call this problem the *centralized round-robin tournament scheduling problem with flexible matches per round* (CRRTSP-FMR), and our objective is to minimize the total travel distance from teams to the selected venues.

Allowing the number of matches per round to vary is not merely a cosmetic extension of the classical round-robin model. By splitting rounds, the scheduler can reduce the set of teams traveling on a given date and choose a venue close to those teams, creating a trade-off between preserving compact round-robin pairings and improving venue proximity. At the same time, amateur and school leagues must usually complete all matches within a limited season, so the number of available match days is bounded. To reflect this operational requirement, we impose an upper bound on the number of rounds. The resulting problem combines matching-type feasibility constraints with partitioning decisions over rounds and venue-selection decisions, which makes a direct exact approach computationally demanding as the number of teams grows.

Our problem is related to, but distinct from, several neighboring scheduling settings. In single-court or single-venue tournament scheduling, matches are typically sequenced at one court or venue, often with objectives such as minimizing waiting times or constructing feasible timetables [9]. Centralized tournament scheduling and venue-assignment models also consider shared venues or venue-related decisions, but they usually retain the classical round-robin structure in which each round is a maximum matching [4, 5]. Sports festival scheduling, more broadly, concerns the selection and timing of activities, shows, or events across days and stages, with objectives related to logistics, cost, and audience experience [10]. In contrast, CRRTSP-FMR keeps the pairwise round-robin requirement among teams while making the number of matches in each round flexible and jointly deciding both the partition of matches into rounds and the common venue assigned to each round.

The contributions of this paper are fourfold. First, we develop two mixed-integer programming formulations: a linearized time-indexed formulation and an assignment-based formulation with valid symmetry-breaking constraints. Second, we establish the computational complexity of the problem by proving that its decision version is NP-complete in the strong sense even when all travel costs are restricted to $\{0, 1, 3\}$. Third, we identify two polynomially solvable regimes for the round limit and develop a polynomial-time optimality test for the two-venue case based on edge coloring of threshold graphs. Fourth, we design an iterated local search heuristic tailored to the problem structure and conduct computational experiments on instances with different numbers of teams, candidate venues, and rounds.

The remainder of this paper is organized as follows. Section 2 defines CRRTSP-FMR. Section 3 presents the mathematical formulations. Section 4 proves that the decision version of CRRTSP-FMR is NP-complete in the strong sense and identifies polynomially solvable cases. Section 5 describes the proposed heuristic. Section 6 reports the computational results. Section 7 concludes the paper and outlines directions for future research.

2 Problem Description

Let $\rho(n)$ denote the minimum number of rounds required for a single round-robin tournament with n teams:

$$\rho(n) = \begin{cases} n - 1, & \text{if } n \text{ is even,} \\ n, & \text{if } n \text{ is odd.} \end{cases}$$

We are given a set of teams $N = \{1, 2, \dots, n\}$, a set of candidate venues $M = \{1, 2, \dots, m\}$, and an upper bound q on the number of rounds, where $q \geq \rho(n)$. Thus, a feasible solution may use any number of rounds up to q . Each round represents a date or time period in which any number of non-overlapping matches may be scheduled. Let $c_{ij} \geq 0$ denote the travel distance from team i to venue j .

The *centralized round-robin tournament scheduling problem with flexible matches per round* (CRRTSP-FMR) is to assign every match of a single round-robin tournament to a round and to select a venue for each active round so as to minimize the total travel distance, subject to the following constraints:

- all matches played in the same round use a common venue;
- every pair of teams plays exactly once;
- each team plays at most one match in each round.

Because the objective depends only on the composition of rounds and their selected venues, not on the chronological order of the rounds, a feasible solution can be viewed as a set of active rounds. This observation is useful both for the mathematical formulation and for the design of the heuristic described later.

3 Mathematical Formulations

We first introduce a time-indexed formulation in which each candidate round is explicitly indexed. Let $Q = \{1, 2, \dots, q\}$ denote the set of candidate rounds, and let $P = \{(i, i') \mid i, i' \in N, i < i'\}$ denote the set of matches. Let $x_{ii'k}$ be a binary variable that takes value 1 if teams i and i' play in round k , and 0 otherwise. Let y_{jk} be a binary variable that takes value 1 if venue j is used in round k , and 0 otherwise. CRRTSP-FMR can be formulated as follows:

$$\min \quad \sum_{j \in M} \sum_{(i, i') \in P} \sum_{k \in Q} (c_{ij} + c_{i'j}) x_{ii'k} y_{jk} \quad (1)$$

$$\text{s.t.} \quad \sum_{k \in Q} x_{ii'k} = 1 \quad \forall (i, i') \in P \quad (2)$$

$$\sum_{j \in M} y_{jk} \leq 1 \quad \forall k \in Q \quad (3)$$

$$\sum_{i': (i', i) \in P} x_{i'ik} + \sum_{i': (i, i') \in P} x_{ii'k} \leq \sum_{j \in M} y_{jk} \quad \forall k \in Q, \forall i \in N \quad (4)$$

$$x_{ii'k} \in \{0, 1\} \quad \forall (i, i') \in P, \forall k \in Q \quad (5)$$

$$y_{jk} \in \{0, 1\} \quad \forall j \in M, \forall k \in Q. \quad (6)$$

The objective function (1) minimizes the total travel distance. Constraint (2) ensures that every pair of teams plays exactly once. Constraint (3) enforces the use of at most one common venue per round. Constraint (4) ensures that each team participates in at most one match in a round and prevents matches from being assigned to inactive rounds.

Although constraints (2)–(6) are linear, the objective function (1) is nonlinear because it contains products of binary variables. To linearize the model, we introduce a binary variable $z_{ii'j}$ that takes value 1 if match (i, i') is assigned to venue j , and 0

otherwise. Using $z_{ii'j}$, we obtain the following linearized formulation:

$$\min \quad \sum_{j \in M} \sum_{(i, i') \in P} (c_{ij} + c_{i'j}) z_{ii'j} \quad (7)$$

$$\text{s.t.} \quad (2)-(6) \quad (8)$$

$$z_{ii'j} + 1 \geq x_{ii'k} + y_{jk} \quad \forall (i, i') \in P, \forall j \in M, \forall k \in Q \quad (9)$$

$$z_{ii'j} \in \{0, 1\} \quad \forall (i, i') \in P, \forall j \in M. \quad (10)$$

Constraint (9) sets $z_{ii'j}$ to 1 whenever match (i, i') is played in round k and round k uses venue j . Because all travel costs are nonnegative, any other $z_{ii'j}$ can be set to 0 without increasing the objective value. We refer to the resulting mixed-integer programming (MIP) model as the time-indexed formulation.

Because the objective does not depend on the chronological order of rounds, any feasible schedule can be represented by several equivalent solutions obtained by permuting the active round indices. Thus, the time-indexed formulation contains substantial symmetry, which may unnecessarily enlarge the branch-and-bound search tree.

To reduce this symmetry, we impose a canonical ordering on the rounds. Assign a unique index $p \in \{1, \dots, |P|\}$ to each match $(i, i') \in P$ according to lexicographic order, and let x'_{pk} denote the variable corresponding to $x_{ii'k}$ for the p -th match. We add the following symmetry-breaking constraints to the time-indexed formulation:

$$x'_{1,1} = 1 \quad (11)$$

$$x'_{pk} \leq \sum_{p'=1}^{p-1} x'_{p',k-1} \quad \forall k \in Q \setminus \{1\}, \forall p \in \{k, \dots, |P|\} \quad (12)$$

$$x'_{pk} = 0 \quad \forall k \in Q \setminus \{1\}, \forall p \in \{1, \dots, k-1\} \quad (13)$$

$$\sum_{j \in M} y_{jk} \geq \sum_{j \in M} y_{j,k+1} \quad \forall k \in Q \setminus \{q\}. \quad (14)$$

Constraint (11) fixes one representative by assigning the first match to the first round. Constraints (12) and (13) impose an ordering on the nonempty rounds based on their smallest-indexed matches. To clarify the index ranges, let r_k be the smallest match index in the k -th nonempty round when these rounds are ordered so that $r_1 < r_2 < \dots$. Because match 1 belongs to the first round, $r_1 = 1$, and strict monotonicity implies $r_k \geq k$. Hence, no match with index $p < k$ can belong to round k , which explains the range in (13). Moreover, if match p belongs to round $k \geq 2$, then the preceding round contains match $r_{k-1} < r_k \leq p$, which explains (12) and its range $p \geq k$. Conversely, applying (12) to the smallest-indexed match in each nonempty round forces these minima to increase strictly. Constraint (14) places the active rounds consecutively at the beginning of the planning horizon.

Thus, every feasible physical schedule has a representation satisfying (11)–(14), obtained by sorting its nonempty rounds by their smallest match indices. The constraints eliminate permutations among the nonempty rounds, although degenerate solutions containing additional empty active rounds may remain.

We also consider an alternative assignment-based formulation. In this formulation, the matches are regarded as jobs, and the machines are venue-use pairs $(j, k) \in M \times Q$, where k represents the k -th possible use of venue j . Let $\beta_{ii'jk}$ be a binary variable that takes value 1 if match (i, i') is assigned to venue-use pair (j, k) , and 0 otherwise. Let y'_{jk} be a binary variable that takes value 1 if the k -th use of venue j is active, and 0 otherwise. The assignment-based formulation is given by

$$\min \quad \sum_{j \in M} \sum_{k \in Q} \sum_{(i, i') \in P} (c_{ij} + c_{i'j}) \beta_{ii'jk} \quad (15)$$

$$\text{s.t.} \quad \sum_{j \in M} \sum_{k \in Q} \beta_{ii'jk} = 1 \quad \forall (i, i') \in P \quad (16)$$

$$\sum_{j \in M} \sum_{k \in Q} y'_{jk} \leq q \quad (17)$$

$$\sum_{i': (i', i) \in P} \beta_{i'ijk} + \sum_{i': (i, i') \in P} \beta_{ii'jk} \leq y'_{jk} \quad \forall i \in N, \forall j \in M, \forall k \in Q \quad (18)$$

$$\beta_{ii'jk} \in \{0, 1\} \quad \forall (i, i') \in P, \forall j \in M, \forall k \in Q \quad (19)$$

$$y'_{jk} \in \{0, 1\} \quad \forall j \in M, \forall k \in Q. \quad (20)$$

The objective function (15) minimizes the total travel distance. Constraint (16) assigns every match to exactly one venue-use pair. Constraint (17) limits the number of active venue-use pairs, and hence the number of rounds, to at most q . Constraint (18) has two roles: It activates a venue-use pair whenever a match is assigned to it, and it prevents two matches involving the same team from being assigned to the same venue-use pair. Thus, each active pair (j, k) corresponds to one round held at venue j . This formulation is linear and directly captures the interpretation of the problem as an assignment of match jobs to venue-use machines.

The assignment-based formulation also contains symmetry. For a fixed venue j , the indices $k \in Q$ represent indistinguishable uses of the same venue; therefore, permuting the use indices of venue j yields an equivalent formulation. To reduce this symmetry, we impose a canonical ordering on the uses of each venue. As above, assign a unique index $p \in \{1, \dots, |P|\}$ to each match $(i, i') \in P$ according to lexicographic order, and let β'_{pjk} denote the variable corresponding to $\beta_{ii'jk}$ for the p -th match. We add the following constraints:

$$y'_{jk} \geq y'_{j,k+1} \quad \forall j \in M, \forall k \in Q \setminus \{q\} \quad (21)$$

$$\beta'_{1,j,k+1} = 0 \quad \forall j \in M, \forall k \in Q \setminus \{q\} \quad (22)$$

$$\beta'_{p,j,k+1} \leq \sum_{p'=1}^{p-1} \beta'_{p',j,k} \quad \forall j \in M, \forall k \in Q \setminus \{q\}, \forall p \in \{2, \dots, |P|\}. \quad (23)$$

Constraint (21) requires the active uses of each venue to appear consecutively. Constraints (22) and (23) order the nonempty uses of each venue by the smallest-indexed match assigned to them: if a match is assigned to the $(k + 1)$ -st use of venue j , then the k -th use of the same venue must contain a smaller-indexed match. The soundness of this symmetry-breaking approach is formalized in the following theorem.

Theorem 1 *The symmetry-breaking constraints (21)–(23) in the assignment-based formulation preserve at least one representation of every feasible physical schedule and eliminate permutations among the nonempty venue-use indices for each venue.*

Proof Let S be any valid physical schedule, defined purely by the partitioning of matches into rounds and the venue assigned to each round (ignoring chronological order). For any venue j , let there be k_j nonempty rounds held at venue j . Represent these rounds by the first k_j venue-use indices, set the corresponding y'_{jk} variables to 1, and set all remaining y'_{jk} variables to 0. This assignment satisfies Constraint (21). Constraints (22) and (23) mandate that the smallest-indexed match assigned to use k must be smaller than the smallest-indexed match assigned to use $k + 1$. Because the match subsets assigned to different uses are disjoint and non-empty, their minimum elements are strictly distinct. Thus, there is exactly one way to sort the k_j active uses such that their minimum match indices strictly increase. Consequently, the k_j nonempty uses have a unique ordering by their minimum match indices. The constraints therefore eliminate the $\prod_{j \in M} k_j!$ equivalent labelings of the nonempty uses without removing every representation of any feasible physical schedule. Degenerate solutions with additional empty active uses may remain, but they do not represent different physical schedules. $\square$

In practice, this assignment-based formulation can be computationally advantageous relative to the time-indexed formulation. First, Theorem 1 shows that it removes a large part of the symmetry associated with repeated uses of the same venue. Second, the conflict constraint (18) gives the formulation a packing structure that may strengthen the linear programming (LP) relaxation and improve lower bounds. Finally, directly coupling the cost coefficients to the assignment variables in the objective (15) provides useful guidance for the solver’s internal heuristics when searching for incumbent solutions.

4 Computational Complexity and Polynomially Solvable Cases

This section first establishes the computational complexity of the general CRRTSP-FMR, and then identifies structural properties that lead to polynomial-time optimality tests in special cases. We consider two extreme regimes for the round limit q , and develop a sharper lower-bound test for the two-venue case.

4.1 Strong NP-completeness of the general problem

The decision version of CRRTSP-FMR asks whether there exists a feasible schedule using at most q rounds whose total travel cost is at most a given threshold K .

Theorem 2 *The decision version of CRR-TSP-FMR is NP-complete in the strong sense, even if the travel costs c_{ij} are restricted to $\{0, 1, 3\}$.*

Proof The problem belongs to NP because a schedule, together with its venue assignments, can be checked in polynomial time. We prove NP-hardness by a reduction from deciding whether a cubic graph is 3-edge-colorable, which is NP-complete [11].

Let G be a cubic graph with vertex set V , edge set $E(G)$, and n vertices. Then n is even, $n \geq 4$, and $|E(G)| = 3n/2$. Let $\bar{G}$ denote the complement of G , so that $E(\bar{G})$ is the set of non-edges of G relative to the complete graph on V . Thus,

$$|E(\bar{G})| = \frac{n(n-1)}{2} - \frac{3n}{2} = \frac{n(n-4)}{2}.$$

We construct an instance of CRR-TSP-FMR as follows.

- **Teams:** Set $N = V$.
- **Venues:** Introduce one base venue $\tilde{v}$ and, for each non-edge $e \in E(\bar{G})$, one dedicated venue v_e . Thus, the number of venues is $1 + |E(\bar{G})| = O(n^2)$.
- **Costs:** Define the travel costs as follows:
 - Set $c_{i\tilde{v}} = 1$ for every team $i \in N$, so every match costs 2 at the base venue.
 - For each non-edge $e = \{u, w\} \in E(\bar{G})$, set $c_{uv_e} = c_{wv_e} = 0$ and $c_{iv_e} = 3$ for every $i \notin \{u, w\}$. Thus, match e costs 0 at v_e , whereas any other match costs at least 3 there.
- **Round limit:** Set $q = 3 + |E(\bar{G})|$.
- **Target cost:** Set $K = 2|E(G)| = 3n$.

The round limit is valid. Because n is even, $\rho(n) = n - 1$, and

$$q - (n - 1) = \frac{(n-2)(n-4)}{2} \geq 0.$$

Moreover,

$$|P| - q = \frac{3(n-2)}{2} \geq 0,$$

so $\rho(n) \leq q \leq |P|$. The construction contains $O(n^3)$ cost entries and is therefore polynomial.

We now prove that G is 3-edge-colorable if and only if the constructed instance admits a schedule of cost at most K using at most q rounds.

($\Rightarrow$) Suppose that G is 3-edge-colorable. Because G is cubic, each of its three color classes is a perfect matching; denote them by M_1, M_2, M_3 . Schedule each M_k in one round at the base venue $\tilde{v}$. These matches contribute $2|E(G)| = 3n$. For every non-edge $e \in E(\bar{G})$, schedule e in a separate round at its dedicated venue v_e , at cost 0. The schedule uses $3 + |E(\bar{G})| = q$ rounds and has total cost K .

($\Leftarrow$) Conversely, suppose that the constructed instance has a schedule of cost at most K . Every edge of G has minimum possible cost 2, attained at the base venue, because it differs from every non-edge defining a dedicated venue and therefore costs at least 3 at each such venue. Every non-edge $e \in E(\bar{G})$ has minimum possible cost 0, attained uniquely at v_e . Hence the sum of the independent minimum match costs is

$$2|E(G)| + 0|E(\bar{G})| = 3n = K.$$

Therefore, a schedule of cost at most K must attain the minimum cost of every match. All edges of G must be played at the base venue, whereas each non-edge e must be played at its dedicated venue v_e , and no other match can be played at v_e .

Because a round uses only one venue, the non-edges require exactly $|E(\overline{G})|$ distinct rounds. Thus, at most $q - |E(\overline{G})| = 3$ rounds remain for the $3n/2$ edges of G . Each round is a matching and can contain at most $n/2$ edges, so at least three rounds are necessary. Consequently, exactly three rounds are used, each containing a perfect matching of G . Assigning one color to each of these rounds implies a proper 3-edge coloring of G .

The reduction uses only the constant travel costs 0, 1, and 3, whereas all other numerical values are polynomially bounded in n . Thus, the decision version of CRR-TSP-FMR is NP-complete in the strong sense. $\square$

4.2 Polynomially solvable cases and lower-bound attainability

We first consider the minimum feasible number of rounds. Recall that $\rho(n)$ equals the chromatic index of the complete graph K_n . When n is even, K_n has a 1-factorization into $n - 1$ perfect matchings; when n is odd, it has a near-1-factorization into n matchings, each of size $(n - 1)/2$.

Theorem 3 *If $q = \rho(n)$, that is, if $q = n - 1$ for even n and $q = n$ for odd n , then CRR-TSP-FMR can be solved in polynomial time.*

Proof Because a round can contain at most $\lfloor n/2 \rfloor$ matches and the tournament contains $n(n - 1)/2$ matches, any feasible schedule with $q = \rho(n)$ must use exactly $\rho(n)$ nonempty rounds, each of maximum possible size. Thus, if n is even, every round is a perfect matching and every team plays in every round. If n is odd, every round contains $(n - 1)/2$ matches, exactly one team has a bye in each round, and each team has a bye exactly once over the n rounds.

If n is even, let q_j denote the number of rounds assigned to venue j . Because every team plays in every round, the total travel distance is

$$\sum_{j \in M} q_j \sum_{i \in N} c_{ij},$$

subject to $\sum_{j \in M} q_j = n - 1$. Therefore, an optimal solution assigns all rounds of any 1-factorization of K_n to a venue j^* satisfying

$$j^* \in \arg \min_{j \in M} \sum_{i \in N} c_{ij}.$$

If n is odd, let $C_j = \sum_{i \in N} c_{ij}$ denote the total travel cost of all teams to venue j . The travel cost of a near-perfect round held at venue j in which team i has a bye is $C_j - c_{ij}$. Because each team has exactly one bye, the venue for the round in which team i rests can be selected independently by choosing

$$j_i^* \in \arg \min_{j \in M} \{C_j - c_{ij}\}.$$

Combining these venue choices with any near-1-factorization of K_n yields a globally optimal schedule. The required venue choices and the corresponding 1-factorization or near-1-factorization can be computed in polynomial time. $\square$

The factorization in Theorem 3 can be generated explicitly by the standard Circle Method. For even n , fix one team and arrange the remaining $n - 1$ teams cyclically. In each round, pair the fixed team with one cyclic position and pair the remaining teams symmetrically across the circle; then rotate the $n - 1$ non-fixed teams. Repeating this procedure for $n - 1$ rotations produces a 1-factorization of K_n . For odd n , add one dummy team, apply the same construction to K_{n+1} , and delete every match involving the dummy team. The team paired with the dummy team receives a bye, yielding a near-1-factorization of K_n into n rounds.

We next examine when every match can be played at one of its cheapest venues. Assign each match to a cheapest venue using any fixed tie-breaking rule, and let $G_j = (N, E_j)$ be the graph of matches assigned to venue j , with $\Delta(G_j)$ denoting its maximum degree. This assignment attains the independent-match lower bound

$$Z^{\text{LB}} := \sum_{(i,i') \in P} \min_{j \in M} \{c_{ij} + c_{i'j}\}. \quad (24)$$

Let $\chi'(G_j)$ denote the chromatic index of G_j , that is, the minimum number of colors required for a proper edge coloring of G_j . Because each feasible round at venue j is a matching, partitioning E_j into the minimum number of rounds is equivalent to finding an optimal edge coloring of G_j . Consequently, exactly $\chi'(G_j)$ rounds are required.

Theorem 4 *If*

$$q \geq \sum_{j \in M} \chi'(G_j),$$

then the independent-match lower bound Z^{LB} is attainable and is therefore the optimal objective value of CRRTSP-FMR.

Proof For each venue j , take an optimal edge coloring of G_j with $\chi'(G_j)$ colors. Each color class is a matching and therefore defines a feasible round at venue j . Combining these color classes over all venues produces a schedule with at most $\sum_{j \in M} \chi'(G_j) \leq q$ rounds. Because every match is played at a cheapest venue, the resulting schedule has cost Z^{LB} and is optimal. $\square$

For general graphs, determining $\chi'(G_j)$ is NP-hard. Nevertheless, a polynomial-time computable upper bound yields the following sufficient condition.

Corollary 5 *For each venue j , define*

$$\kappa_j := \begin{cases} 0, & \text{if } E_j = \emptyset, \\ \min\{\rho(n), \Delta(G_j) + 1\}, & \text{otherwise.} \end{cases}$$

If $q \geq \sum_{j \in M} \kappa_j$, then an optimal solution of CRRTSP-FMR can be constructed in polynomial time.

Proof Restricting a $\rho(n)$ -edge-coloring of K_n to E_j yields an edge coloring of G_j with at most $\rho(n)$ colors. Alternatively, Vizing’s theorem [12] guarantees an edge coloring of G_j with at most $\Delta(G_j) + 1$ colors, and such a coloring can be constructed in polynomial time using the Misra–Gries algorithm [13]. Choosing the construction that uses fewer colors partitions E_j into at most κ_j matchings. Combining these matchings over all venues produces a schedule with at most $\sum_{j \in M} \kappa_j \leq q$ rounds and objective value Z^{LB} . Its optimality follows from Theorem 4. $\square$

Because $\kappa_j \leq \rho(n)$ for every venue, the simpler condition $q \geq m\rho(n)$ is an immediate consequence of Corollary 5.

We also record a simple observation that will be used both in the exact test below and in the heuristic construction. It shows that, once the round limit is at least the theoretical minimum $\rho(n)$, requiring exactly q nonempty rounds causes no loss of optimality as long as q does not exceed the number of matches.

Lemma 6 *If $\rho(n) \leq q \leq |P|$, then there exists an optimal solution that uses exactly q nonempty rounds.*

Proof Because $q \geq \rho(n)$, at least one feasible schedule exists. Consider an optimal solution that uses fewer than q nonempty rounds. Moreover, because $q \leq |P|$, the number of nonempty rounds in this solution is also less than $|P|$. Therefore, at least one round contains two or more matches. Choose such a round R with selected venue j , and partition it into two nonempty subsets R_1 and R_2 . Assign both new rounds to the same venue j . Because each new round contains only a subset of the matches originally assigned to R , the constraint that each team plays at most once in a round remains satisfied. The total travel distance is unchanged, because every match is still played at the same venue as before. Thus, we obtain another optimal solution with one additional nonempty round. Repeating this splitting operation yields an optimal solution with exactly q nonempty rounds. $\square$

Theorem 3 and Corollary 5 provide polynomial-time solution methods for arbitrary numbers of venues. Theorem 4 gives the underlying chromatic-index condition, but testing it is difficult for general cheapest-venue graphs. The two-venue case considered next has additional structure that makes the relevant chromatic indices polynomial-time computable.

4.3 Exact optimality test for the two-venue case

In the special case $m = 2$, the structure of the cheapest-venue partition yields a sharper polynomial-time lower-bound test. The underlying idea remains the same: if the two sets of matches assigned to their individually cheapest venues can be partitioned into at most q matchings in total, then the independent-match lower bound is attainable and therefore optimal. The distinctive feature of the two-venue case is that these two match sets can be represented by a pair of complementary threshold graphs, allowing this condition to be tested in polynomial time.

Reduction to a pair of threshold graphs

Suppose that $m = 2$, with candidate venues indexed by 1 and 2. For each team $i \in N$, define the *venue-preference weight*

$$w_i := c_{i2} - c_{i1}.$$

For a match $(i, i') \in P$, the travel costs at venues 1 and 2 are $c_{i1} + c_{i'1}$ and $c_{i2} + c_{i'2}$, respectively. Venue 1 is no more expensive than venue 2 for this match if and only if

$$(c_{i2} + c_{i'2}) - (c_{i1} + c_{i'1}) = w_i + w_{i'} \geq 0.$$

Definition 1 Define $G_1 = (N, E_1)$ and $G_2 = (N, E_2)$ by

$$E_1 := \{(i, i') \in P : w_i + w_{i'} \geq 0\},$$

$$E_2 := P \setminus E_1 = \{(i, i') \in P : w_i + w_{i'} < 0\}.$$

Thus, G_2 is the complement of G_1 relative to the complete graph $K_n = (N, P)$.

Lemma 7 *Both G_1 and G_2 are threshold graphs.*

Proof The edge set E_1 is generated by the vertex weights $(w_i)_{i \in N}$ and threshold $T = 0$ according to the rule “ $(i, i') \in E_1$ if and only if $w_i + w_{i'} \geq T$.” This is a standard equivalent definition of a threshold graph [14, 15]; hence, G_1 is a threshold graph.

Threshold graphs are precisely the graphs with no induced P_4 , C_4 , or $2K_2$ [15]. Complementation maps P_4 to P_4 , C_4 to $2K_2$, and $2K_2$ to C_4 . The family of forbidden induced subgraphs is therefore closed under complementation. Because $G_2 = \overline{G_1}$, it follows that G_2 is also a threshold graph. $\square$

Every threshold graph is both a *split-interval* graph and a *split-comparability* graph [15, 16]. We use the latter property below.

The chromatic index of a threshold graph in polynomial time

Because a round at a fixed venue is a matching, the minimum number of rounds required to schedule all edges of E_1 at venue 1 is the *chromatic index* $\chi'(G_1)$. Indeed, a proper edge coloring of G_1 is equivalent to a partition of E_1 into matchings. The same argument applies to G_2 and venue 2. By Vizing’s theorem [12], $\chi'(H) \in \{\Delta(H), \Delta(H) + 1\}$ for every simple graph H . Deciding which value applies, known as the *classification problem*, is NP-complete for general graphs [11] and remains open for general split graphs [17]. For threshold graphs, however, the following characterization yields a polynomial-time test.

For a vertex v of a graph H , let

$$\Gamma_H[v] := \{v\} \cup \{u \in V(H) : \{u, v\} \in E(H)\}$$

denote the closed neighborhood of v in H , and let $H[\Gamma_H[v]]$ denote the subgraph of H induced by this vertex set.

Theorem 8 ([18]) *Let H be a threshold graph with maximum degree Δ . Then $\chi'(H) = \Delta + 1$ if and only if H is neighborhood-overfull: there exists a vertex v whose degree in H , denoted by $d_H(v)$, satisfies $d_H(v) = \Delta$ and*

$$|E(H[\Gamma_H[v]])| > \Delta \left\lfloor \frac{\Delta + 1}{2} \right\rfloor.$$

The theorem follows directly from the corresponding characterization for split-comparability graphs, because every threshold graph belongs to this class [18].

To apply this characterization algorithmically, order the vertices of a threshold graph H by nonincreasing normalized threshold weights, $\omega_1 \geq \omega_2 \geq \dots \geq \omega_n$, where two vertices are adjacent if and only if the sum of their weights is nonnegative. The vertices with nonnegative weights form a clique C , whereas those with negative weights form an independent set S . Moreover, the neighborhood in C of every vertex in S is a prefix of C , and these neighborhoods are nested. This ordering permits compact preprocessing of the required degree and induced-edge counts.

For $H = G_1$ (and analogously for G_2), the weight representation makes both $\Delta(H)$ and the neighborhood-overfull test computable in $O(n \log n)$ time:

- Sort the weights $(w_i)_{i \in N}$. For each vertex i , determine

$$d_{G_1}(i) = |\{i' \in N \setminus \{i\} : w_i + w_{i'} \geq 0\}|$$

by binary search. The complete degree sequence and $\Delta_1 := \Delta(G_1)$ are therefore obtained in $O(n \log n)$ time. Because $G_2 = \overline{G_1}$, we have $d_{G_2}(i) = n - 1 - d_{G_1}(i)$, from which $\Delta_2 := \Delta(G_2)$ follows immediately.

- Precompute the number of edges induced by every prefix of the threshold ordering. For any vertex v , its closed neighborhood $\Gamma_{G_1}[v]$ consists of such a prefix, possibly together with v itself. Consequently, $|E(G_1[\Gamma_{G_1}[v]])|$ can be obtained in constant time from the prefix data. Checking all vertices of maximum degree then determines whether G_1 is neighborhood-overfull. The same calculation applies to G_2 after reversing the threshold ordering and using the strict inequality that defines its edge set.

These observations yield the following result.

Theorem 9 *If $m = 2$, then the chromatic indices $\chi'(G_1)$ and $\chi'(G_2)$ of the cheapest-venue graphs in Definition 1 can be determined in $O(n \log n)$ time.*

Proof By Lemma 7, both G_1 and G_2 are threshold graphs. Their degree sequences and maximum degrees can be computed in $O(n \log n)$ time from the sorted weight representation. The required neighborhood edge counts are then obtained from the prefix data, so the neighborhood-overfull condition in Theorem 8 can be checked for all maximum-degree vertices in linear time after sorting. The theorem therefore determines each chromatic index in $O(n \log n)$ time. $\square$

Algorithm 1 Chromatic index of a threshold graph

Require: A threshold graph H with vertex set $V(H)$ and edge set $E(H)$, represented by normalized threshold weights or by a threshold ordering.

- 1: Sort the vertices by nonincreasing threshold weight and precompute the required prefix data.
 - 2: Compute $d_H(i)$ for every $i \in V(H)$ and set $\Delta \leftarrow \max_{i \in V(H)} d_H(i)$.
 - 3: Check the neighborhood-overfull condition in Theorem 8 for every vertex of degree Δ .
 - 4: **if** H is neighborhood-overfull **then**
 - 5: $\chi'(H) \leftarrow \Delta + 1$.
 - 6: **else**
 - 7: $\chi'(H) \leftarrow \Delta$.
 - 8: **end if**
 - 9: **return** $\chi'(H)$.
-

Exact optimality for $m = 2$

Corollary 10 *Suppose $m = 2$. If $\chi'(G_1) + \chi'(G_2) \leq q$, then the lower bound Z^{LB} in (24) is attainable and therefore optimal. Moreover, if $w_i + w_{i'} \neq 0$ for every $(i, i') \in P$, this condition is also necessary for attaining Z^{LB} .*

Proof Sufficiency follows directly from Theorem 4. In the absence of ties, attaining Z^{LB} requires every match in E_1 to be played at venue 1 and every match in E_2 at venue 2. These matches require at least $\chi'(G_1)$ and $\chi'(G_2)$ rounds, respectively. Because each round uses only one venue, any such schedule requires at least $\chi'(G_1) + \chi'(G_2)$ rounds, proving necessity. $\square$

5 Iterated Local Search

In this section, we propose an iterated local search (ILS) heuristic for the CRR-TSP-FMR. The method maintains feasibility throughout the search and exploits the key structural decisions of the problem: how a set of matches should be partitioned into rounds and how a common venue should be assigned to each round.

5.1 Preprocessing and lower-bound test

The polynomial-time results in Section 4 can be used as preprocessing before the heuristic search. If $q = \rho(n)$, Theorem 3 directly constructs an optimal schedule. Otherwise, assign each match to a cheapest venue and compute the values κ_j in Corollary 5. If $q \geq \sum_{j \in M} \kappa_j$, the corollary constructs an optimal schedule attaining Z^{LB} . If this condition fails and $m = 2$, we compute $\chi'(G_1)$ and $\chi'(G_2)$ using Theorem 9 and apply the exact lower-bound test in Corollary 10, as summarized in Algorithm 2. When the test establishes that Z^{LB} is attainable, this value is used as a proven optimal target, and the subsequent ILS can terminate as soon as it finds a schedule with this cost. In all other cases, including instances with $m \geq 3$, the algorithm proceeds to the ILS without a proven optimal target.

By Theorem 9, Algorithm 2 runs in polynomial time. If $\chi'(G_1) + \chi'(G_2) \leq q$, the test certifies that Z^{LB} is attainable and hence optimal. If the inequality fails and there are no ties in the cheapest-venue assignments, Corollary 10 certifies that this lower bound is unattainable. When ties are present, failure of the inequality is inconclusive because alternative assignments of tied matches may yield a different partition.

Algorithm 2 Polynomial-time lower-bound test for CRRTSP-FMR with $m = 2$

Require: Travel costs c_{i1}, c_{i2} for all $i \in N$, and round bound q .

- 1: Compute $w_i \leftarrow c_{i2} - c_{i1}$ for all $i \in N$; form G_1, G_2 as in Definition 1.
 - 2: Run Algorithm 1 on G_1 and G_2 to obtain $\chi'(G_1)$ and $\chi'(G_2)$.
 - 3: **if** $\chi'(G_1) + \chi'(G_2) \leq q$ **then**
 - 4: **return** ATTAINABLE, with proven optimal target Z^{LB} .
 - 5: **else if** $w_i + w_{i'} \neq 0$ for every $(i, i') \in P$ **then**
 - 6: **return** UNATTAINABLE.
 - 7: **else**
 - 8: **return** INCONCLUSIVE.
 - 9: **end if**
-

5.2 Initial solution generation

The quality of the initial solution strongly affects the subsequent local search. A straightforward approach is to construct a random single round-robin schedule and repeatedly split its rounds. However, this approach ignores venue-dependent travel costs during schedule construction and may therefore produce an unnecessarily expensive initial solution. To address this issue, we generate two candidate schedules using different graph-theoretic methods, apply a preliminary local search to both, and retain the better one.

Method 1: 1-factorization and greedy split

This method is motivated by Theorem 3. We first use the Circle Method to construct a valid schedule with exactly $\rho(n)$ rounds: a 1-factorization of K_n when n is even and a near-1-factorization when n is odd. For each round, we perturb each venue-specific round cost by an independent factor drawn uniformly from $[1.0, 1.05]$ and select the venue with the minimum perturbed cost.

If $q > \rho(n)$, we iteratively split existing rounds. In each iteration, we consider every round R containing at least two matches and every venue pair $(v_1, v_2) \in M \times M$, allowing $v_1 = v_2$. For a fixed pair, each match in R is assigned to the less expensive of the two venues. Among all candidates, we apply the split with the largest perturbed reduction, replacing R with two nonempty feasible rounds. This process continues until the schedule contains exactly q rounds.

Method 2: Constructive edge coloring and packing

This method combines spatial assignment and temporal feasibility. It proceeds in three stages: greedy venue assignment, constructive edge coloring, and Kempe-chain packing.

Step 1: Greedy venue assignment. We first assign each match to a venue without considering round feasibility. Each match $e = (i, i') \in P$ is assigned to the venue j with the smallest perturbed cost, obtained by multiplying $c_{ij} + c_{i'j}$ by an independent factor drawn uniformly from $[1.0, 1.05]$. This partitions the edge set of K_n into m disjoint venue-specific subgraphs G_j .

Step 2: Constructive edge coloring. For each venue j , the assigned matches must be partitioned into feasible rounds. We apply the constructive Misra–Gries algorithm [13], which implements Vizing’s $\Delta(G_j) + 1$ upper bound [12]. Each color class is a matching and therefore defines a feasible round at venue j . Let $\mathcal{R}$ denote the resulting set of rounds.

Step 3: Kempe-chain packing. To reduce $|\mathcal{R}|$ to q , we first search exhaustively for pairs of mutually compatible rounds and apply feasible merges in nondecreasing order of travel-cost increase. If $|\mathcal{R}| > q$ still holds, we retain the q largest rounds and eject all matches from the remaining rounds. Each ejected match is inserted into a compatible surviving round whenever possible. If direct insertion fails, an alternating-path exchange based on Kempe chains is used to resolve conflicts while preserving feasibility. If this packing procedure fails to produce a feasible q -round schedule, the Method 2 candidate is discarded and Method 1 provides the initial solution.

Selection of the initial solution

We improve each successfully generated candidate by repeatedly applying the merge-and-split neighborhood described in the next subsection until a local optimum is reached. The candidate with the lower travel distance is then selected as the initial solution for the main ILS. This two-method strategy provides diverse starting solutions without the computational cost of a large multistart scheme.

5.3 Local search

After obtaining the initial solution, we apply a best-improvement local search based on four neighborhood operations: merge-and-split, simple ejection chain, variable-depth ejection chain, and eject-and-inject.

Merge-and-split operation

The merge-and-split neighborhood selects two rounds, merges their match sets, and repartitions the merged set into two nonempty feasible rounds. We construct a conflict graph G_R whose vertices are the merged matches and whose edges join pairs of matches sharing a team. Because each of the two original rounds is a matching, G_R is bipartite, and a feasible two-way repartition corresponds to a proper two-coloring of G_R .

Algorithms 3 and 4 evaluate this neighborhood. Algorithm 3 decomposes G_R into connected components and computes a two-coloring. Within each component, the

two color classes may be assigned to the new rounds in either orientation. For a match set R and venue j , let $d(R, j)$ denote the total travel distance incurred by playing all matches in R at j . Algorithm 4 selects these orientations jointly with the two venue assignments to minimize travel distance. This component-based evaluation avoids explicit enumeration of all feasible repartitions.

The time complexities of Algorithms 3 and 4 are $O(|V(G_R)| + |E(G_R)|)$ and $O(m|V(G_R)| + m^2|\mathcal{C}|)$, respectively, where $\mathcal{C}$ is the set of connected components. Hence, one merge-and-split move can be evaluated in $O(|E(G_R)| + m|V(G_R)| + m^2|\mathcal{C}|)$ time.

The merge-and-split operation can also be viewed through the notion of *Kempe chains* in graph coloring [19]. In the current schedule, the two selected rounds correspond to two colors assigned to the vertices of the conflict graph. A feasible repartition of the merged match set is obtained by interchanging these two colors on one or more connected components of the subgraph induced by the two colors. Each such component is a two-color Kempe chain, or an alternating-path component, and swapping the two colors on it is a *Kempe change*. Thus, optimizing a merge-and-split move is equivalent to selecting the best set of Kempe changes between the two round colors, together with the best pair of venue assignments, to minimize the travel distance.

Algorithm 3 Constrained splitting algorithm (part 1: graph coloring)

Require: Merged round R .

- 1: Build the conflict graph G_R , where $V(G_R) = R$ and $\{u, v\} \in E(G_R)$ if matches u and v share a team.
 - 2: Initialize the set of connected components as $\mathcal{C} \leftarrow \emptyset$.
 - 3: Mark every vertex $v \in V(G_R)$ as uncolored by setting $\gamma_v \leftarrow -1$.
 - 4: **while** there exists an uncolored vertex $\bar{v}$ **do**
 - 5: Initialize a new component $C \leftarrow \emptyset$ and an empty stack.
 - 6: Push the uncolored vertex $\bar{v}$ onto the stack and set $\gamma_{\bar{v}} \leftarrow 0$.
 - 7: **while** the stack is not empty **do**
 - 8: Pop a vertex v and add it to C .
 - 9: **for all** uncolored neighbors v' of v **do**
 - 10: Push v' onto the stack and set $\gamma_{v'} \leftarrow 1 - \gamma_v$.
 - 11: **end for**
 - 12: **end while**
 - 13: Add C to $\mathcal{C}$.
 - 14: **end while**
 - 15: **return** Connected components $\mathcal{C}$ and coloring γ .
-

Simple ejection chain operation

The simple ejection chain operation provides a structured way to resolve conflicts through sequential displacement. It attempts to move a single match m_1 from its current round R_i to a destination round R_j . If this insertion creates exactly one conflict with an existing match $m_2 \in R_j$ (i.e., m_1 and m_2 share exactly one team),

Algorithm 4 Constrained splitting algorithm (part 2: best splitting)

Require: Connected components $\mathcal{C}$ and coloring γ .

```
1: Initialize the best cost as  $\sigma^* \leftarrow +\infty$ .
2: for all  $C \in \mathcal{C}$  do
3:   Define  $R_C^{(0)} \leftarrow \{v \in C \mid \gamma_v = 0\}$  and  $R_C^{(1)} \leftarrow \{v \in C \mid \gamma_v = 1\}$ .
4:   Precompute  $d(R_C^{(0)}, j)$  and  $d(R_C^{(1)}, j)$  for each  $j \in M$ .
5: end for
6: for  $j = 1$  to  $m$  do
7:   for  $j' = j$  to  $m$  do
8:     Initialize  $\sigma \leftarrow 0$ ,  $R^{(1)} \leftarrow \emptyset$ , and  $R^{(2)} \leftarrow \emptyset$ .
9:     for all  $C \in \mathcal{C}$  do
10:      if  $d(R_C^{(0)}, j) + d(R_C^{(1)}, j') < d(R_C^{(1)}, j) + d(R_C^{(0)}, j')$  then
11:         $R^{(1)} \leftarrow R^{(1)} \cup R_C^{(0)}$  and  $R^{(2)} \leftarrow R^{(2)} \cup R_C^{(1)}$ .
12:         $\sigma \leftarrow \sigma + d(R_C^{(0)}, j) + d(R_C^{(1)}, j')$ .
13:      else
14:         $R^{(1)} \leftarrow R^{(1)} \cup R_C^{(1)}$  and  $R^{(2)} \leftarrow R^{(2)} \cup R_C^{(0)}$ .
15:         $\sigma \leftarrow \sigma + d(R_C^{(1)}, j) + d(R_C^{(0)}, j')$ .
16:      end if
17:    end for
18:    if  $R^{(1)} = \emptyset$  or  $R^{(2)} = \emptyset$  then
19:      continue
20:    end if
21:    if  $\sigma < \sigma^*$  then
22:       $\sigma^* \leftarrow \sigma$ ,  $R_1^* \leftarrow R^{(1)}$ , and  $R_2^* \leftarrow R^{(2)}$ .
23:    end if
24:  end for
25: end for
26: return New rounds  $R_1^*$  and  $R_2^*$ .
```

the operation ejects m_2 from R_j . It then attempts to feasibly insert the ejected match m_2 into a third distinct round R_k without causing any further conflicts.

From a graph-theoretic perspective, this operation corresponds to an alternating path of length 2 traversing the independent sets of the conflict graph. A vertex m_1 is shifted from independent set I_i to I_j , displacing exactly one conflicting neighbor $m_2 \in I_j$, which is subsequently relocated to I_k , thereby preserving the proper coloring.

To evaluate this neighborhood, we consider $O(n^2)$ matches and $O(q^2)$ ordered pairs of destination rounds. A direct scan for the unique conflicting match takes $O(n)$ time, yielding a worst-case complexity of $O(q^2 n^3)$, or $O(n^5)$ when $q = O(n)$. In the implementation, bitsets represent the teams appearing in each round, enabling constant-time conflict detection for the tested instances. Consistent with the best-improvement strategy, the best improving chain found in the neighborhood is applied.

Variable-depth ejection chain operation

The variable-depth ejection chain (VEC) operation generalizes the simple ejection chain by allowing longer sequences of displacements. It uses a depth-limited search with a beam-width restriction to explore promising ejection paths while controlling computational effort.

For a seed match m_0 removed from round R_0 , the search evaluates inserting m_0 into any round R . To ensure computational tractability, we only consider insertions that produce at most one conflict. If an insertion creates exactly one conflict m' , m' is ejected, and the search recurses on m' . The search tracks the cumulative change in travel distance and uses a lower-bound estimate of the reassignment cost for the unassigned matches to prune unpromising branches. At each step, the search is restricted to the top B most promising branches (beam width) based on the estimated cost, and the chain terminates either when a feasible insertion with no conflicts is found that reduces the global objective or when the maximum depth D is reached.

Graph-theoretically, VEC explores longer alternating paths in the conflict graph, allowing deeper reorganizations across multiple independent sets. The search retains at most B branches at each of at most D levels, so its search tree contains $O(B^D)$ nodes. Because there are $O(n^2)$ seed matches and each node evaluates $O(q)$ rounds, the overall time complexity is $O(n^2qB^D)$. Based on preliminary experiments, we set the maximum depth to $D = 4$ and the beam width to $B = 2$, providing a practical balance between diversification and computational effort.

Eject-and-inject operation

The eject-and-inject neighborhood explores structural changes by completely removing an existing round and then creating a new one to restore the required number of rounds q . For a chosen round R_i , the *eject* phase attempts to reassign all its matches to the remaining $q - 1$ rounds. For each match in R_i , we greedily select the feasible target round that minimizes the increase in travel distance. If all matches of R_i can be feasibly reassigned, R_i is eliminated.

To maintain exactly q rounds, the *inject* phase then forms a new round. It selects a seed match from one of the existing $q - 1$ rounds and initializes a new round R_{new} containing only this seed match. It then evaluates the benefit of moving other matches from their current rounds into R_{new} under the best possible venue assignment for R_{new} . Matches are greedily pulled into R_{new} as long as the total travel distance strictly decreases.

By evaluating all possible choices of the ejected round R_i and the seed match for the injected round, we identify the eject-and-inject combination that yields the largest reduction in total travel distance.

Algorithm 5 describes the complete local search procedure. It explores the merge-and-split, simple ejection chain, variable-depth ejection chain, and eject-and-inject neighborhoods in a sequential best-improvement manner. The search first exhausts the merge-and-split neighborhood; if no improving move exists, it explores the simple ejection chain, followed by the variable-depth ejection chain, and finally the

eject-and-inject neighborhood. Any improvement triggers a return to the merge-and-split neighborhood. The process continues until all neighborhoods yield no further improvement. For severely constrained or extremely large problem instances under strict time limits, the more computationally intensive neighborhoods, specifically the variable-depth ejection chain and the eject-and-inject operation, can be disabled. In our computational experiments, the default ILS configuration relies primarily on the merge-and-split and simple ejection chain neighborhoods to balance solution quality and computational efficiency.

Algorithm 5 Local search algorithm

Require: Current schedule S .

```

1: repeat
2:   Initialize the best improvement as  $\sigma \leftarrow 0$ .
3:   Evaluate all pairs of rounds  $(R_1, R_2)$  in  $S$  using Algorithms 3 and 4 to find the
     best split. Let  $d_{\text{old}} - d_{\text{new}}$  be its improvement.
4:   if  $d_{\text{old}} - d_{\text{new}} > 0$  then
5:     Apply the best merge-and-split move to  $S$  and update  $\sigma \leftarrow d_{\text{old}} - d_{\text{new}}$ .
6:   else
7:     Evaluate the simple ejection chain operation. Let  $d_{\text{old}} - d_{\text{new}}$  be the improve-
       ment.
8:     if  $d_{\text{old}} - d_{\text{new}} > 0$  then
9:       Apply the best simple ejection chain move to  $S$  and update  $\sigma \leftarrow d_{\text{old}} - d_{\text{new}}$ .
10:    else
11:      Evaluate the variable-depth ejection chain operation. Let  $d_{\text{old}} - d_{\text{new}}$  be the
        improvement.
12:      if  $d_{\text{old}} - d_{\text{new}} > 0$  then
13:        Apply the best variable-depth ejection chain move to  $S$  and update  $\sigma \leftarrow$ 
           $d_{\text{old}} - d_{\text{new}}$ .
14:      else
15:        Evaluate the eject-and-inject operation for all candidate rounds and seed
          matches. Let  $d_{\text{old}} - d_{\text{new}}$  be the improvement of the best eject-and-inject
          move.
16:        if  $d_{\text{old}} - d_{\text{new}} > 0$  then
17:          Apply the best eject-and-inject move to  $S$  and update  $\sigma \leftarrow d_{\text{old}} - d_{\text{new}}$ .
18:        end if
19:      end if
20:    end if
21:  end if
22: until  $\sigma = 0$ 
23: return  $S$ .
```

5.4 Hybrid kick move

To escape local optima and avoid search stagnation, we introduce a hybrid adaptive kick move. Its primary perturbation is a *ruin-and-recreate* (R&R) procedure: randomly select $\lfloor \alpha \rfloor$ rounds, pool their matches, and reconstruct the same number of nonempty rounds by randomized sequential insertion. Because R&R may reach a feasibility dead end when q is close to its theoretical minimum, we make up to $A_{\max}$ attempts, choosing a new set of rounds in each attempt. We use $A_{\max} = 5$ in the experiments. If all attempts fail, the method applies a *generalized Kempe chain shift*.

The perturbation size parameter α is dynamically controlled: It is maintained at a base value of 4. Whenever a local optimum does not improve the incumbent solution, α is incremented by 1 before the kick move is applied (meaning the first kick uses a size of 5). It resets to 4 when a new best schedule is found or when the incremented α exceeds $q/2$.

Given $\lfloor \alpha \rfloor$ rounds, we construct a conflict graph of their matches and identify its connected components. The validity of this approach is established by the following theorem, which shows that applying independent permutations to these connected components preserves feasibility.

Theorem 11 *Let K be a set of active rounds in a feasible schedule S , and let $R_K = \bigcup_{k \in K} R_k$ be the set of matches assigned to these rounds. Let G_K be the conflict graph with vertex set R_K , where two vertices are adjacent if the corresponding matches share a team, and let $\mathcal{C}$ be the set of its connected components. For any collection of permutations $\{\pi_C\}_{C \in \mathcal{C}}$, where $\pi_C : K \rightarrow K$, assigning every match $m \in C$ originally in round $k \in K$ to round $\pi_C(k)$ preserves feasibility.*

Proof To prove feasibility, we must show that in the resulting schedule, no team plays more than once in any round $k' \in K$. Suppose, for the sake of contradiction, that there exists a team t that plays in two distinct matches m_1 and m_2 in the same round k' after the shift. Because m_1 and m_2 share team t , they are adjacent in G_K . By the definition of connected components, m_1 and m_2 must belong to the same component $C \in \mathcal{C}$. Consequently, both matches undergo the same permutation π_C . Let k_1 and k_2 be the original rounds of m_1 and m_2 in S . Because both are assigned to round k' in the new schedule, we have $\pi_C(k_1) = k'$ and $\pi_C(k_2) = k'$. Because π_C is a bijection, this implies $k_1 = k_2$. However, in the original feasible schedule S , no team can play twice in the same round, making it impossible for m_1 and m_2 to have been in the same original round k_1 . This contradiction proves that the reassignment is feasible. $\square$

To implement the generalized Kempe chain shift, we extract the connected components of the conflict graph induced by the matches in the selected rounds. We then sample component-wise permutations and reassign each match according to the permutation of its component. We accept only a reassignment that leaves every selected round nonempty and moves at least one match. Theorem 11 guarantees feasibility of every accepted reassignment. In the degenerate case where the conflict graph has only one component, this operation may amount only to relabeling the selected rounds; otherwise, different component-wise permutations can change the round partition.

Algorithm 6 summarizes the hybrid kick move. The fallback uses the round set selected in the final unsuccessful R&R attempt.

Algorithm 6 Hybrid adaptive kick move

Require: Current schedule S , kick parameter α , maximum R&R retries $A_{\max}$.

```

1:  $success \leftarrow \text{false}$ .
2: for  $attempt = 1$  to  $A_{\max}$  do
3:   Randomly choose  $\lfloor \alpha \rfloor$  rounds from  $S$  and denote their index set by  $K$ .
4:   Let  $R_{\text{merged}} \leftarrow \bigcup_{k \in K} R_k$ .
5:   Prepare empty round sets  $T_1, T_2, \dots, T_{\lfloor \alpha \rfloor}$ .
6:   Randomly shuffle the matches in  $R_{\text{merged}}$ .
7:   Greedily assign each match in  $R_{\text{merged}}$  to the first feasible set  $T_r$ .
8:   if all matches are assigned and every  $T_r$  is nonempty then
9:     Replace the rounds indexed by  $K$  with  $T_1, \dots, T_{\lfloor \alpha \rfloor}$  and update their venues.
10:     $success \leftarrow \text{true}$ .
11:    break
12:   end if
13: end for
14: if not  $success$  then
15:   Build  $G_K$  and find its connected components  $\mathcal{C}$ .
16:   Sample permutations  $\{\pi_C\}_{C \in \mathcal{C}}$  until every selected round remains nonempty
   and at least one match moves.
17:   for all  $C \in \mathcal{C}$  do
18:     for all match  $m \in C$  currently in round  $k \in K$  do
19:       Reassign  $m$  to round  $\pi_C(k)$ .
20:     end for
21:   end for
22:   Update venue assignments for the modified rounds in  $S$ .
23: end if
24: return  $S$ .
```

5.5 Overall framework

The overall algorithm is an iterated local search that combines initial solution generation, local search, and kick moves.

Algorithm 7 summarizes the complete procedure. The search alternates between intensification via best-improvement local search and diversification via adaptive kicks until the time limit is reached or an available proven target value is attained. The best incumbent schedule is returned.

5.6 Computational enhancement

Venue selection is evaluated repeatedly during splitting, local search, and kick moves. To avoid redundant computation, we precompute the cost of playing each match (i, i')

Algorithm 7 Iterated local search (ILS)

Require: Time limit T_{limit} and an optional proven target value Z_{target} .

```
1:  $S^* \leftarrow$  initial solution described in Section 5.2.
2:  $\sigma^* \leftarrow$  total travel distance of  $S^*$ .
3:  $\alpha \leftarrow 4$ .
4:  $S \leftarrow S^*$ .
5: while elapsed time  $< T_{\text{limit}}$  and either no target is available or  $\sigma^* > Z_{\text{target}}$  do
6:   Apply Algorithm 5 to  $S$  and obtain a locally optimal solution  $S'$ .
7:   Let  $\sigma'$  be the total travel distance of  $S'$ .
8:   if  $\sigma' < \sigma^*$  then
9:      $S \leftarrow S'$ ,  $S^* \leftarrow S'$ ,  $\sigma^* \leftarrow \sigma'$ , and  $\alpha \leftarrow 4$ .
10:  else
11:     $\alpha \leftarrow \alpha + 1$ .
12:    if  $\alpha > q/2$  then
13:       $\alpha \leftarrow 4$ .
14:    end if
15:     $S \leftarrow S^*$ .
16:    Apply Algorithm 6 to  $S$  with parameter  $\alpha$ , and let the resulting solution be  $S$ .
17:  end if
18: end while
19: return  $S^*$ .
```

at each venue j , denoted by $c_{(i,i'),j} = c_{ij} + c_{i'j}$. When modifying a round R (e.g., adding or removing a match during local search), we dynamically maintain the aggregated costs $\sum_{(i,i') \in R} c_{(i,i'),j}$ for each venue $j \in M$. Therefore, the optimal venue assignment and the total travel distance of a round can be updated in $O(m)$ time per match insertion or removal, which accelerates the exploration of the eject-and-inject and other local search neighborhoods.

Furthermore, constructive edge coloring and Kempe-chain modifications repeatedly traverse alternating paths in the graph. To make these traversals efficient, we track visited vertices using a generational flat array rather than tree-based or hash-based sets. This enables $O(1)$ membership checks and updates and reduces overhead in repeated coloring and perturbation operations.

6 Computational Experiments

This section evaluates the performance of the mathematical formulation introduced in Section 3 and the heuristic methods proposed in Section 5. We first describe the computational environment and benchmark instances, and then present the computational results.

6.1 Computational environment and benchmark instances

All computational experiments were conducted on a PC with an Intel Xeon E-2286G CPU (4.00 GHz) and 64 GB of memory. The mixed-integer programming (MIP) models were solved using Gurobi Optimizer 10.0.0, and the proposed iterated local search (ILS) heuristic was implemented in C++. Both the MIP solver and the heuristic were run in single-threaded mode to ensure a fair comparison. The time limit for each run was set to 3600 seconds for all methods tested. Let UB denote the objective value of the best feasible solution found and LB the best valid lower bound obtained by the solver. Following Laporte and Toth [20], we calculate the optimality gap as $(UB - LB)/LB \times 100\%$ whenever $LB > 0$. This lower-bound-normalized measure provides a conservative assessment of the quality of the incumbent solution for a minimization problem. When $LB = 0$ or the calculated gap is at least 100%, we report the gap as “ ≥ 100.0 ”.

To evaluate the proposed methods comprehensively, we generated a diverse set of benchmark instances. Because no standard benchmark dataset is available for the CRR-TSP-FMR, the instances were generated randomly to cover different problem sizes and geographic distributions. The benchmark set consists of 64 instances characterized by three main parameters: the number of teams n , the number of venues m , and the number of available rounds q . The number of teams was set to $n \in \{8, 16, 32, 64\}$, and the number of venues was selected from $m \in \{2, 4, 8, 16\}$. The number of rounds q controls the flexibility of the schedule. To examine the effect of schedule tightness, we generated instances with $q \in \{\frac{5}{4}n, \frac{3}{2}n, \frac{7}{4}n, 2n\}$.

To evaluate the effects of the numbers of venues and rounds fairly, instances sharing the same n use the same randomly generated team coordinates, and instances sharing the same m use the same randomly generated venue coordinates. All problem instances and source code for the proposed ILS heuristic and MIP formulations are publicly available.¹ Each heuristic configuration was executed once per instance with a fixed random seed.

To generate the distance data, we randomly sampled geographic locations for teams and venues to simulate a realistic regional tournament scenario. Both teams and venues are placed on a two-dimensional Euclidean plane of size 1000×1000 . Specifically, the x - and y -coordinates of each team $i \in N$ and each venue $j \in M$ are drawn independently from a continuous uniform distribution on $[0, 1000]$. The travel distance c_{ij} between team i and venue j is then defined as the straight-line Euclidean distance.

6.2 Results on MIP models

Table 1 summarizes the computational results for the MIP models. We compare the time-indexed formulation (7)–(10) without and with the symmetry-breaking constraints (11)–(14), denoted by TI and TI-SB, respectively. Similarly, the assignment-based formulation (15)–(20) without and with the symmetry-breaking constraints (21)–(23) is denoted by AB and AB-SB, respectively. Detailed instance-by-instance results are provided in Appendix A. For each value of n , the table reports the number of instances, out of 16, for which no feasible solution was found or the solver ran

¹<https://doi.org/10.17632/wnbd6cm5tm.1>

out of memory (#f). For the successfully processed instances, it reports the average optimality gap (%gap) and the average time to the best solution (TTB).

Table 1: Summary of MIP results

n	TI			TI-SB			AB			AB-SB		
	#f	%gap	TTB	#f	%gap	TTB	#f	%gap	TTB	#f	%gap	TTB
8	0	0.7	350.1	0	0.8	253.7	0	0.0	0.2	0	0.0	0.3
16	0	5.6	2056.2	0	5.6	1793.9	0	0.0	3.3	0	0.0	13.5
32	0	≥ 100.0	2824.6	0	18.0	2552.6	0	0.0	1060.8	0	≥ 100.0	1316.8
64	0	≥ 100.0	1264.4	0	≥ 100.0	949.2	9	7.6	829.3	13	≥ 100.0	1646.2

As shown in Table 1, the performance of the time-indexed formulations deteriorates rapidly as n increases. The average optimality gap of TI increases from 0.7% for $n = 8$ and 5.6% for $n = 16$ to at least 100% for $n \geq 32$. The symmetry-breaking constraints have little effect for $n \leq 16$, but they substantially improve the average gap for $n = 32$, reducing it to 18.0%. Nevertheless, TI-SB also leaves an average gap of at least 100% for $n = 64$. Thus, symmetry breaking is beneficial for some difficult instances, but it does not prevent the time-indexed formulation from becoming ineffective at the largest scale.

In contrast, AB performs well on small and medium-sized instances. It solves all instances with $n \leq 16$ and 13 of the 16 instances with $n = 32$ to proven optimality; the remaining three $n = 32$ instances have gaps of at most 0.2%. This result indicates that the assignment-based formulation provides substantially stronger bounds than the time-indexed formulation. The effect of symmetry breaking is formulation dependent: although AB-SB also solves all instances with $n \leq 16$, its average gap is at least 100% for $n = 32$. Hence, the additional symmetry-breaking constraints degrade rather than improve the assignment-based formulation on the more difficult instances.

The main limitation of the assignment-based formulation is its scalability. For $n = 64$, AB runs out of memory on nine instances, although it obtains an average gap of 7.6% on the remaining instances. The AB-SB model is even less robust: it fails to find a feasible solution for 13 instances, and its average gap on the remaining instances is at least 100%. These results show a clear transition between $n = 32$, for which the assignment-based formulation remains highly effective, and $n = 64$, for which memory consumption and feasible-solution generation become major bottlenecks. This scalability limitation motivates the proposed heuristic method.

Two additional patterns emerge from the detailed results in Appendix A. First, increasing the number of available venues improves the attainable travel cost but also makes exact optimization more difficult. For example, when $n = 16$ and $q = 20$, the optimal objective value decreases from 139168.9 for $m = 2$ to 78567.5 for $m = 16$, corresponding to a reduction of 43.5%. Over the same range, the optimality gap of TI increases from 0.0% to 25.3%. Thus, the geographic flexibility provided by additional venues can substantially reduce travel, but the larger venue-assignment decision space increases the computational burden.

Second, the benefit of additional rounds exhibits diminishing returns and depends on the number of venues. When $n = 16$ and $m = 2$, increasing q from 20 to 24 reduces the optimal objective value by only 0.3%, and no further improvement is obtained for $q = 28$ or $q = 32$. When $n = 16$ and $m = 8$, the successive objective reductions obtained by increasing q from 20 to 24, from 24 to 28, and from 28 to 32 are 9.8%, 3.6%, and 1.7%, respectively. Moreover, the optimality gap of TI decreases from 17.2% at $q = 20$ to 0.9% at $q = 32$. Hence, relaxing the round limit can improve both solution quality and computational tractability, although its marginal value decreases as more rounds become available.

6.3 Results of the proposed method

We evaluated the proposed ILS on all 64 instances. Solution quality was assessed using the optimality gap relative to the best lower bound obtained from the MIP models. Because the merge-and-split (M&S) neighborhood is computationally efficient, it was included in every tested configuration. We evaluated all eight combinations of the other three neighborhood structures: simple ejection chain (SEC), variable-depth ejection chain (VEC), and eject-and-inject (E&I).

Table 2: Average optimality gaps for different local search combinations

neighborhood			number of teams n				
SEC	VEC	E&I	8	16	32	64	64 _{AB}
off	off	off	0.00	0.00	0.22	10.70	0.27
off	off	on	0.00	0.00	0.29	10.83	0.24
off	on	off	0.00	0.00	0.17	10.55	0.16
off	on	on	0.00	0.00	0.21	10.67	0.15
on	off	off	0.00	0.00	0.17	10.57	0.18
on	off	on	0.00	0.00	0.24	10.66	0.16
on	on	off	0.00	0.00	0.15	10.50	0.13
on	on	on	0.00	0.00	0.19	10.59	0.14

Table 2 reports the average optimality gap for each of the eight combinations of SEC, VEC, and E&I. The labels “on” and “off” indicate whether the corresponding neighborhood is enabled. Each entry in the columns labeled 8, 16, 32, and 64 is averaged over the 16 instances with the corresponding number of teams, covering all combinations of m and q . The column 64_{AB} restricts the average to the seven instances with $n = 64$ for which AB obtains a feasible solution. Detailed instance-by-instance gaps and times to the best solution are provided in Appendix B.

All eight configurations attain a gap of 0.00% for $n = 8$ and $n = 16$, indicating that the common M&S neighborhood is already sufficient for the small instances. Differences among the additional neighborhoods become apparent for $n \geq 32$. The configuration with SEC and VEC enabled and E&I disabled achieves the smallest average gap for both $n = 32$ and $n = 64$, with gaps of 0.15% and 10.50%, respectively. SEC and VEC are therefore complementary, although either neighborhood alone also

improves upon the configuration using only M&S. On the seven instances represented by 64_{AB} , the same configuration attains an average gap of only 0.13%, compared with 7.6% for AB. Moreover, every ILS configuration has an average gap of at most 0.27% on this subset, which is comparable to the gaps obtained for $n = 32$. This contrast suggests that the large average gaps over the complete set of $n = 64$ instances are attributable, at least in substantial part, to weak lower bounds on the instances for which the exact formulations are ineffective, rather than solely to poor heuristic solutions. Consequently, the reported gaps for these difficult instances should be interpreted as conservative quality guarantees that may overestimate the actual distance from optimality. The results on 64_{AB} also show that the proposed heuristic finds substantially better feasible solutions than AB even on the instances that the MIP formulation can process successfully.

In contrast, enabling E&I increases the average gap for every corresponding SEC-VEC setting at both $n = 32$ and $n = 64$. Thus, E&I does not provide an improvement in solution quality under the present parameter setting, and the configuration with SEC and VEC but without E&I is selected as the best-performing variant. Compared with the MIP approaches, the proposed ILS also provides greater robustness on the largest instances: the time-indexed formulations leave large gaps, whereas the assignment-based formulations frequently encounter memory limitations or fail to obtain a feasible solution. The ILS, by contrast, produces a feasible schedule for every tested instance while maintaining a moderate average gap.

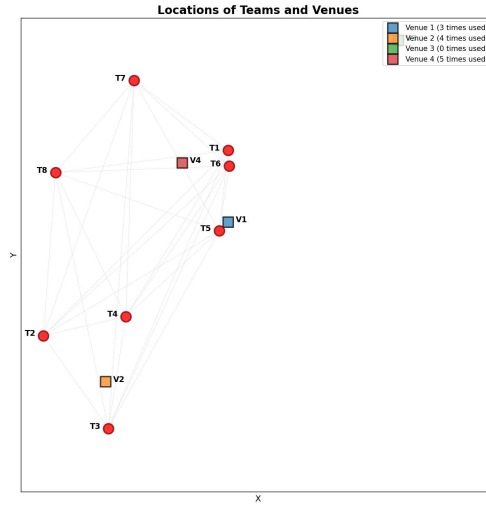


Fig. 1: Locations of teams and venues for the instance $n = 8$, $m = 4$, $q = 12$

Figure 1 shows the positions of teams and venues for the instance with $n = 8$, $m = 4$, and $q = 12$, whereas Figure 2 shows the corresponding schedule obtained by the proposed method.

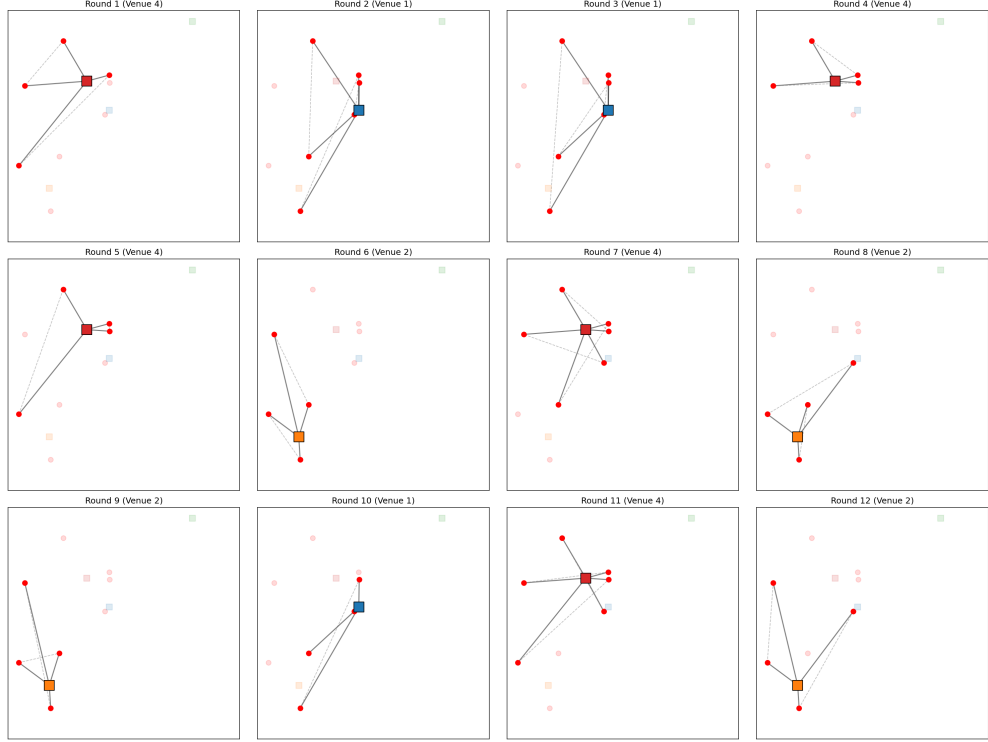


Fig. 2: Resulting schedule for the instance $n = 8$, $m = 4$, $q = 12$

7 Conclusion

This study addresses the centralized round-robin tournament scheduling problem with flexible matches per round (CRR-TSP-FMR), in which the objective is to minimize total travel distance.

First, we proposed two mathematical programming formulations: a time-indexed (TI) model that explicitly indexes candidate rounds, and an assignment-based (AB) model that treats matches as jobs assigned to venue-use slots, along with valid symmetry-breaking constraints.

Second, we proved that the decision version of CRR-TSP-FMR is NP-complete in the strong sense even when all travel costs are restricted to $\{0, 1, 3\}$. This result establishes that the computational difficulty is intrinsic and does not arise from large or complicated cost values.

Third, we identified structural properties and polynomially solvable cases, notably an exact optimality test for the two-venue case based on a lower bound derived from edge coloring of threshold graphs.

Fourth, we developed an iterated local search (ILS) heuristic featuring a fast initial-solution construction strategy based on the circle method and constructive edge coloring. The heuristic combines an adaptive framework with best-improvement

local search using several neighborhoods, including merge-and-split, ejection-chain, and eject-and-inject moves. These neighborhoods are designed to directly exploit the problem structure.

Finally, computational experiments were conducted on 64 instances with $n \in \{8, 16, 32, 64\}$, $m \in \{2, 4, 8, 16\}$, and $q \in \{\frac{5}{4}n, \frac{3}{2}n, \frac{7}{4}n, 2n\}$ under a time limit of 3600 seconds. The results show that the mathematical models perform well on smaller instances because they may prove optimality, whereas on medium and large instances the proposed ILS heuristic finds high-quality solutions, achieving a mean optimality gap of 0.15% for $n = 32$. In particular, when a sufficient number of venues is available, splitting rounds reduces the set of participating teams in each round, which makes it easier to choose venues close to team home bases and thereby reduce total travel distance.

Future work includes incorporating additional constraints motivated by real-world elementary- and junior-high-school sports activities, such as venue availability, match start times, referee assignments, the avoidance of overly dense match sequences, and minimum-rest requirements. Another promising direction is to relax the requirement that each team play at most once per round by allowing each team to play up to a fixed number of matches. The objective could also be extended from pure travel minimization to a multi-objective setting that incorporates break minimization and fairness among teams. The inclusion of break minimization would make the sequence of rounds an additional decision, whereas fairness could be measured by, for example, the maximum or variance of team travel distances. From the viewpoint of heuristic design, further improvements may be achieved by diversifying the neighborhood structures beyond splitting and merging, adopting a more systematic approach to parameter tuning, and evaluating the method on larger real-world data sets.

Declarations

Funding

This work was supported by JSPS KAKENHI Grant Number 25K00180.

References

- [1] Ribeiro, C.C.: Sports scheduling: Problems and applications. *International Transactions in Operational Research* **19**(1-2), 201–226 (2012) <https://doi.org/10.1111/j.1475-3995.2011.00819.x>
- [2] Kendall, G., Knust, S., Ribeiro, C.C., Urrutia, S.: Scheduling in sports: An annotated bibliography. *Computers & Operations Research* **37**(1), 1–19 (2010) <https://doi.org/10.1016/j.cor.2009.05.013>
- [3] Rasmussen, R.V., Trick, M.A.: Round robin scheduling – a survey. *European Journal of Operational Research* **188**(3), 617–636 (2008) <https://doi.org/10.1016/j.ejor.2007.05.046>

- [4] Ribeiro, C.C., Urrutia, S., Werra, D.: A tutorial on graph models for scheduling round-robin sports tournaments. *International Transactions in Operational Research* **30**(6), 3267–3295 (2023) <https://doi.org/10.1111/itor.13290>
- [5] Lewis, R., Thompson, J.: On the application of graph colouring techniques in round-robin sports scheduling. *Computers & Operations Research* **38**(1), 190–204 (2011) <https://doi.org/10.1016/j.cor.2010.04.012>
- [6] Easton, K., Nemhauser, G., Trick, M.: The traveling tournament problem description and benchmarks. In: *International Conference on Principles and Practice of Constraint Programming*, pp. 580–584 (2001). https://doi.org/10.1007/3-540-45578-7_43 . Springer
- [7] Cao, Y., Wu, W., Yagiura, M.: A metaheuristic algorithm for the carry-over effect value minimization problem in round-robin tournaments. *Journal of Advanced Mechanical Design, Systems, and Manufacturing* **16**(4), 0042 (2022) <https://doi.org/10.1299/jamdsm.2022jamdsm0042>
- [8] Zhao, J., Xiao, M.: The APX-hardness of the traveling tournament problem. *Operations Research Letters*, 107311 (2025) <https://doi.org/10.1016/j.orl.2025.107311>
- [9] Knust, S.: Scheduling sports tournaments on a single court minimizing waiting times. *Operations Research Letters* **36**(4), 471–476 (2008) <https://doi.org/10.1016/j.orl.2007.11.006>
- [10] Alçada-Almeida, L., Sousa, N., Coutinho-Rodrigues, J.: A multi-objective model for publicly funded festival planning. *International Transactions in Operational Research* **33**(1), 210–244 (2026) <https://doi.org/10.1111/itor.13500>
- [11] Holyer, I.: The NP-completeness of edge-coloring. *SIAM Journal on Computing* **10**(4), 718–720 (1981) <https://doi.org/10.1137/0210055>
- [12] Vizing, V.G.: On an estimate of the chromatic class of a p-graph. *Diskret. Analiz* **3**, 25–30 (1964)
- [13] Misra, J., Gries, D.: A constructive proof of Vizing’s theorem. *Information Processing Letters* **41**(3), 131–133 (1992) [https://doi.org/10.1016/0020-0190\(92\)90041-S](https://doi.org/10.1016/0020-0190(92)90041-S)
- [14] Chvátal, V., Hammer, P.L.: Aggregation of inequalities in integer programming. *Annals of Discrete Mathematics* **1**, 145–162 (1977) [https://doi.org/10.1016/s0167-5060\(08\)70731-3](https://doi.org/10.1016/s0167-5060(08)70731-3)
- [15] Mahadev, N.V., Peled, U.N.: *Threshold Graphs and Related Topics* vol. 56. Elsevier, Amsterdam (1995). [https://doi.org/10.1016/s0167-5060\(13\)71063-x](https://doi.org/10.1016/s0167-5060(13)71063-x)

- [16] Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs vol. 57. Elsevier, Amsterdam (2004). <https://doi.org/10.1016/c2013-0-10739-8>
- [17] Couto, F., Ferraz, D.A., Klein, S.: New results on edge-coloring and total-coloring of split graphs. *Discrete Applied Mathematics* **360**, 297–306 (2025) <https://doi.org/10.1016/j.dam.2024.09.008>
- [18] Gonzaga, L.G.S., Sousa Cruz, J.B., Almeida, S.M., Silva, C.N.: The overfull conjecture on split-comparability and split-interval graphs. *Discrete Applied Mathematics* **340**, 228–238 (2023) <https://doi.org/10.1016/j.dam.2023.06.040>
- [19] Kempe, A.B.: On the geographical problem of the four colours. *American Journal of Mathematics* **2**(3), 193–200 (1879) <https://doi.org/10.2307/2369235>
- [20] Laporte, G., Toth, P.: A gap in scientific reporting. *4OR* **20**(1), 169–171 (2022) <https://doi.org/10.1007/s10288-021-00483-0>

Appendix A Detailed Results of MIP Models

Tables [A1](#)–[A2](#) report the detailed results for the MIP models. For each configuration, we present the obtained upper bound (UB), lower bound (LB), optimality gap (%gap), time to the best solution (TTB), and total runtime (TTO). Values marked as “t.l.” indicate that the method reached the 3600-second time limit; “-” indicates that no feasible solution was found; and “m.o.” indicates that the solver ran out of memory.

Appendix B Detailed Results of the ILS

Table [B3](#) presents the detailed results of the ILS for all 64 instances and all eight neighborhood configurations. Each configuration is represented by an ordered triple indicating whether SEC, VEC, and E&I are enabled, respectively; the M&S neighborhood is enabled in every configuration. For each instance, the table reports the best valid lower bound obtained from the MIP models and, for each ILS configuration, the resulting optimality gap (%gap) and the time in seconds at which the best solution was found (TTB).

Table A1: Detailed MIP results for the time-indexed formulations

instance			TI						TI-SB					
n	m	q	UB	LB	%gap	TTB	TTO		UB	LB	%gap	TTB	TTO	
8	2	10	14661.8	14661.8	0.0	0.1	0.1		14661.8	14661.8	0.0	0.3	2.9	
8	2	12	14476.7	14476.7	0.0	0.1	0.1		14476.7	14476.7	0.0	0.1	0.1	
8	2	14	14472.8	14472.8	0.0	0.0	0.0		14472.8	14472.8	0.0	0.0	0.0	
8	2	16	14472.8	14472.8	0.0	0.0	0.0		14472.8	14472.8	0.0	0.0	0.0	
8	4	10	13978.9	13978.9	0.0	3.0	66.5		13978.9	13978.9	0.0	37.3	149.9	
8	4	12	13417.3	13417.3	0.0	18.0	24.7		13417.3	13417.3	0.0	1.9	55.2	
8	4	14	13169.9	13169.9	0.0	2.3	3.7		13169.9	13169.9	0.0	4.4	8.1	
8	4	16	13058.9	13058.9	0.0	0.1	0.2		13058.9	13058.9	0.0	0.4	12.8	
8	8	10	13801.2	13435.9	2.7	2314.5	3600.0		13801.2	13259.3	4.1	777.0	3600.0	
8	8	12	12738.0	12738.0	0.0	264.1	363.8		12738.0	12737.9	0.0	682.2	689.0	
8	8	14	12261.9	12261.9	0.0	3.9	6.7		12261.9	12261.9	0.0	2.3	24.6	
8	8	16	12142.7	12142.7	0.0	2.0	5.6		12142.7	12142.7	0.0	3.1	5.3	
8	16	10	13801.2	12987.7	6.3	90.7	3600.0		13801.2	12935.0	6.7	745.6	3600.0	
8	16	12	12725.4	12542.1	1.5	2858.6	3600.0		12725.4	12483.8	1.9	1581.8	3600.0	
8	16	14	12249.5	12249.5	0.0	39.9	178.7		12249.5	12249.5	0.0	216.0	221.5	
8	16	16	12083.8	12083.8	0.0	3.8	9.5		12083.8	12083.8	0.0	7.0	61.5	
16	2	20	139168.9	139168.9	0.0	156.7	517.5		139168.9	139168.9	0.0	190.1	1504.6	
16	2	24	138788.3	138788.3	0.0	9.8	9.6		138788.3	138788.3	0.0	0.3	0.2	
16	2	28	138788.3	138788.3	0.0	4.1	3.9		138788.3	138788.3	0.0	0.3	0.2	
16	2	32	138788.3	138788.3	0.0	2.9	2.7		138788.3	138788.3	0.0	0.4	0.3	
16	4	20	102211.2	94929.1	7.7	3598.4	3600.0		102211.2	94361.4	8.3	1268.5	3600.0	
16	4	24	98780.0	94477.1	4.6	3110.6	3600.0		98780.0	94361.4	4.7	3177.8	3600.0	
16	4	28	95410.2	94361.4	1.1	2830.9	3600.0		95542.5	94361.4	1.3	2366.1	3600.0	
16	4	32	94513.2	94374.2	0.1	2201.1	3600.0		94513.2	94513.2	0.0	141.4	555.6	
16	8	20	80353.6	68558.8	17.2	3079.9	3600.0		80174.6	67875.5	18.1	3589.7	3600.0	
16	8	24	72244.4	68032.6	6.2	2263.1	3600.0		72716.8	67875.5	7.1	3229.5	3600.0	
16	8	28	69696.3	67908.1	2.6	835.0	3600.0		69709.4	67875.5	2.7	1118.0	3600.0	
16	8	32	68487.5	67875.5	0.9	2249.2	3600.0		68487.5	67875.5	0.9	822.3	3600.0	
16	16	20	80280.2	64085.6	25.3	2764.1	3600.0		78943.7	63322.7	24.7	3249.4	3600.0	
16	16	24	71897.2	63377.5	13.4	3242.7	3600.0		70375.3	63322.7	11.1	3495.9	3600.0	
16	16	28	67591.3	63322.7	6.7	3590.3	3600.0		67385.8	63322.7	6.4	3074.0	3600.0	
16	16	32	66122.2	63322.7	4.4	2959.9	3600.0		65797.7	63322.7	3.9	2979.2	3600.0	
32	2	40	412723.5	355087.6	16.2	3571.4	3600.0		414199.5	354822.1	16.7	3114.9	3600.0	
32	2	48	373582.3	354822.1	5.3	3599.7	3600.0		380241.2	354822.1	7.2	2190.2	3600.0	
32	2	56	355849.9	354822.1	0.3	3519.3	3600.0		355171.7	354822.1	0.1	1468.1	3600.0	
32	2	64	354822.1	354822.1	0.0	1218.8	1213.3		354822.1	354822.1	0.0	236.2	234.8	
32	4	40	415731.3	350290.5	18.7	3418.7	3600.0		435164.2	349934.3	24.4	2621.6	3600.0	
32	4	48	375036.4	349934.3	7.2	3519.3	3600.0		387964.0	349934.3	10.9	3473.0	3600.0	
32	4	56	355718.2	349934.3	1.7	3045.5	3600.0		357544.4	349934.3	2.2	1856.5	3601.5	
32	4	64	351180.8	349934.3	0.4	3234.7	3601.3		353051.2	349934.3	0.9	1432.4	3600.0	
32	8	40	-	6561.5	-	-	t.l.		441263.9	305612.0	44.4	2993.5	3600.0	
32	8	48	-	6561.5	-	-	t.l.		400864.9	305612.0	31.2	3573.5	3600.0	
32	8	56	536051.9	4940.4	≥ 100.0	27.3	3600.1		375301.4	305612.0	22.8	1841.6	3600.0	
32	8	64	352764.6	305612.0	15.4	2669.3	3600.1		336384.9	305612.0	10.1	3603.1	3600.0	
32	16	40	416436.4	289137.8	44.0	1771.4	3600.0		444162.9	288998.4	53.7	2187.1	3600.0	
32	16	48	384203.2	289051.2	32.9	3358.7	3600.1		382689.6	288998.4	32.4	3158.4	3600.0	
32	16	56	382284.4	289002.5	32.3	3415.6	3600.1		344905.0	288998.4	19.3	3517.8	3600.0	
32	16	64	332581.8	288998.4	15.1	3174.4	3600.1		323527.6	288998.4	11.9	3573.5	3600.1	
64	2	80	-	1704490.4	-	-	t.l.		2309147.0	1704490.4	35.5	350.3	3600.0	
64	2	96	2303695.9	1704490.4	35.2	1189.7	3601.0		1932905.3	1704490.4	13.4	193.2	3600.0	
64	2	112	-	1704490.4	-	-	t.l.		1880154.9	1704490.4	10.3	176.2	3600.1	
64	2	128	-	1704490.4	-	-	t.l.		1864034.2	1704490.4	9.4	1777.5	3600.0	
64	4	80	-	35863.6	-	-	t.l.		2336767.7	1519038.5	53.8	152.0	3600.0	
64	4	96	2336767.7	4088.9	≥ 100.0	1120.4	3601.0		2336767.7	1519038.5	53.8	257.7	3600.1	
64	4	112	2336767.7	4088.9	≥ 100.0	1293.7	3601.3		2336767.7	1519038.5	53.8	469.9	3600.1	
64	4	128	2336767.7	4088.9	≥ 100.0	1453.7	3601.5		2336767.7	1519038.5	53.8	482.3	3600.1	
64	8	80	-	4074.4	-	-	t.l.		2254852.1	1384398.8	62.9	736.6	3600.1	
64	8	96	-	4074.4	-	-	t.l.		2268736.8	1384398.8	63.9	899.8	3600.1	
64	8	112	-	4074.4	-	-	t.l.		2273676.0	1384398.8	64.2	1007.8	3600.1	
64	8	128	-	3230.3	-	-	t.l.		2293816.9	1384398.8	65.7	1097.1	3600.2	
64	16	80	-	0.0	-	-	t.l.		2713629.6	1224245.3	≥ 100.0	1512.9	3600.2	
64	16	96	-	44.2	-	-	t.l.		2124970.0	1224245.3	73.6	1584.8	3600.2	
64	16	112	-	44.2	-	-	t.l.		2064780.0	1224245.3	68.7	2106.5	3600.2	
64	16	128	-	44.2	-	-	t.l.		2127757.5	1224245.3	73.8	2383.0	3600.4	

Table A2: Detailed MIP results for the assignment-based formulations

instance			AB					AB-SB				
n	m	q	UB	LB	%gap	TTB	TTO	UB	LB	%gap	TTB	TTO
8	2	10	14661.8	14661.8	0.0	0.0	0.0	14661.8	14661.8	0.0	0.0	0.0
8	2	12	14476.7	14476.7	0.0	0.0	0.0	14476.7	14476.7	0.0	0.0	0.0
8	2	14	14472.8	14472.8	0.0	0.0	0.0	14472.8	14472.8	0.0	0.0	0.0
8	2	16	14472.8	14472.8	0.0	0.0	0.0	14472.8	14472.8	0.0	0.0	0.0
8	4	10	13978.9	13978.9	0.0	0.2	0.3	13978.9	13978.9	0.0	0.3	0.3
8	4	12	13417.3	13417.3	0.0	0.0	0.1	13417.3	13417.3	0.0	0.1	0.1
8	4	14	13169.9	13169.9	0.0	0.1	0.1	13169.9	13169.9	0.0	0.1	0.1
8	4	16	13058.9	13058.9	0.0	0.0	0.1	13058.9	13058.9	0.0	0.1	0.1
8	8	10	13801.2	13801.2	0.0	0.5	0.5	13801.2	13801.2	0.0	0.5	0.5
8	8	12	12738.0	12738.0	0.0	0.4	0.6	12738.0	12738.0	0.0	0.6	0.6
8	8	14	12261.9	12261.9	0.0	0.1	0.2	12261.9	12261.9	0.0	0.2	0.2
8	8	16	12142.7	12142.7	0.0	0.2	0.2	12142.7	12142.7	0.0	0.2	0.2
8	16	10	13801.2	13801.2	0.0	0.7	0.7	13801.2	13801.2	0.0	0.7	0.7
8	16	12	12725.4	12725.4	0.0	0.6	1.2	12725.4	12725.4	0.0	1.2	1.2
8	16	14	12249.5	12249.5	0.0	0.3	0.5	12249.5	12249.5	0.0	0.5	0.5
8	16	16	12083.8	12083.8	0.0	0.1	0.3	12083.8	12083.8	0.0	0.3	0.3
16	2	20	139168.9	139168.9	0.0	0.1	2.0	139168.9	139168.9	0.0	2.0	2.0
16	2	24	138788.3	138788.3	0.0	0.1	1.8	138788.3	138788.3	0.0	1.8	1.8
16	2	28	138788.3	138788.3	0.0	0.1	2.6	138788.3	138788.3	0.0	2.6	2.6
16	2	32	138788.3	138788.3	0.0	1.9	1.9	138788.3	138788.3	0.0	1.8	1.8
16	4	20	102211.2	102211.2	0.0	0.3	3.5	102211.2	102211.2	0.0	3.5	3.5
16	4	24	98702.6	98702.6	0.0	1.6	7.4	98702.6	98702.6	0.0	5.9	7.4
16	4	28	95410.2	95410.2	0.0	0.3	3.2	95410.2	95410.2	0.0	3.2	3.2
16	4	32	94513.2	94513.2	0.0	0.9	7.7	94513.2	94513.2	0.0	7.7	7.7
16	8	20	80120.1	80120.1	0.0	12.2	57.3	80120.1	80120.1	0.0	48.8	57.3
16	8	24	72244.4	72244.4	0.0	1.4	5.0	72244.4	72244.4	0.0	5.0	5.0
16	8	28	69671.1	69671.1	0.0	0.8	5.1	69671.1	69671.1	0.0	5.1	5.1
16	8	32	68465.1	68465.1	0.0	3.0	10.2	68465.1	68465.1	0.0	10.2	10.2
16	16	20	78567.5	78567.5	0.0	4.8	47.4	78567.5	78567.5	0.0	47.4	47.4
16	16	24	70044.7	70044.7	0.0	2.8	13.9	70044.7	70044.7	0.0	13.9	13.9
16	16	28	67291.9	67291.9	0.0	10.9	26.5	67291.9	67291.9	0.0	26.5	26.5
16	16	32	65726.1	65726.1	0.0	11.2	30.3	65726.1	65726.1	0.0	30.3	30.3
32	2	40	411226.4	411093.4	0.0	2764.0	t.l.	411226.4	411158.8	0.0	3551.4	t.l.
32	2	48	371790.0	371790.0	0.0	45.4	393.9	371790.0	371790.0	0.0	393.9	393.9
32	2	56	355015.6	355015.6	0.0	3.1	290.6	355015.6	355015.6	0.0	290.6	290.6
32	2	64	354822.1	354822.1	0.0	2.8	3418.1	354822.1	354822.1	0.0	3418.1	3418.1
32	4	40	411226.4	411158.8	0.0	1262.9	t.l.	411672.6	411158.8	0.1	2199.3	t.l.
32	4	48	371790.0	371790.0	0.0	45.6	528.3	371790.0	371790.0	0.0	528.3	528.3
32	4	56	352660.9	352660.9	0.0	11.0	828.2	352660.9	352660.9	0.0	828.2	828.2
32	4	64	350743.9	350743.9	0.0	6.8	618.3	350743.9	350743.9	0.0	618.3	618.3
32	8	40	410318.2	410318.2	0.0	1843.6	t.l.	410717.9	410318.2	0.1	3435.6	t.l.
32	8	48	368355.8	368341.8	0.0	1263.1	t.l.	454758.3	0.0	≥ 100.0	158.4	t.l.
32	8	56	338633.9	338539.4	0.0	990.8	t.l.	342926.0	338519.1	1.3	2331.2	t.l.
32	8	64	322949.8	322949.8	0.0	30.8	1834.6	322949.8	322949.8	0.0	1834.6	1834.6
32	16	40	358884.5	358884.5	0.0	2312.7	t.l.	450219.3	358428.6	25.6	283.9	t.l.
32	16	48	337678.6	336868.2	0.2	1888.9	t.l.	450219.3	0.0	≥ 100.0	339.4	t.l.
32	16	56	319660.3	319409.0	0.1	3237.3	t.l.	450219.3	0.0	≥ 100.0	395.8	t.l.
32	16	64	306921.2	306459.9	0.2	1263.9	t.l.	450219.3	0.0	≥ 100.0	461.4	t.l.
64	2	80	1973639.5	1973639.5	0.0	1454.9	t.l.	2056316.0	655.7	≥ 100.0	1363.6	t.l.
64	2	96	1799387.5	1798748.6	0.0	2627.4	t.l.	1917950.9	655.7	≥ 100.0	1626.8	t.l.
64	2	112	1708569.1	1708569.1	0.0	505.2	t.l.	1799043.8	655.7	≥ 100.0	1948.1	t.l.
64	2	128	1704490.4	1704490.4	0.0	82.3	82.3	-	-	-	-	t.l.
64	4	80	2336767.7	1948968.9	19.9	64.2	t.l.	-	-	-	-	t.l.
64	4	96	2336767.7	1755212.3	33.1	79.6	t.l.	-	-	-	-	t.l.
64	4	112	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	4	128	1563830.4	1563830.4	0.0	991.6	991.6	-	-	-	-	t.l.
64	8	80	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	8	96	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	8	112	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	8	128	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	16	80	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	16	96	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	16	112	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.
64	16	128	m.o.	m.o.	m.o.	m.o.	m.o.	-	-	-	-	t.l.

Table B3: Detailed results for all ILS configurations

instance			neighborhood settings (SEC, VEC, and E&I)															
			off,off,off		off,off,on		off,on,off		off,on,on		on,off,off		on,off,on		on,on,off		on,on,on	
n	m	q	LB	%gap	TTB	%gap	TTB	%gap	TTB	%gap	TTB	%gap	TTB	%gap	TTB	%gap	TTB	
8	2	10	14661.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	2	12	14476.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	2	14	14472.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	2	16	14472.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	4	10	13978.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	4	12	13417.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	4	14	13169.9	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	4	16	13058.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	8	10	13801.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	8	12	12738.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	
8	8	14	12261.9	0.0	0.0	0.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	8	16	12142.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	16	10	13801.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	
8	16	12	12725.4	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.1	
8	16	14	12249.5	0.0	0.1	0.0	0.3	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.0	0.0	
8	16	16	12083.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
16	2	20	139168.9	0.0	1607.5	0.0	3.5	0.0	0.0	0.0	2.2	0.0	0.1	0.0	0.6	0.0	0.6	
16	2	24	138788.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
16	2	28	138788.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
16	2	32	138788.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
16	4	20	102211.2	0.0	1.5	0.0	0.2	0.0	0.1	0.0	0.2	0.0	0.0	0.0	0.0	0.1	0.0	
16	4	24	98702.6	0.0	0.9	0.0	0.1	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
16	4	28	95410.2	0.0	0.5	0.0	165.3	0.0	2.4	0.0	104.3	0.0	17.8	0.0	186.8	0.0	357.0	
16	4	32	94513.2	0.0	11.8	0.0	1.0	0.0	1.8	0.0	0.8	0.0	4.5	0.0	4.2	0.0	0.8	
16	8	20	80120.1	0.0	96.1	0.0	1.6	0.0	3.3	0.0	2.5	0.0	0.7	0.0	0.1	10.4	0.1	
16	8	24	72244.4	0.0	0.0	0.0	0.1	0.0	0.1	0.0	0.1	0.0	0.1	0.0	0.1	0.0	0.1	
16	8	28	69671.1	0.0	0.1	0.0	1.6	0.0	0.5	0.0	4.7	0.0	0.4	0.0	4.3	0.0	0.4	
16	8	32	68465.1	0.0	194.0	0.0	31.5	0.0	8.6	0.0	3432.5	0.0	316.0	0.0	1289.1	0.0	3415.2	
16	16	20	78567.5	0.0	0.9	0.0	64.9	0.0	7.8	0.0	29.9	0.0	51.4	0.0	78.6	0.0	143.0	
16	16	24	70044.7	0.0	34.6	0.0	2743.8	0.0	0.4	0.0	1.3	0.0	622.4	0.0	2758.2	0.0	8.3	
16	16	28	67291.9	0.0	357.5	0.0	2810.5	0.0	24.5	0.0	47.5	0.0	45.5	0.0	73.4	0.0	133.5	
16	16	32	65726.1	0.0	16.4	0.0	116.2	0.0	12.9	0.0	3349.6	0.0	8.1	0.0	3216.2	0.0	1213.9	
32	2	40	411158.8	0.1	2580.3	0.1	2701.8	0.1	706.3	0.1	708.9	0.1	88.0	0.1	123.2	0.1	1215.8	
32	2	48	371790.0	0.1	3453.7	0.1	2484.3	0.1	1999.1	0.1	3325.0	0.1	1860.2	0.1	1612.8	0.1	1126.2	
32	2	56	355015.6	0.0	10.9	0.0	1.7	0.0	0.2	0.0	4.1	0.0	0.2	0.0	0.2	0.0	0.2	
32	2	64	354822.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
32	4	40	411158.8	0.1	40.0	0.1	51.4	0.1	699.9	0.1	3360.4	0.1	16.0	0.1	16.8	0.1	127.0	
32	4	48	371790.0	0.2	3559.8	0.1	3321.5	0.1	2520.7	0.1	3113.4	0.1	1031.8	0.1	3502.4	0.1	1693.9	
32	4	56	352660.9	0.0	1111.0	0.0	1613.6	0.0	574.6	0.0	3028.6	0.0	410.5	0.0	142.3	0.0	1542.6	
32	4	64	350743.9	0.0	27.8	0.0	89.0	0.0	0.4	0.0	922.1	0.0	1575.8	0.0	27.9	0.0	28.4	
32	8	40	410318.2	0.1	844.6	0.1	3013.2	0.1	2513.7	0.1	3243.9	0.1	302.9	0.1	2492.6	0.1	2741.6	
32	8	48	368341.8	0.4	2547.3	0.6	3320.8	0.3	2407.7	0.2	1879.0	0.3	1824.3	0.3	3084.3	0.3	3531.0	
32	8	56	338539.4	0.3	3557.6	0.5	2011.9	0.1	3431.7	0.2	1851.4	0.1	3206.6	0.3	3283.1	0.2	3517.0	
32	8	64	322949.8	0.2	2783.8	0.4	3107.2	0.1	3514.8	0.2	3580.8	0.1	3220.9	0.3	3031.7	0.1	3459.5	
32	16	40	358884.5	0.3	1173.1	0.4	2068.0	0.2	1259.1	0.3	3336.7	0.2	772.6	0.4	1682.1	0.2	2235.4	
32	16	48	336868.2	0.7	2905.2	0.8	3519.7	0.7	3565.1	0.8	2519.8	0.6	3082.8	0.8	2735.7	0.5	1988.1	
32	16	56	319409.0	0.5	2728.2	0.8	872.5	0.5	3175.7	0.6	2277.3	0.4	2508.9	0.6	2787.4	0.4	3573.5	
32	16	64	306459.9	0.5	3074.8	0.7	2794.0	0.3	3387.8	0.6	3393.0	0.3	2730.0	0.6	2425.2	0.3	2949.6	
64	2	80	1973639.5	0.2	3378.1	0.1	2935.8	0.1	495.2	0.1	3373.8	0.1	2449.3	0.0	2280.2	0.0	3502.0	
64	2	96	1798748.6	0.4	3289.5	0.1	3448.8	0.1	3199.5	0.0	299.1	0.2	2489.6	0.0	710.1	0.1	3050.7	
64	2	112	1708569.1	0.0	326.4	0.0	155.8	0.0	150.0	0.0	52.6	0.0	3.3	0.0	27.8	0.0	17.5	
64	2	128	1704490.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
64	4	80	1948968.9	0.3	3274.5	0.3	3319.9	0.2	2842.5	0.2	2850.7	0.2	3295.5	0.2	3189.4	0.2	3028.3	
64	4	96	1755212.3	0.8	2012.4	0.9	3319.6	0.5	1545.8	0.5	2115.2	0.6	3166.1	0.7	2927.9	0.5	3309.4	
64	4	112	1519038.5	7.8	3274.6	7.8	2773.3	7.5	3326.7	7.5	1485.4	7.7	3287.4	7.6	2670.3	7.5	2401.0	
64	4	128	1563830.4	0.2	3375.4	0.3	3263.5	0.1	2961.9	0.1	3313.7	0.1	2975.6	0.2	3100.5	0.1	3458.9	
64	8	80	1384398.8	35.9	2965.8	35.9	3298.5	35.8	3579.7	35.9	1810.1	35.8	2715.8	35.9	3013.2	35.8	2213.3	
64	8	96	1384398.8	21.3	3284.0	21.4	2407.3	20.9	3154.4	21.1	1494.2	21.1	2716.7	21.2	3275.7	21.0	2966.5	
64	8	112	1384398.8	11.6	3204.7	11.8	2903.6	11.4	3394.4	11.5	3442.6	11.4	2913.9	11.5	3591.2	11.3	1040.1	
64	8	128	1384398.8	6.3	3322.0	7.1	3404.3	6.3	3224.4	6.9	3600.1	6.2	2950.2	6.8	3453.6	6.2	3457.9	
64	16	80	1224245.3	36.4	3580.4	36.5	2096.8	36.2	3048.8	36.2	1212.2	36.1	3365.9	36.3	1367.4	36.1	3006.4	
64	16	96	1224245.3	24.6	3425.8	24.5	1208.9	24.3	3072.6	24.4	2130.7	24.3	3291.1	24.3	1664.1	24.2	2883.6	
64	16	112	1224245.3	15.9	3586.2	16.4	2738.5	15.9	1428.6	16.3	3549.5	15.8	2969.6	16.1	3534.8	15.8	3300.0	
64	16	128	1224245.3	9.4	3567.1	10.2	3600.6	9.2	3063.3	9.9	3561.9	9.3	609.6	9.8	3552.7	9.2	2597.9	