An exact approach for the Train Single-Routing Selection Problem

Anna Livia Croella[a,*], Fabio Furini[b], Ivana Ljubić[c], Bianca Pascariu[d], Paola Pellegrini[d], Pablo San Segundo[e]

[a] *Universitas Mercatorum. Rome. Italy.*

[b] *Department of Computer Control and Management Engineering Antonio Ruberti. Sapienza University of Rome. Rome. Italy.*

[c] *ESSEC Business School. Cergy-Pontoise. France.*

[d] *Universite Gustave Eiffel COSYS-ESTAS F-59650 Villeneuve dAscq. France.*

[e] *Universidad Politecnica de Madrid (UPM). Centre for Automation and Robotics (CAR). Madrid. Spain.*

Abstract

Given a set of train routes with route costs and a set of compatible route pairs with pairing costs, the Train Single-Routing Selection Problem (TSRSP) seeks to assign one route to each train, minimizing the total cost while ensuring pairwise compatibility among the selected routes. This problem is of significant practical relevance in rail traffic management and it is a special case of the Train Routing Selection Problem. We introduce a Binary Quadratic Programming formulation for the TSRSP and apply a linearization technique from the literature that requires a linear number of additional variables and constraints. This yields a novel and effective Mixed Integer Linear Programming (MILP) formulation for the TSRSP. By exploiting the structure of the problem, we further propose a method to strengthen the Linear Programming (LP) relaxation of the MILP formulation, substantially improving its computational performance. The resulting formulation is the first to efficiently solve real-world TSRSP instances to optimality within short computational times—instances that, until now, have only been addressed using heuristic approaches in the literature. Extensive computational results show that our MILP model consistently outperforms existing formulations, both in terms of computational time and the number of instances solved to optimality, primarily due to the strength of its LP relaxation.

Keywords: Rail Traffic Management, Weighted Clique Problems, Binary Quadratic Programming, Linearization Techniques, Integer Linear Programming

*Corresponding author. E-mail address: `annalivia.croella@unimercatorum.it`

1. Introduction

Railway Traffic Management (RTM) focuses on coordinating train movements to ensure both efficiency and punctuality. It involves planning train routes and schedules to minimize delay propagation in the event of disturbances. The two key tasks are: *i*) selecting the sequence of tracks that allows trains to move through a control area from their origin to their destination, and *ii*) determining the order for trains to pass on shared tracks. Different approaches have been proposed in the literature to address these tasks; interested readers are referred to [8, 10, 13, 18, 23, 44] for further details. In summary, two main types of approaches exist: some tackle the overall train routing and scheduling problem [11, 19, 29], while others focus on pure train scheduling, considering a single route per train [12, 14, 15, 16]. In general, the approaches of the former type assume that track sequences and passing orders must be selected concurrently for the effective exploitation of infrastructure capacity. The concurrent selection typically comes at the cost of accepting suboptimal solutions due to computational resource limitations. The approaches of the latter type assume, instead, that the specific track sequences to be traversed can be selected a priori, and the most effective use of computational resources is to focus on approaching the global optimum of the scheduling problem. Both assumptions are adopted in the approach by Pellegrini et al. [36], where the pure scheduling problem is solved first, and if computational resources remain, the solution found is used as a warm start for the routing and scheduling problem solution.

1.1. Train routing selection problems

A key challenge for all RTM approaches is the assignment of routes to trains, whether determining multiple routes or exactly one. This problem becomes particularly complex in junctions and station areas, where dozens of alternative routes may be available for each train. To address this challenge, the *Train Routing Selection Problem* (TRSP) aims to find a given number of route assignments to the trains with minimum cost, ensuring pairwise compatibility between selected routes and with the additional requirement that any two assignments differ in at least one train-route selection. The total assignment cost has two components: one related to the route itself and the other originating from interactions between pairs of routes and reflecting their interference. The TRSP was first introduced in the literature by Samà et al. [40], where the authors proposed an *Ant Colony Optimization* (ACO) heuristic algorithm to solve it. This algorithm was further refined by Pascariu et al. [34] and Sharma et al. [43], proposing alternative methods for setting assignment costs and incorporating parallelism into the algorithm. Samà et al. [39] and Pascariu et al. [35] demonstrated that integrating TRSP solutions into RTM significantly improves traffic performance, regardless of whether the specific disruption is known when solving the TRSP, the delay function being minimized, or the approach used for routing and scheduling.

In this paper, we address a specific case of the TRSP, where a single route is assigned to each train. We call this problem the *Train Single-Routing Selection Problem* (TSRSP). This variant is important because an optimal TSRSP solution can serve as the designated route for the trains in RTM approaches. On the one hand, the TSRSP is crucial in RTM: selecting a single route per train is a requirement in scheduling-based RTM approaches, and it can provide a high-quality warm start for methods that involve rerouting. On the other hand, it has potential applications in tactical traffic management problems, where train routing optimization is often overlooked. For instance, a single route per train must be specified in the timetable, whether it is planned months in advance or adjusted shortly before operations to handle unexpected disruptions, such as, e.g., track unavailability. Additionally, an efficient TSRSP approach can be integrated into an iterative framework to generate different assignments, serving as the basis for advanced exact and heuristic algorithms for the TRSP.

The TSRSP has been previously addressed only by Pascariu et al. [33], where an *Integer Linear Programming* (ILP) model is proposed and tested. This ILP model was used to develop a heuristic approach, which was then compared to the ACO heuristic algorithm from the literature. However, the heuristic solutions derived from this ILP model were inferior to those obtained by the ACO algorithm in short computing times. Additionally, the ILP model of Pascariu et al. [33] struggled to compute optimal solutions even with extended computational time.

1.2. Contributions and remainder of the article

The main contributions of this article are summarized as follows. In Section 2, we mathematically formalize the TSRSP and provide its graph-theoretic interpretation. In Section 3, we present a *Binary Quadratic Programming* (BQP) formulation and discuss the associated solution approaches. Section 4 contains the core contribution of this article: a novel and effective *Mixed Integer Linear Programming* (MILP) formulation for the TSRSP, derived by applying the linearization technique of Glover [21] to the BQP formulation. This linearization requires only a linear number of additional variables and constraints, making it both scalable and computationally efficient. Moreover, by exploiting the specific structure of the TSRSP, we propose a strengthening of the *Linear Programming* (LP) relaxation of the novel MILP formulation, which yields significant improvements in computational performance. Section 5 describes two additional formulations of the TSRSP used for comparison purposes: the first is based on the standard linearization technique, which requires a quadratic number of additional variables and constraints; the second is the only ILP formulation currently available in the literature, proposed by Pascariu et al. [33]. Section 6 analyzes and compares the formulations in terms of the number of variables and constraints and the strength of their LP relaxations. Extensive computational results are reported in Section 7. These tests show that the novel MILP formulation is capable of solving real-world TSRSP instances to optimality within very short computing times. For particularly large instances, the model also produces high-quality heuristic solutions with small optimality gaps, thereby offering a strong trade-off between efficiency and accuracy. The experiments include instances derived from the French railway network and demonstrate the superior performance of the new formulation compared to the existing ones. Finally, Section 8 concludes the paper and outlines directions for future research.

2. The Train Single-Routing Selection Problem

An instance of the TSRSP consists of a set $\mathcal{K} = \{1, 2, \ldots, k\}$ of k *trains* and a set $\mathcal{V} = \{1, 2, \ldots, n\}$ of n *routes* for the trains. The route set $\mathcal{V}$ is partitioned into k subsets $\mathcal{V}_i$, one for each train $i \in \mathcal{K}$, i.e., we have:

$$\mathcal{V} = \bigcup_{i \in \mathcal{K}} \mathcal{V}_i \quad \text{and} \quad \mathcal{V}_i \cap \mathcal{V}_j = \emptyset, \quad i, j \in \mathcal{K}, i < j. \tag{1}$$

For each pair of distinct trains $i, j \in \mathcal{K}$, with $i < j$, we are given a set $\mathcal{E}_{ij}$ containing pairs $\{r, s\}$ of *compatible routes*, with $r \in \mathcal{V}_i$ and $s \in \mathcal{V}_j$. We denote $\mathcal{E}$ the set of all compatible pairs of routes, i.e., we have:

$$\mathcal{E} = \bigcup_{i,j \in \mathcal{K},\, i<j} \mathcal{E}_{ij}. \tag{2}$$

A pair of routes is considered compatible if they can be assigned to two distinct trains simultaneously without compromising the feasibility of the RTM solution. The specific rules determining the compatible pairs of routes are described in detail in [40, 43]. For instance, routes for trains involved in rolling stock reutilization operations, such as in train turnarounds, joining, or splitting at stations, are compatible if their endpoints—where the turnaround, joining, or splitting occurs—align on the same track section. Additionally, compatibility may also be influenced by passenger transfer requirements. Specifically, when passengers need to transfer between arriving and departing trains at a station, the corresponding routes are considered compatible only if the trains stop at nearby platforms. Conversely, if the trains use tracks associated with distant platforms, their routes are classified as *non-compatible.*

2.1. The route and the pairing costs of the TSRSP

Each route $r \in \mathcal{V}$ has a nonnegative *route cost* $p_r \geq 0$ and each pair of routes $\{r, s\} \in \mathcal{E}$ has a nonnegative *pairing cost* $c_{rs} \geq 0$. This cost model is designed to provide a reliable estimate of the impact of route choices on railway operations. Specifically, it reflects how selecting a particular train route may contribute to delays in the schedule and how different pairs of route choices can influence scheduling decisions. The route cost p_r, for each route $r \in \mathcal{V}$, accounts for potential delays a train may experience due to increased travel time along the selected route. For example, selecting a longer-than-planned route can lead to delays, especially if no buffer time is available. The value of the route cost is calculated as the nonnegative difference between the minimum running time of a train on the selected route and its minimum running time on the originally planned route. Here, the minimum running time represents the time required for the train to complete its journey along a route without any conflicts or delays. If the minimum running time on the chosen route is less than or equal to that on the planned route, the route cost is zero. The pairing cost c_{rs}, for each pair of routes $\{r, s\} \in \mathcal{E}$, quantifies the potential delays that may arise from the scheduling decisions made when that pair of compatible train routes is assigned simultaneously. These scheduling decisions are necessary when multiple trains share the same track sections, as a passing order must be established for each shared section, and the train passing second may then experience a delay. In particular, the pairing cost is calculated by identifying the shared track section that presents the maximum potential delay for each possible train order, ensuring that costs accurately reflect the worst-case delay scenario for scheduling decisions. The pairing cost is zero when the pair of train routes shares no track section. The specific rules determining these costs are detailed in [40].

2.2. The compatibility graph of the TSRSP

A TSRSP instance can be represented by a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices, associated with the routes of the trains, and $\mathcal{E}$ is the set of edges, associated with pairs of compatible routes. Since the vertex set $\mathcal{V}$ is partitioned into k subsets and the edge set $\mathcal{E}$ contains only edges between different subsets of the partition, $\mathcal{G}$ is a k-partite graph.

Each subset of this partition is referred to as a *layer* which is, by construction, associated with a train. The graph $\mathcal{G}$ is weighted on the vertices and the edges. The weight of a vertex corresponds to the route cost, while the weight of an edge corresponds to the pairing cost. We call the graph $\mathcal{G}$ the *compatibility graph* of a TSRSP instance.

For each vertex $r \in \mathcal{V}$, let

$$\mathcal{N}(r) = \left\{ s \in \mathcal{V} : \{r, s\} \in \mathcal{E} \right\} \tag{3}$$

be the *neighborhood* of vertex r, i.e., the set of vertices in $\mathcal{V}$ that belong to different trains and are compatible with r.

For each pair of trains $i, j \in \mathcal{K}$ with $i < j$, let $\tilde{\mathcal{E}}_{ij}$ denote the set of non-compatible route pairs $\{r, s\}$, where $r \in \mathcal{V}_i$ and $s \in \mathcal{V}_j$. The complete set of non-compatible route pairs is then given by

$$\tilde{\mathcal{E}} = \bigcup_{i,j \in \mathcal{K},\, i<j} \tilde{\mathcal{E}}_{ij}. \tag{4}$$

For each route $r \in \mathcal{V}$, let

$$\tilde{\mathcal{N}}(r) = \left\{ s \in \mathcal{V} : \{r, s\} \in \tilde{\mathcal{E}} \right\} \tag{5}$$

be the *anti-neighborhood* of route r, i.e., the set of routes in $\mathcal{V}$ that belong to different trains and are not compatible with r.

We say a compatibility graph $\mathcal{G}$ to be *layer-wise complete* if for every pair of trains $i, j \in \mathcal{K}$, $i < j$, the set $\mathcal{E}_{ij}$ contains all possible edges $\{r, s\}$ with $r \in \mathcal{V}_i$ and $s \in \mathcal{V}_j$. Moreover, we define the *edge density* of $\mathcal{G}$ as the ratio between the number of pairs of compatible routes and the number of all potential pairs of routes between every pair of trains $i, j \in \mathcal{K}$, $i < j$. The edge density, denoted by ε, is then given by the formula:

$$\varepsilon = \frac{|\mathcal{E}|}{|\mathcal{E}| + |\tilde{\mathcal{E}}|} \tag{6}$$

For a *layer-wise* complete compatibility graph, we have $\varepsilon = 1$ since the set $\tilde{\mathcal{E}}$ is empty.

Two vertices of $\mathcal{G}$ are *adjacent* if an edge exists between them. A subset of pairwise adjacent vertices is called a *clique*. Given a clique $\mathcal{C} \subseteq \mathcal{V}$, the *clique weight* $\varphi(\mathcal{C})$ is defined as the sum of the weights of its vertices and the weights of the edges connecting its vertices. Formally, we have:

$$\varphi(\mathcal{C}) = \sum_{r \in \mathcal{C}} p_r + \sum_{\{r,s\} \in \mathcal{E}[\mathcal{C}]} c_{rs} \tag{7}$$

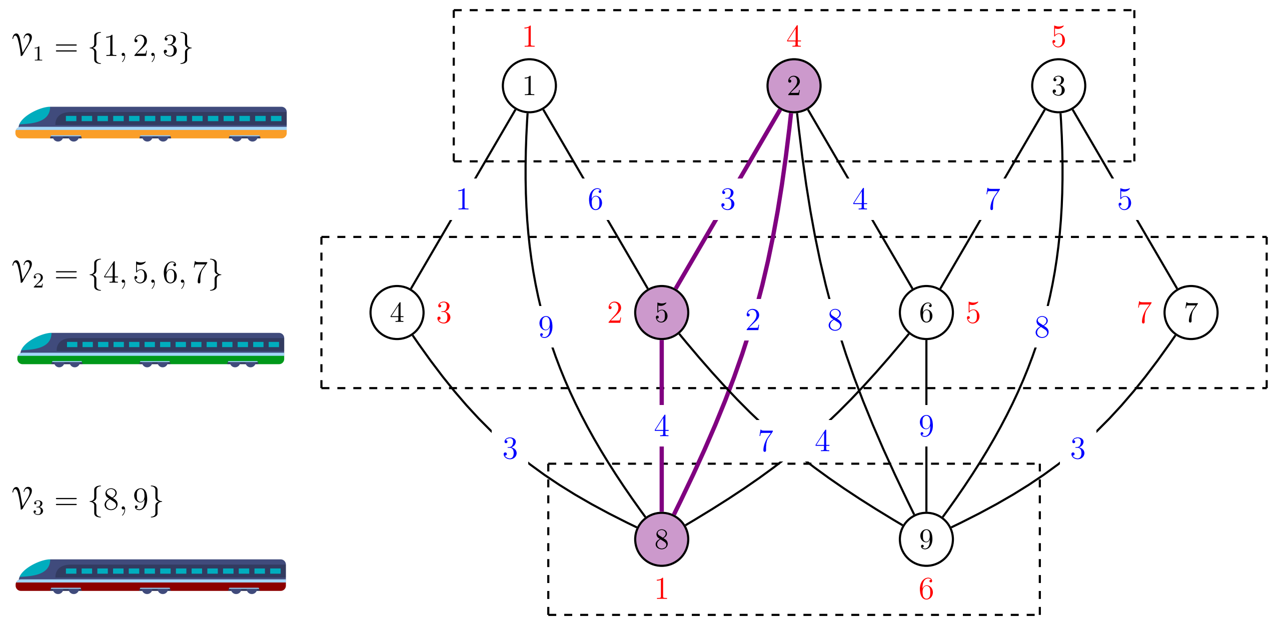
where $\mathcal{E}[\mathcal{C}]$ is the set of edges of the clique, i.e., $\mathcal{E}[\mathcal{C}] = \big\{\{r, s\} \in \mathcal{E} : r, s \in \mathcal{C}\big\}$. Since $\mathcal{G}$ is k-partite, the size of the largest clique it contains is at most k. If a clique of size k, called a k-clique, exists, it must contain a vertex for each layer $\mathcal{V}_i$ with $i \in \mathcal{K}$. We denote a k-clique as $\mathcal{C}^k$ in the remainder of the article.

2.3. The formal problem definition of the TSRSP

The TSRSP can then be formally defined as follows.

Definition 1. *Given a compatibility graph $\mathcal{G}$, the TSRSP calls for finding a k-clique $\mathcal{C}^k \subseteq \mathcal{V}$ in $\mathcal{G}$ of minimum clique weight $\varphi(\mathcal{C}^k)$, or certifying that a k-clique does not exist in $\mathcal{G}$.*

Figure 1: The compatibility graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of a TSRSP instance with $k = 3$ train layers $(\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3)$, $n = 9$ routes, and edge density $\varepsilon = \frac{16}{26}$. Each vertex shows the route index $r \in \mathcal{V}$ (black) and route cost p_r (red). Edge values (blue) represent pairing costs c_{rs}. The violet-filled vertices form the optimal k-clique $\mathcal{C}^k = \{2, 5, 8\}$, assigning trains to routes 2, 5, and 8. The optimal solution value, given by the clique weight, is 16. The edges of $\mathcal{C}^k$, $\mathcal{E}[\{2, 5, 8\}] = \{\{2, 5\}, \{5, 8\}, \{2, 8\}\}$, are highlighted in thick violet.


$\mathcal{V}_1 = \{1, 2, 3\}$
$\mathcal{V}_2 = \{4, 5, 6, 7\}$
$\mathcal{V}_3 = \{8, 9\}$

Determining the existence of a k-clique in a k-partite graph is $\mathcal{NP}$-complete (see e.g., [42]); it follows that the TSRSP is $\mathcal{NP}$-hard. In practice, a k-clique provides an assignment of routes to the k trains; the TSRSP asks for finding an assignment such that the sum of the total cost of the routes together with the total pairing cost is minimized. Figure 1 reports the compatibility graph of a TSRSP instance and an optimal solution to the problem.

Throughout the remainder of this paper, we use the terms vertex and route, as well as layer and train, interchangeably. The terminology vertex and layer is principally employed when discussing the problem in graph-theoretical terms, while route and train are used in contexts related to railway operations and transportation planning.

3. A new BQP formulation for the TSRSP

For each vertex $r \in \mathcal{V}$ in the compatibility graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, let $x_r \in \{0, 1\}$ be a binary variable that takes the value 1 if vertex r is included in a k-clique of $\mathcal{G}$, and 0 otherwise. Using these $|\mathcal{V}|$ binary variables, we propose the following Binary Quadratic Programming (BQP) formulation for the TSRSP:

$$\min_{\boldsymbol{x} \in \{0,1\}^{|\mathcal{V}|}} \left\{ \sum_{r \in \mathcal{V}} p_r \, x_r + \sum_{\{r,s\} \in \mathcal{E}} c_{rs} \, x_r \, x_s \; : \; \boldsymbol{x} \text{ is a } k\text{-clique in } \mathcal{G} \right\}. \tag{8}$$

The quadratic objective function in formulation (8) minimizes the total cost, consisting of route and pairing costs. A route cost for $r \in \mathcal{V}$ is incurred when $x_r = 1$, while a pairing cost for two vertices r and s, connected by an edge $\{r, s\} \in \mathcal{E}$, is incurred when both variables are set to 1, i.e., when $x_r\, x_s = 1$. A polyhedral description of the feasible region of formulation (8), ensuring that the selected routes form k-cliques in $\mathcal{G}$, is provided by the assignment and clique constraints introduced in Section 3.1.

To the best of our knowledge, no existing studies have directly addressed the BQP formulation (8). However, related problems modeled by BQP formulations have been addressed in the literature. Alidaee et al. [2] addressed the maximum edge weight clique problem via unconstrained quadratic programming. In Billionnet [3], a BQP formulation is proposed for the *heaviest k-subgraph problem* that asks for determining a subset of k vertices such that the total edge weight of the induced subgraph is maximized. This problem appears in the literature under alternative names, such as the *p-dispersion-sum problem* [37] or *maximum diversity problem* [27]. A variant is the densest k-subgraph problem, which assumes all edge weights are set to one [24]. All these problems share similarities with the TSRSP, including the presence of graphs with weighted edges and the requirement that feasible solutions consist of subsets of vertices with a fixed cardinality. However, the TSRSP differs in that its feasible solutions must also form a clique. Additionally, the input graph for the TSRSP is k-partite, and the objective function involves minimization, whereas the cited problems typically involve maximization.

3.1. Assignment and clique constraints

In feasible solutions of formulation (8), exactly one vertex must be selected for each layer of the compatibility graph $\mathcal{G}$. To enforce this, we introduce the *assignment constraints*, given by:

$$\sum_{r \in \mathcal{V}_i} x_r = 1, \qquad i \in \mathcal{K}. \tag{9}$$

For each layer $i \in \mathcal{K}$, constraints (9) ensure that exactly one vertex is selected from the set $\mathcal{V}_i$, which represents the vertices available for that layer. These constraints collectively enforce the selection of k distinct routes, one for each train.

Furthermore, to ensure that feasible solutions of formulation (8) form a clique in $\mathcal{G}$, only vertex pairs connected by edges must be selected. To this end, we introduce two classes of *clique constraints* that enforce the selection of subsets of pairwise compatible routes for the trains. The first class of clique constraints is called *non-edge constraints* and reads as follows:

$$x_r + x_s \leq 1, \qquad \{r, s\} \in \tilde{\mathcal{E}}. \tag{10}$$

For each pair of non-compatible routes $\{r, s\} \in \tilde{\mathcal{E}}$, constraints (10) ensure that at most one of the two routes is selected. These constraints are the classical constraints used in ILP formulations for the *Maximum-Clique Problem* (MCP), see, e.g., [30].

The second class of clique constraints, called *anti-neighborhood constraints*, was first introduced by [17] for the MCP and reads as follows:

$$\sum_{s \in \tilde{\mathcal{N}}(r)} x_s \leq |\tilde{\mathcal{N}}(r)|\, (1 - x_r), \qquad r \in \mathcal{V}. \tag{11}$$

For each vertex $r \in \mathcal{V}$, constraints (11) ensure that if the vertex is selected, no vertex in its anti-neighborhood can be chosen; conversely, if the vertex is not selected, at most the $|\tilde{\mathcal{N}}(r)|$ vertices from its anti-neighborhood can be chosen. Since constraints (9) require selecting exactly one vertex per layer, at most $k-1$ vertices can be chosen from the anti-neighborhood of any vertex $r \in \mathcal{V}$. Consequently, the value $|\tilde{\mathcal{N}}(r)|$ on the right-hand side of constraints (11) can be reduced to $\min\{|\tilde{\mathcal{N}}(r)|, k-1\}$. Moreover, if for a vertex $r \in \mathcal{V}$, the clique number $\omega\big(\mathcal{G}(\tilde{\mathcal{N}}(r), \mathcal{E}[\tilde{\mathcal{N}}(r)])\big)$ of the graph induced by the anti-neighborhood of the vertex is smaller than $\min\{|\tilde{\mathcal{N}}(r)|, k-1\}$, the value $|\tilde{\mathcal{N}}(r)|$ can be further reduced to $\omega\big(\mathcal{G}(\tilde{\mathcal{N}}(r), \mathcal{E}[\tilde{\mathcal{N}}(r)])\big)$. This is valid because the clique number represents the maximum number of vertices that can be simultaneously selected in the anti-neighborhood. Since computing the clique number can be computationally challenging, this reduction can also be applied using any upper bound on the clique number (e.g., $\min\{|\tilde{\mathcal{N}}(r)|, k-1\}$) that satisfies the required condition.

We denote by Q the BQP formulation (8), which includes the assignment and clique constraints. The formulation with the two alternative classes of clique constraints is formally defined as follows.

Definition 2. *Let Q_a and Q_b be the BQP formulation* (8) *with clique constraints* (10) *and* (11)*, respectively.*

3.2. Solution approaches for BQP formulations

Formulation (8) is a BQP model [25, 31] with linear constraints, specifically the assignment and clique constraints. Such BQP formulations can be solved using commercial mixed-integer quadratic programming (MIQP) solvers, such as IBM ILOG CPLEX and Gurobi. These exact solvers handle BQP formulations in two main ways: *(i)* extending the standard branch-and-cut framework for MILP formulations and solving convex relaxations via a generalization of the simplex method or the barrier algorithm for quadratic programming, or *(ii)* linearizing the quadratic terms in the objective function and then applying the standard branch-and-cut framework for MILP formulations. For further details on MIQP solvers, we refer the interested readers to [5, 6].

Although the objective function of formulation (8) is generally non-convex, convexity can be reached since the model involves only binary variables. To achieve this, various approaches exist, such as the Quadratic Convex Reformulation (QCR) method introduced by Billionnet et al. [4]. Commercial MIQP solvers often employ a preprocessing step to convexify the objective function before resorting to the branch-and-cut phase [5]. It is also worth noting that recent advances have been made in the automatic selection of solution strategies. In particular, Bonami et al. [7] present a classifier designed to determine when the objective function linearization should be applied, a feature integrated into CPLEX starting from version 12.10.0.

Another common approach to solving a BQP formulation in the literature is to apply a problem-specific linearization of the quadratic objective function, leading to a MILP formulation of the underlying problem. Various linearization techniques for BQPs have been developed, differing in the number of auxiliary variables and constraints they introduce. For a comprehensive survey of these approaches, we refer to Furini and Traversi [20].

4. A new MILP Formulation for the TSRSP

In this Section, we build upon the BQP formulation (8) and introduce a novel and effective MILP formulation for the TSRSP, specifically designed by adapting the linearization technique proposed by Glover [21] to the unique structure of the problem. The resulting model incorporates a linear number of additional non-negative variables and constraints. Crucially, it leverages the k-partite structure of the TSRSP's compatibility graph, leading to a strong LP relaxation and very good computational performance, as discussed in Section 7. We refer to this linearization as the *Glover linearization technique* throughout the remainder of the paper.

4.1. The Glover Linearization Technique

Following Glover [21], we introduce an additional set of continuous variables, denoted by $\boldsymbol{w} \in \mathbb{R}^{|\mathcal{V}|}$, where each variable is associated with a vertex $r \in \mathcal{V}$ in the compatibility graph $\mathcal{G}$. Variable w_r represents the total pairing cost of incident edges when vertex r is selected as part of a k-clique in $\mathcal{G}$, and takes the value zero otherwise. Formally:

$$w_r = \begin{cases} \sum_{s \in \mathcal{N}(r)} c_{rs}\, x_s & \text{if } x_r = 1, \\ 0 & \text{if } x_r = 0, \end{cases} \qquad r \in \mathcal{V}.$$

Since all pairing costs c_{rs} are nonnegative by definition, variables $\boldsymbol{w}$ are also nonnegative. Let $\underline{L}_1^r$ and $\overline{L}_1^r$ denote valid lower and upper bounds, respectively, on the value of w_r when vertex r is selected, i.e., when $x_r = 1$. These values must then satisfy the following inequality:

$$\underline{L}_1^r\, x_r \leq \sum_{s \in \mathcal{N}(r)} c_{rs}\, x_s\, x_r \leq \overline{L}_1^r\, x_r.$$

The MILP formulation resulting from the Glover linearization technique of the BQP formulation (8) for the TSRSP reads as follows:

$$\underset{\substack{\boldsymbol{x} \in \{0,1\}^{|\mathcal{V}|} \\ \boldsymbol{w} \geq \boldsymbol{0}}}{\text{minimize}} \quad \sum_{r \in \mathcal{V}} \left(p_r\, x_r + \frac{1}{2}\, w_r \right) \tag{12a}$$

$$\text{subject to} \quad w_r \geq \underline{L}_1^r\, x_r, \qquad r \in \mathcal{V}, \tag{12b}$$

$$w_r \geq \sum_{s \in \mathcal{N}(r)} c_{rs}\, x_s - \overline{L}_1^r\, (1 - x_r), \quad r \in \mathcal{V}, \tag{12c}$$

$$\boldsymbol{x} \text{ is a } k\text{-clique in } \mathcal{G}. \tag{12d}$$

The linear objective function (12a) sums the route and pairing costs associated with the selected k-clique in the compatibility graph $\mathcal{G}$. The coefficient $\frac{1}{2}$ is introduced to prevent the double counting of pairing costs, as each edge contributes to the total cost through both of its endpoints. For each $r \in \mathcal{V}$, constraints (12b) ensure that if r is selected (i.e., $x_r = 1$), then w_r must be at least $\underline{L}_1^r$. If r is not selected, the constraint enforces nonnegativity of w_r. While these constraints are redundant for integer solutions, they are important in improving the quality of the LP relaxation by tightening the feasible region and raising the lower bound. Finally, for each $r \in \mathcal{V}$, the associated linear constraint (12c) ensures that variable w_r is at least as large as the total pairing cost incurred by vertex r when it is part of the selected k-clique. If vertex r is not selected, i.e., $x_r = 0$, the constraint becomes inactive due to the presence of the value $\overline{L}_1^r$, which acts as a so-called big-M value. When properly calibrated, such big-M values are known to significantly enhance the strength of the LP relaxation [1, 20].

Following the approach proposed by Glover [21], for each $r \in \mathcal{V}$, valid $\underline{L}_1^r$ and $\overline{L}_1^r$ values can be computed as follows:

$$\underline{L}_1^r = \underbrace{\sum_{s \in \mathcal{N}(r)} \min\{0, c_{rs}\}}_{=D_r^-} = 0, \tag{13a}$$

$$\overline{L}_1^r = \underbrace{\sum_{s \in \mathcal{N}(r)} \max\{0, c_{rs}\}}_{=D_r^+} = \sum_{s \in \mathcal{N}(r)} c_{rs}. \tag{13b}$$

The expression in (13a) (denoted D_r^-) is valid because all pairing costs are nonnegative by definition. The expression in (13b) (denoted D_r^+) is valid, as it sums the pairing costs of all edges incident to vertex r in the compatibility graph $\mathcal{G}$. These values are called the *standard values.*

We denote by G the MILP formulation (12), which includes the assignment and clique constraints. The formulation with the two alternative classes of clique constraints and the values defined in (13a) and (13b) is formally defined as follows.

Definition 3. *Let G_a and G_b be the MILP formulation* (12) *with values D_r^-* (13a) *and D_r^+* (13b) *for each $r \in \mathcal{V}$, and with clique constraints* (10) *and* (11)*, respectively.*

It is worth noting that the Glover linearization technique includes additional constraints, which are not necessary in formulation (12). These constraints are redundant in the minimization context since all pairing costs are nonnegative by definition (see also Furini and Traversi [20] for further details).

4.2. Strengthening the LP relaxation of the G formulation

In this section, we explore how to strengthen the LP relaxation of the G formulation (12) by refining the values $\underline{L}_1^r$ and $\overline{L}_1^r$ for each $r \in \mathcal{V}$. Our strategy leverages the combinatorial structure induced by constraints (12d), which impose selecting a k-clique in the compatibility graph $\mathcal{G}$. We propose two approaches: the first, referred to as the *clique-based values*, computes the tightest possible values by solving auxiliary optimization problems. The second, which we call the *layer-based values*, provides strong values computable in linear time with respect to the number of edges of the compatibility graph. From the computational results (see Section 7), the second approach provides very strong values in negligible time, while the first one is very time-consuming and is used to benchmark the values of the second approach.

4.2.1. Clique-Based Values

In line with [1], the tightest values $\underline{L}_1^r$ and $\overline{L}_1^r$ for each $r \in \mathcal{V}$ can be computed by exploiting the structure of the feasible region of the problem. To this end, given a vertex $r \in \mathcal{V}$, let $\tilde{\mathcal{X}}(r)$ be the collection of all incidence vectors $\boldsymbol{x} \in \{0,1\}^{|\mathcal{N}(r)|}$ that represent $(k-1)$-cliques in the subgraph $\mathcal{G}[\mathcal{N}(r)]$ induced by the vertices in the neighborhood $\mathcal{N}(r)$ of the vertex r, formally:

$$\tilde{\mathcal{X}}(r) = \left\{ \boldsymbol{x} \in \{0,1\}^{|\mathcal{N}(r)|} : \quad \boldsymbol{x} \text{ is a } (k-1)\text{-clique in } \mathcal{G}[\mathcal{N}(r)] \right\}.$$

For every $r \in \mathcal{V}$, the tightest values $\underline{L}_1^r$ and $\overline{L}_1^r$, denoted as $\tilde{D}_r^-$ and $\tilde{D}_r^+$, can be then calculated as follows:

$$\underline{L}_1^r = \underbrace{\min_{\boldsymbol{x} \in \tilde{\mathcal{X}}(r)} \left\{ \sum_{s \in \mathcal{N}(r)} c_{rs}\, x_s \right\}}_{=\tilde{D}_r^-}, \tag{14a}$$

$$\overline{L}_1^r = \underbrace{\max_{\boldsymbol{x} \in \tilde{\mathcal{X}}(r)} \left\{ \sum_{s \in \mathcal{N}(r)} c_{rs}\, x_s \right\}}_{=\tilde{D}_r^+}. \tag{14b}$$

Given a vertex $r \in \mathcal{V}$, in the objective function of (14a) and (14b), every vertex $s \in \mathcal{N}(r)$ has a weight equal to the pairing cost c_{rs}. By construction, all these vertex weights are nonnegative, and the subgraph $\mathcal{G}[\mathcal{N}(r)]$ is $(k-1)$-partite. Specifically, problem (14a) is a *vertex-weighted minimum* $(k-1)$*-clique problem* in the $(k-1)$-partite subgraph $\mathcal{G}[\mathcal{N}(r)]$. It is associated with the case where vertex r is selected, and the remaining $k-1$ vertices of the clique must lie in its neighborhood. Conversely, problem (14b) is a *vertex-weighted maximum* $(k-1)$*-clique problem* in the $(k-1)$-partite subgraph $\mathcal{G}[\mathcal{N}(r)]$.

Problem (14a) and problem (14b) are $\mathcal{NP}$-hard, since determining whether a $(k-1)$-partite graph contains a $(k-1)$-clique is $\mathcal{NP}$-complete [42]. Consequently, computing the tightest values $\tilde{D}_r^-$ and $\tilde{D}_r^+$ requires solving $2 \cdot |\mathcal{V}|$ $\mathcal{NP}$-hard problems, which can be very time-consuming, especially for large graphs.

The G formulation (12) with values defined in (14a) and (14b) is formally defined as follows.

Definition 4. *Let $\tilde{G}_a$ and $\tilde{G}_b$ be the MILP formulation* (12) *with values $\tilde{D}_r^-$* (14a) *and $\tilde{D}_r^+$* (14b) *for each $r \in \mathcal{V}$, and with clique constraints* (10) *and* (11)*, respectively.*

It is worth noticing that there is a one-to-one correspondence between the optimal solutions of problem (14b) and the ones of the *Vertex-Weighted Maximum Clique Problem* (VWMCP) on the subgraph $\mathcal{G}[\mathcal{N}(r)]$ with vertex weights defined by the pairing costs. Moreover, the optimal value of the two problems coincides and determines the value $\tilde{D}_r^+$. Several exact algorithms have been proposed for the VWMCP in the literature; among the most effective ones, there is the combinatorial branch-and-bound method described in [41], which we use in the computational experiments to determine the value $\tilde{D}_r^+$ for each vertex $r \in \mathcal{V}$ in formulations $\tilde{G}_a$ and $\tilde{G}_b$.

The following proposition states that the optimal value of problem (14a) is equal to a constant term minus the optimal value of problem (14b) with *ad hoc* nonnegative vertex weights. This result holds since the clique size in both problems (14a) and (14b) is fixed and equal to $k-1$. Moreover, this proposition allows us to use the same exact algorithm for the VWMCP when solving also problem (14a) to calculate $\tilde{D}_r^-$.

Proposition 1. *For every $r \in \mathcal{V}$, we have:*

$$\min_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ \sum_{s\in\mathcal{N}(r)} c_{rs}\, x_s \right\} = (k-1) \max_{t\in\mathcal{N}(r)}\{c_{rt}\} - \max_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ \sum_{s\in\mathcal{N}(r)} \underbrace{\left(\max_{t\in\mathcal{N}(r)}\{c_{rt}\} - c_{rs} \right)}_{\geq 0} x_s \right\}$$

Proof. By definition, for every $r \in \mathcal{V}$, we have $\max_{t\in\mathcal{N}(r)}\{c_{rt}\} \geq c_{rs}$ for every $s \in \mathcal{N}(r)$. Since all pairing costs are nonnegative, it follows that

$$\max_{t\in\mathcal{N}(r)}\{c_{rt}\} - c_{rs} \geq 0.$$

Moreover, by definition, any vector $\boldsymbol{x} \in \tilde{\mathcal{X}}(r)$ satisfies $\sum_{s\in\mathcal{N}(r)} x_s = k-1$; we have:

$$(k-1) \max_{t\in\mathcal{N}(r)}\{c_{rt}\} = \sum_{s\in\mathcal{N}(r)} \max_{t\in\mathcal{N}(r)}\{c_{rt}\}\, x_s.$$

Hence, we can write:

$$\begin{aligned}
\max_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ \sum_{s\in\mathcal{N}(r)} \left(\max_{t\in\mathcal{N}(r)}\{c_{rt}\} - c_{rs} \right) x_s \right\} &= \max_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ \sum_{s\in\mathcal{N}(r)} \left(\max_{t\in\mathcal{N}(r)}\{c_{rt}\}\, x_s - c_{rs}\, x_s \right) \right\} \\
&= \max_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ (k-1) \max_{t\in\mathcal{N}(r)}\{c_{rt}\} - \sum_{s\in\mathcal{N}(r)} c_{rs}\, x_s \right\} \\
&= (k-1) \max_{t\in\mathcal{N}(r)}\{c_{rt}\} + \max_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ - \sum_{s\in\mathcal{N}(r)} c_{rs}\, x_s \right\} \\
&= (k-1) \max_{t\in\mathcal{N}(r)}\{c_{rt}\} - \min_{\boldsymbol{x}\in\tilde{\mathcal{X}}(r)} \left\{ \sum_{s\in\mathcal{N}(r)} c_{rs}\, x_s \right\},
\end{aligned}$$

and, accordingly, the desired identity follows. □

4.2.2. Layer-Based Values

Strong values $\underline{L}_1^r$ and $\overline{L}_1^r$ for each $r \in \mathcal{V}$ can be computed in a very small computational effort using the following approach. Given a layer $i \in \mathcal{K}$ and a vertex $r \in \mathcal{V}_i$, let $\hat{\mathcal{X}}(r)$ be the collection of all incidence vectors $\boldsymbol{x} \in \{0,1\}^{|\mathcal{N}(r)|}$ that satisfy the assignment constraints (9) in the subgraph $\mathcal{G}[\mathcal{N}(r)]$ induced by the vertices in the neighborhood $\mathcal{N}(r)$, formally:

$$\hat{\mathcal{X}}(r) = \left\{ \boldsymbol{x} \in \{0,1\}^{|\mathcal{N}(r)|} : \sum_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} x_s = 1, \;\; j \in \mathcal{K} \setminus \{i\} \right\}.$$

The $k-1$ constraints defining the collection $\hat{\mathcal{X}}(r)$ impose that exactly one vertex from the neighbors of vertex r is selected in each layer except the layer i of the vertex r. By nature, for each $i \in \mathcal{K}$ and $r \in \mathcal{V}$, we have $\tilde{\mathcal{X}}(r) \subseteq \hat{\mathcal{X}}(r)$, that is, $\hat{\mathcal{X}}(r)$ contains all the $(k-1)$-cliques in $\tilde{\mathcal{X}}(r)$, but also the sets of $k-1$ vertices (one per layer in the neighborhood of r) not pairwise adjacent.

For each $i \in \mathcal{K}$ and $r \in \mathcal{V}_i$, valid values $\underline{L}_1^r$ and $\overline{L}_1^r$, denoted $\hat{D}_r^-$ and $\hat{D}_r^+$, can be calculated as follows:

$$\underline{L}_1^r = \underbrace{\min_{\boldsymbol{x} \in \hat{\mathcal{X}}(r)} \left\{ \sum_{s \in \mathcal{N}(r)} c_{rs} x_s \right\}}_{=\hat{D}_r^-}, \tag{15a}$$

$$\overline{L}_1^r = \underbrace{\max_{\boldsymbol{x} \in \hat{\mathcal{X}}(r)} \left\{ \sum_{s \in \mathcal{N}(r)} c_{rs} x_s \right\}}_{=\hat{D}_r^+}. \tag{15b}$$

For each vertex in a given layer, problem (15a) aims at selecting a neighbor vertex with the smallest pairing cost from each of the other layers. Similarly, problem (15b) aims at selecting a neighbor vertex with the largest pairing cost from each of the other layers.

The G formulation (12) with values defined in (15a) and (15b) is formally defined as follows.

Definition 5. *Let $\hat{G}_a$ and $\hat{G}_b$ be the MILP formulation* (12) *with values $\hat{D}_r^-$* (15a) *and $\hat{D}_r^+$* (15b) *for each $r \in \mathcal{V}$, and with clique constraints* (10) *and* (11)*, respectively.*

The following proposition provides closed formulas to compute the layer-based values.

Proposition 2. *For every layer $i \in \mathcal{K}$ and every vertex $r \in \mathcal{V}_i$, we have:*

$$\hat{D}_r^- = \sum_{j \in \mathcal{K} \setminus \{i\}} \min_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} \{c_{rs}\}, \tag{16a}$$

$$\hat{D}_r^+ = \sum_{j \in \mathcal{K} \setminus \{i\}} \max_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} \{c_{rs}\}. \tag{16b}$$

Proof. By construction, given a layer $i \in \mathcal{K}$ and a vertex $r \in \mathcal{V}_i$, the objective function of problems (15a) and (15b) can be rewritten as:

$$\sum_{s \in \mathcal{N}(r)} c_{rs} x_s = \sum_{j \in \mathcal{K} \setminus \{i\}} \sum_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} c_{rs} x_s.$$

Due to the assignment constraints (9), exactly one binary variable must be equal to one in each inner sum. To minimize the objective function as in problem (15a), for each $j \in \mathcal{K} \setminus \{i\}$, the vertex $s \in \mathcal{N}(r) \cap \mathcal{V}_j$ with the smallest pairing cost c_{rs} must be selected. Hence, we have:

$$\hat{D}_r^- = \min_{\boldsymbol{x} \in \hat{\mathcal{X}}(r)} \left\{ \sum_{j \in \mathcal{K} \setminus \{i\}} \sum_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} c_{rs} x_s \right\} = \sum_{j \in \mathcal{K} \setminus \{i\}} \min_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} \{c_{rs}\}.$$

Analogously, to maximize the objective function as in Problem (15b), for each $j \in \mathcal{K} \setminus \{i\}$, the vertex $s \in \mathcal{N}(r) \cap \mathcal{V}_j$ with the largest pairing cost c_{rs} must be selected. Hence, we have:

$$\hat{D}_r^+ = \max_{\boldsymbol{x} \in \hat{\mathcal{X}}(r)} \left\{ \sum_{j \in \mathcal{K} \setminus \{i\}} \sum_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} c_{rs} x_s \right\} = \sum_{j \in \mathcal{K} \setminus \{i\}} \max_{s \in \mathcal{N}(r) \cap \mathcal{V}_j} \{c_{rs}\}.$$

$\square$

Thanks to the previous proposition, the layer-based values can be computed in linear time with respect to the edges of the compatibility graph, i.e., in $\mathcal{O}(|\mathcal{E}|)$. Accordingly, the computation of the layer-based values can be performed much more efficiently than the computation of the clique-based values.

4.2.3. Comparison between the two types of values

As previously discussed, the quality of the clique-based and the layer-based values directly influences the strength of the lower bound obtained from the LP relaxation of the G formulation (12). Our computational experiments (see Section 7) indicate that, for the considered test bed of real-world instances, the layer-based values are consistently close to the clique-based values, and the lower bounds provided by the LP relaxation are of comparable quality.

Moreover, it is worth noting that if problems (15a) and (15b) are both infeasible for a vertex $r \in \mathcal{V}$, then the corresponding variable x_r can be set to zero. These are the vertices that lack at least one neighbor in every layer other than the layer of vertex r. Similarly, if problems (14a) and (14b) are both infeasible, then the corresponding variable x_r can also be set to zero. These are the vertices whose neighborhood $\mathcal{N}(r)$ does not contain a clique of size $k-1$. In addition, if all variables associated with the vertices in at least one layer are set to zero, then the entire TSRSP instance becomes infeasible.

Finally, in the special case where the compatibility graph $\mathcal{G}$ is *layer-wise* complete, the layer-based and clique-based values coincide. This result follows from the fact that, in a layer-wise complete compatibility graph, for every vertex $r \in \mathcal{V}$, any subset selecting one vertex per layer forms a $(k-1)$-clique in the subgraph $\mathcal{G}[\mathcal{N}(r)]$ induced by the neighborhood of vertex r. As a consequence, the clique constraints in the problems (14a) and (14b) are redundant, and the sets $\hat{\mathcal{X}}(r)$ and $\tilde{\mathcal{X}}(r)$ coincide.

We conclude this section analyzing the TSRSP instance presented in Section 2.3. Table 1 reports the standard values (13) from [21], the layer-based values (15), and the clique-based values (14) for all vertices of the compatibility graph. All D_r^- values are zero, while

$D_r^+ \in [0, 35]$ with an average of $\frac{166}{9}$. For the layer-based and clique-based values, we have $\hat{D}_r^-, \tilde{D}_r^- \in [4, 13]$, with averages $\frac{71}{9}$ and 8, respectively; and $\hat{D}_r^+, \tilde{D}_r^+ \in [4, 17]$, with averages $\frac{113}{9}$ and $\frac{110}{9}$, respectively. In this example, they differ only at vertices 5 and 8, highlighted in bold in table 1, since the optimal solutions of problems (15a) and (15b) do not form 2-cliques for these two vertices. Overall, the table shows that the layer-based and clique-based values are very close for this TSRSP instance.

Table 1: Comparison between the clique-based values, the layer-based values, and the standard values from [21] for all the vertices of the compatibility graph of the TSRSP instance shown in figure 1.

Vertex r	*standard values [21]*		*layer-based values*		*clique-based values*	
	D_r^- (13a)	D_r^+ (13b)	$\hat{D}_r^-$(15a)	$\hat{D}_r^+$ (15b)	$\tilde{D}_r^-$ (14a)	$\tilde{D}_r^+$ (14b)
1	0	16	10	15	10	15
2	0	17	5	12	5	12
3	0	20	13	15	13	15
4	0	4	4	4	4	4
5	0	20	7	13	7	**10**
6	0	24	8	16	8	16
7	0	8	8	8	8	8
8	0	22	5	13	**6**	13
9	0	35	11	17	11	17

The optimal solution value for the TSRSP instance shown in figure 1 is 16. The optimal solution values of the LP relaxations for the G_a and G_b formulations are $\frac{4563}{556}$ and $\frac{2223}{271}$, respectively (computed in rational arithmetic using `SCIP`), whereas those for the $\hat{G}_a$, $\hat{G}_b$, $\tilde{G}_a$, and $\tilde{G}_b$ formulations are all equal to 16. Accordingly, the LP relaxation gap for G_a and G_b is approximately 48.7%, while it drops to 0% for the other formulations. This result highlights the substantial improvement in relaxation quality achieved by incorporating the layer-based and clique-based values in this TSRSP instance.

5. Other formulations for the TSRSP

In this section, we introduce two additional formulations for the TSRSP used for benchmarking purposes. The first one is a MILP formulation obtained through the classical linearization method proposed by Glover and Woolsey [22] for the Q formulation (8), which involves a quadratic number of additional variables and constraints (see also [9, 38]). We refer to this linearization as the *Glover-Woolsey linearization technique* throughout the remainder of the paper. The second one is the only ILP formulation for the TSRSP available in the literature, proposed by Samà et al. [40].

5.1. The Glover-Woolsey linearization technique

Following Glover and Woolsey [22], we introduce an additional set of continuous non-negative variables, denoted by $\boldsymbol{z} \in \mathbb{R}_+^{|\mathcal{E}|}$, where each variable is associated with an edge $e = \{r, s\} \in \mathcal{E}$ in the compatibility graph $\mathcal{G}$. Variable z_e models the product of variables x_r and x_s.

The MILP formulation resulting from the Glover-Woolsey linearization technique of the BQP formulation (8) for the TSRSP reads as follows:

$$\underset{\substack{\boldsymbol{x}\in\{0,1\}^{|\mathcal{V}|}\\ \boldsymbol{z}\geq\boldsymbol{0}}}{\text{minimize}} \quad \sum_{r\in\mathcal{V}} p_r\, x_r + \sum_{e\in\mathcal{E}} c_e\, z_e \tag{17a}$$

$$\text{subject to} \quad x_r + x_s - z_e \leq 1, \quad e=\{r,s\}\in\mathcal{E}, \tag{17b}$$

$$\boldsymbol{x} \text{ is a } k\text{-clique in } \mathcal{G}\,. \tag{17c}$$

The quadratic objective function in the Q formulation (8) is replaced by the linear objective function (17a), which is composed by the sum of two linear terms: one associated with vertex variables and the other with edge variables. Constraints (17b) and binary variables $\boldsymbol{x}$ ensure that if both endpoints r and s of the edge e belong to the selected clique, then variable $z_e = 1$. Otherwise, due to the minimization of the objective function, z_e takes the value zero.

We denote by GW the MILP formulation (17), which includes the assignment and clique constraints. The formulation with the two alternative classes of clique constraints is formally defined as follows.

Definition 6. *Let GW_a and GW_b be the MILP formulation* (17) *with clique constraints* (10) *and* (11)*, respectively.*

It is worth noting that the Glover-Woolsey linearization technique includes additional constraints that are not required in the GW formulation (17). These constraints are redundant in a minimization form, as all pairing costs are nonnegative by definition (see also Furini and Traversi [20] for further discussion).

Our computational results (see Section 7) demonstrate that when the standard values (13) proposed by Glover [21] are employed, the LP relaxation of the G formulation (12) provides slightly weaker lower bounds than those obtained from the LP relaxation of the GW formulation (17). However, this is no longer the case when the clique-based values (14) or the layer-based values (15) are used in the G formulation (12).

5.2. An ILP formulation from the literature

To the best of our knowledge, the only ILP formulation for the TSRSP available in the literature is the formulation proposed by Samà et al. [40]. It employs two sets of binary variables. The first set consists of variables $\boldsymbol{x} \in \{0,1\}^{|\mathcal{V}|}$, which are also used in the formulations introduced in this paper. The second set consists of variables $\boldsymbol{y} \in \{0,1\}^{|\mathcal{E}|}$, with one binary variable for each edge $e = \{r,s\} \in \mathcal{E}$ of the compatibility graph $\mathcal{G}$. Specifically, variable y_e takes the value 1 if and only if both its endpoints are selected in the solution, and 0 otherwise.

The ILP formulation of [40] for the TSRSP reads as follows:

$$\begin{aligned}
\underset{\substack{\boldsymbol{x}\in\{0,1\}^{|\mathcal{V}|}\\ \boldsymbol{y}\in\{0,1\}^{|\mathcal{E}|}}}{\text{minimize}} \quad & \sum_{r\in\mathcal{V}} p_r\, x_r + \sum_{e\in\mathcal{E}} c_e\, y_e && \text{(18a)}\\
\text{subject to} \quad & \sum_{r\in\mathcal{V}_i} x_r = 1, \qquad i\in\mathcal{K}, && \text{(18b)}\\
& \sum_{e\in\delta(r)} y_e = (k-1)\, x_r, \quad r\in\mathcal{V}, && \text{(18c)}
\end{aligned}$$

where, for each $r \in \mathcal{V}$, $\delta(r) \subseteq \mathcal{E}$ denotes the set of edges incident to r. The linear objective function (18a) minimizes the cost of the k-clique selected in the compatibility graph. Constraints (18b) are the assignment constraints, also used by all other formulations presented in this paper. Constraints (18c) are derived from the *star inequalities* [32, 28, 26]; they ensure that each selected vertex is incident to exactly $k-1$ edges in any feasible solution. Since an edge can be included only if both its endpoints are selected, and the assignment constraints (18b) ensure that exactly one vertex is chosen per layer, these conditions are necessary and sufficient to construct a k-clique in the compatibility graph.

We denote by S the ILP formulation (18) which is formally defined as follows.

Definition 7. *Let S be the ILP formulation* (18).

Our computational results (see Section 7) indicate that, for the test bed of real-world instances, the LP relaxation of the S formulation (18) may yield poor-quality lower bounds.

6. Theoretical comparison of the MILP formulations

In this section, we provide a theoretical comparison of the MILP formulations for the TSRSP studied in this paper. The analysis focuses on the number of variables and constraints of the formulations, and the strength of their LP relaxations. This theoretical investigation is then complemented by the computational tests in Section 7.

6.1. Comparison between the number of variables and constraints

Table 2 presents the main features of the MILP formulations studied in this paper. The first column reports the name of each formulation, while the second column indicates the section in which it is presented. For each formulation, the columns labeled “clique constraints” specify whether the non-edge constraints (10) or the anti-neighborhood constraints (11) are used (the S formulation does not include any clique constraints). Finally, the last three columns report the number of binary variables, continuous variables, and constraints.

As far as the number of variables is concerned, all formulations use $|\mathcal{V}|$ binary variables associated with the vertices of the compatibility graph. In addition, formulation GW includes $|\mathcal{E}|$ continuous variables, while formulation S includes $|\mathcal{E}|$ binary variables, both associated with the edges. Formulation G introduces an additional set of $|\mathcal{V}|$ continuous variables, also associated with the vertices. Among the MILP formulations, formulation G has the smallest number of variables, depending only on the number of vertices. In contrast, the other MILP formulations include a number of variables that depend on the number of edges; hence, dense

Table 2: Main features of the MILP formulations for the TSRSP problem studied in this paper.

		Clique Constraints				
Form.	Section	Constr. (10)	Constr. (11)	# Bin.Var.	# Cont.Var.	# Constr.
G_a	§4	✓		$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}\rvert$	$2\lvert\mathcal{V}\rvert + \lvert\mathcal{K}\rvert + \lvert\tilde{\mathcal{E}}\rvert$
G_b			✓	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}\rvert$	$3\lvert\mathcal{V}\rvert + \lvert\mathcal{K}\rvert$
GW_a	§5.1	✓		$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{E}\rvert + \lvert\mathcal{K}\rvert + \lvert\tilde{\mathcal{E}}\rvert$
GW_b			✓	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{E}\rvert + \lvert\mathcal{K}\rvert + \lvert\mathcal{V}\rvert$
S	§5.2			$\lvert\mathcal{V}\rvert + \lvert\mathcal{E}\rvert$		$\lvert\mathcal{K}\rvert + \lvert\mathcal{V}\rvert$

compatibility graphs result in a larger number of variables. It is worth noting that the number of additional continuous variables in formulation GW is $O(|\mathcal{V}|^2)$; that is, it grows at most quadratically with the number of vertices $|\mathcal{V}|$. By contrast, the number of additional continuous variables in formulation G grows linearly with the number of vertices.

As far as the clique constraints are concerned, formulations using the non-edge constraints (10) include $|\tilde{\mathcal{E}}|$ additional constraints, while those based on the anti-neighborhood constraints (11) include $|\mathcal{V}|$ additional constraints. Accordingly, the number of constraints in the latter case does not depend on the density of the compatibility graph. It is worth noting that the number of non-edge constraints (10) is $O(|\mathcal{V}|^2)$; that is, it grows quadratically with the number of vertices $|\mathcal{V}|$, whereas the number of anti-neighborhood constraints (11) grows linearly. Therefore, the cardinality of constraints (10) is greater than that of constraints (11), but the former may provide a stronger polyhedral description of the feasible region (see Section 6.2). Formulation G has $2|\mathcal{V}|$ additional constraints, formulation GW has $|\mathcal{E}|$ additional constraints, and formulation S includes a total of $|\mathcal{K}| + |\mathcal{V}|$ constraints due to the linearization techniques. Among these, formulation S has the smallest number of constraints.

Finally, it is worth noting that the only formulation whose number of variables and constraints does not depend on the number of edges in the compatibility graph is formulation G_b.

6.2. Comparison of the strength of the LP relaxations

In this section, we compare the strength of the LP relaxation of the MILP formulations studied in this paper. For a given formulation F, let $LP(F)$ denote the optimal solution value of its LP relaxation. For two formulations, F_1 and F_2, we write $F_1 \rightarrow F_2$ to indicate that $LP(F_1) \geq LP(F_2)$, meaning that the value of the lower bound provided by the LP relaxation of F_1 is greater than or equal to the one provided by the LP relaxation of F_2, for any given instance. In such cases, we say that F_1 dominates F_2. We say that F_1 strictly dominates F_2 if F_1 dominates F_2 and there exist instances for which $LP(F_1) > LP(F_2)$. The optimal solutions and their corresponding objective function values reported in the following proofs are computed using rational arithmetic in the `SCIP` solver, ensuring numerical exactness.

The following proposition states that the formulations using the non-edge constraints (10) dominate the ones using the anti-neighborhood constraints (11).

Proposition 3. *We have:*

$$GW_a \to GW_b, \quad G_a \to G_b, \quad \hat{G}_a \to \hat{G}_b \quad \textit{and} \quad \tilde{G}_a \to \tilde{G}_b\,.$$

Moreover, strict domination holds.

Proof. The anti-neighborhood constraints (11) can be obtained by summing, for each vertex $r \in \mathcal{V}$, the non-edge constraints (10) over all vertices $s \in \tilde{\mathcal{N}}(r)$. In this sense, the former represent a surrogate relaxation of the latter. Therefore, the formulations using the non-edge constraints (10) dominate those using the anti-neighborhood constraints (11).

Consider now the TSRSP instance depicted in figure 1. The optimal solution values of the LP relaxations of formulations G_a and G_b are $\frac{4563}{556}$ and $\frac{2223}{271}$, respectively. Therefore, for this instance, we have $LP(G_a) > LP(G_b)$. Examples proving the other strict dominance relationships can be found among the instances tested in the computational section of this manuscript. $\square$

The following proposition states that formulations GW_a and GW_b dominate formulations G_a and G_b, respectively. The scheme of the proof is first to show that any feasible solution of the LP relaxation of GW_a can be mapped into a feasible solution for the LP relaxation of G_a with the same objective value. Then, exhibit an instance for which $LP(GW_a) > LP(G_a)$. The same reasoning holds for formulations GW_b and G_b.

Proposition 4. *We have:*

$$GW_a \to G_a \quad \textit{and} \quad GW_b \to G_b\,.$$

Moreover, strict domination holds.

Proof. Let $(\boldsymbol{x}, \boldsymbol{z}) \in \mathbb{R}_+^{|\mathcal{V}|+|\mathcal{E}|}$ be a feasible solution to the LP relaxation of GW_a. For every $r \in \mathcal{V}$, we define:

$$w_r = \sum_{s \in \mathcal{N}(r)} c_{rs}\, z_{rs}.$$

By definition, $(\boldsymbol{x}, \boldsymbol{w}) \in \mathbb{R}_+^{2|\mathcal{V}|}$ satisfies constraints (12d). Moreover, always by definition, for every $e = \{r, s\} \in \mathcal{E}$ we have $z_e \geq 0$ and, for every $r \in \mathcal{V}$ we have $D_r^- = 0$. It follows that $(\boldsymbol{x}, \boldsymbol{w})$ satisfies constraints (12b). It also follows that $w_r \geq 0$ for every $r \in \mathcal{V}$.

Finally, for every $r \in \mathcal{V}$, we have:

$$\begin{aligned} w_r &= \sum_{s \in \mathcal{N}(r)} c_{rs}\, z_{rs} \\ &\geq \sum_{s \in \mathcal{N}(r)} c_{rs}\, (x_r + x_s - 1) \qquad (\text{ by constraints (17b) }) \end{aligned}$$

$$= \sum_{s\in\mathcal{N}(r)} c_{rs}\, x_s - (1-x_r) \underbrace{\sum_{s\in\mathcal{N}(r)} c_{rs}}_{=D_r^+ \text{ (13b)}} .$$

Hence, $(\boldsymbol{x},\boldsymbol{w})$ is a feasible solution for the LP relaxation of G_a and yields the same objective function value, since:

$$\sum_{r\in\mathcal{V}} p_r\, x_r + \sum_{e\in\mathcal{E}} c_e\, z_e = \sum_{r\in\mathcal{V}} p_r\, x_r + \frac{1}{2}\sum_{r\in\mathcal{V}}\sum_{s\in\mathcal{N}(r)} c_{rs}\, z_{rs} = \sum_{r\in\mathcal{V}} p_r\, x_r + \frac{1}{2}\sum_{r\in\mathcal{V}} w_r.$$

Consider now the TSRSP instance depicted in figure 1. An optimal solution to the LP relaxation of GW_a is given by $x_1 = x_2 = x_4 = x_5 = x_8 = x_9 = \frac{1}{2}$, $x_3 = x_6 = x_7 = 0$, and $z_e = 0$ for every $e \in \mathcal{E}$. The corresponding optimal objective value is $\frac{17}{2}$.

In contrast, an optimal solution to the LP relaxation of G_a is given by $x_1 = \frac{721}{1668}$, $x_2 = \frac{943}{1668}$, $x_3 = \frac{1}{417}$, $x_4 = \frac{725}{1668}$, $x_5 = \frac{293}{556}$, $x_6 = \frac{16}{417}$, $x_7 = 0$, $x_8 = \frac{339}{556}$, $x_9 = \frac{217}{556}$, and $w_r = 0$ for every $r \in \mathcal{V}$. The corresponding objective value is $\frac{4563}{556}$.

Hence, for this instance, we have $LP(GW_a) > LP(G_a)$. □

The following proposition establishes a chain of dominance relationships among formulations G when using different values of the coefficients $\underline{L}_r^1$ and $\overline{L}_r^1$ for the vertices $r \in \mathcal{V}$, i.e., the clique-based values (14), the layer-based values (15) and the standard values (13) from [21].

Proposition 5. *We have:*

$$\tilde{G}_a \to \hat{G}_a \to G_a \quad \textit{and} \quad \tilde{G}_b \to \hat{G}_b \to G_b\, .$$

Moreover, strict domination holds.

Proof. For each $i \in \mathcal{K}$ and $r \in \mathcal{V}_i$, by definition of $\hat{D}_r^-$(15a) and $\hat{D}_r^+$(15b) and since, by definition, the pairing costs are nonnegative, we have:

$$\hat{D}_r^- = \sum_{j\in\mathcal{K}\setminus\{i\}} \min_{s\in\mathcal{N}(r)\cap\mathcal{V}_j} \{c_{rs}\} \geq 0 = D_r^-\,,$$

$$\hat{D}_r^+ = \sum_{j\in\mathcal{K}\setminus\{i\}} \max_{s\in\mathcal{N}(r)\cap\mathcal{V}_j} \{c_{rs}\} \leq \sum_{s\in\mathcal{N}(r)} c_{rs} = D_r^+.$$

Moreover, always by definition, we have $\tilde{\mathcal{X}}(r) \subseteq \hat{\mathcal{X}}(r)$, which leads to the following relations between the clique-based values, layer-based values and standard values:

$$D_r^- \leq \hat{D}_r^- \leq \tilde{D}_r^- \qquad \text{and} \qquad D_r^+ \geq \hat{D}_r^+ \geq \tilde{D}_r^+.$$

Strict dominance of $\hat{G}_a$ over G_a is demonstrated by the TSRSP instance depicted in figure 1. The optimal solution values of the LP relaxations of formulations G_a and $\hat{G}_a$ are $\frac{4563}{556}$ and

16, respectively. Therefore, we have $LP(G_a) > LP(\hat{G}_a)$. Furthermore, the strict dominance of $\tilde{G}_a$ over $\hat{G}_a$ is shown by the example discussed at the end of this section, where the LP relaxation of $\hat{G}_a$ admits a feasible solution, while that of $\tilde{G}_a$ is infeasible. □

Figure 2: Dominance relationship diagram of the different variants of the G and GW formulations. An arrow from F to F' indicates that formulation F dominates formulation F'.

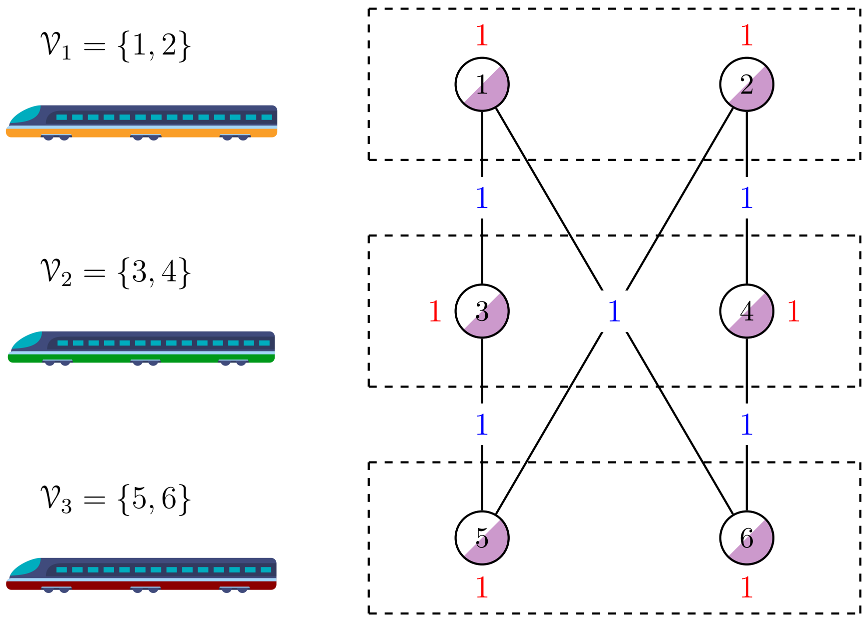
$$\begin{tikzcd}
GW_a \arrow[r] \arrow[d] & G_a \arrow[d] & \hat{G}_a \arrow[l] \arrow[d] & \tilde{G}_a \arrow[l] \arrow[d] \\
GW_b \arrow[r] & G_b & \hat{G}_b \arrow[l] & \tilde{G}_b \arrow[l]
\end{tikzcd}$$

Figure 2 summarizes the dominance relationships proven in this section between the different variants of the G and GW formulations in terms of the strength of their LP relaxation.

As far as the LP relaxation of formulation S is concerned, no general dominance relationship can be established with respect to formulations G and GW. In particular, formulation S is not dominated by GW, as there exist instances where $LP(S) > LP(GW)$. For example, in the TSRSP instance depicted in figure 1, we have $LP(S) = \frac{25}{2}$, while $LP(GW_a) = LP(GW_b) = \frac{17}{2}$. Conversely, there are other instances—such as the majority of those considered in the computational study of Section 7.2.1—for which $LP(S) < LP(GW)$. For the same reason, no clear dominance relationship can be asserted between formulation S and formulation G. For instance, in the example of figure 1, we observe $LP(S) = \frac{25}{2}$, which is strictly greater than $LP(G_a) = \frac{4563}{556}$. However, our computational experiments show that, on the tested instances, formulation $\hat{G}$ (and, by extension, $\tilde{G}$) systematically yields stronger LP relaxations than formulation S.

The following example shows that the LP relaxation of the MILP formulations considered in this manuscript may admit feasible solutions even when the TSRSP instance is infeasible. Consider the TSRSP instance depicted in figure 3, involving three trains ($k = 3$), each with two routes. The layers are defined as $\mathcal{V}_1 = \{1, 2\}$, $\mathcal{V}_2 = \{3, 4\}$, and $\mathcal{V}_3 = \{5, 6\}$. The sets of compatible route pairs are $\mathcal{E}_{12} = \{\{1, 3\}, \{2, 4\}\}$, $\mathcal{E}_{13} = \{\{1, 6\}, \{2, 5\}\}$, and $\mathcal{E}_{23} = \{\{3, 5\}, \{4, 6\}\}$, and all route and pairing costs are equal to one. Since the compatibility graph does not contain any 3-clique, the instance is infeasible. Nevertheless, the LP relaxations of formulations S, GW_a, GW_b, G_a, G_b, as well as $\hat{G}_a$ and $\hat{G}_b$, admit feasible solutions. For all these formulations, a feasible solution can be obtained by setting all variables equal to $\frac{1}{2}$. In contrast, the LP relaxations of $\tilde{G}_a$ and $\tilde{G}_b$ correctly identify the infeasibility and do not admit any feasible solution. For every vertex $r \in \mathcal{V}$, the set $\tilde{\mathcal{X}}(r)$ used in problems (14a) and (14b) is empty, as no 2-clique can be formed in the compatibility graph induced by any vertex r. However, as discussed in Section 7.2.2, computing the clique-based values can be computationally prohibitive for large instances. Therefore, the trade-off between infeasibility detection and computational cost must be carefully considered when tackling large-scale TSRSP problems.

Figure 3: An infeasible TSRSP instance with three trains, each having two available routes. The half-colored vertices represent a feasible solution to the LP relaxation of the formulations S, G_a, G_b, $\hat{G}_a$, $\hat{G}_b$, GW_a, and GW_b. In contrast, the LP relaxation of the formulations $\tilde{G}_a$ and $\tilde{G}_b$ is infeasible for this instance.



$\mathcal{V}_1 = \{1, 2\}$
$\mathcal{V}_2 = \{3, 4\}$
$\mathcal{V}_3 = \{5, 6\}$
1
2
3
4
5
6
1
1
1
1
1
1
1
1
1
1
1

7. Computational tests

In this section, we present the results of our extensive computational tests to evaluate the performance of the novel formulation G for the TSRSP (see Section 4) against the benchmark formulations discussed in the previous sections—namely, the Q, GW, and S formulations (see Sections 3 and 5). Our main objectives are threefold:

1. To evaluate the computational effectiveness of the formulations by identifying the largest instance sizes that can be solved to optimality within a short time limit suitable for the operational phases of RTM;
2. To empirically analyze the strength of the lower bounds provided by the LP relaxations of the different formulations, thereby complementing the theoretical dominance results presented in Section 6.2;
3. To empirically assess the impact on the computational performance of formulation G when using different values for the coefficients $\underline{L}_r^1$ and $\overline{L}_r^1$ associated with the vertices $r \in \mathcal{V}$ of the compatibility graph, specifically the standard values (13) from [21], the layer-based values (15), and clique-based values (14).

Our computational experiments are conducted on a diverse set of real-world instances derived from the French railway network, ensuring both the practical relevance and robustness of the analysis.

All tests are executed on an Ubuntu server equipped with an Intel(R) Xeon(R) Gold 6252N CPU and 96 GB of RAM. The models are implemented in C++ and solved using *IBM ILOG CPLEX v12.10* with default settings, running in single-thread mode. A time limit of 300 seconds is imposed for each instance, but our results show that the novel G formulation is capable of solving most of the tested instances in significantly shorter times, demonstrating its practical effectiveness even under tighter operational constraints.

To the best of our knowledge, this is the first comprehensive computational study that compares multiple formulations for the TSRSP, with the aim of either solving instances to optimality or establishing meaningful optimality gaps when a time limit is reached. In contrast, previous computational works have primarily focused on heuristic approaches (see, e.g., Samà et al. [40]). The only exception is Pascariu et al. [33], where some preliminary computational results for the S formulation are reported. However, these results show that the S formulation struggled to find optimal solutions, even under large time limits.

7.1. Real-world TSRSP instances

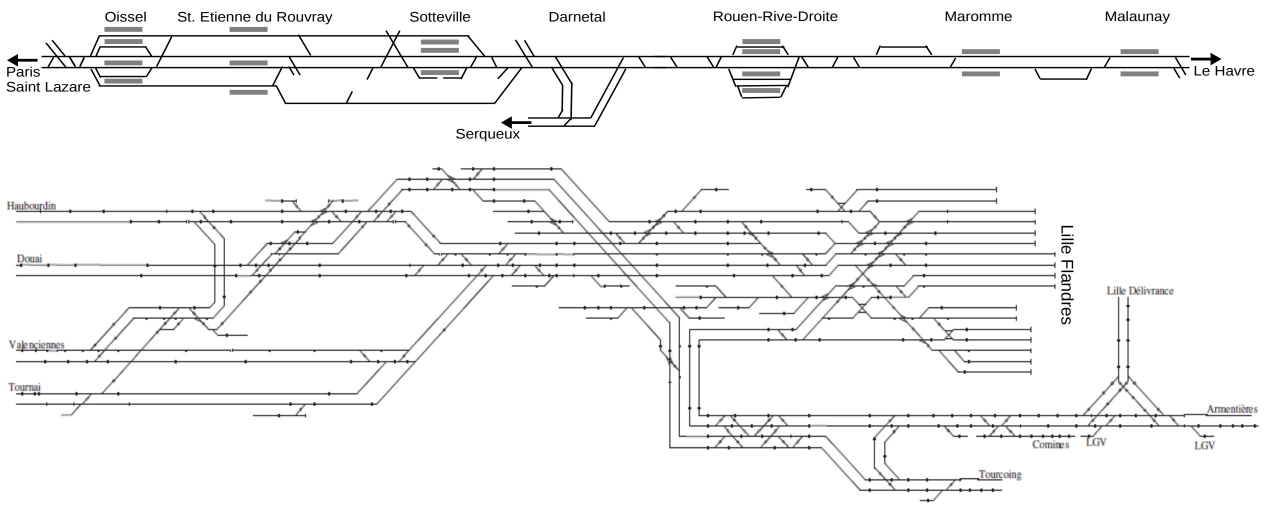
The instances considered in our computational study come from two key control areas of the French railway network, where a control area is a portion of the network in which traffic is controlled by an operator:

1. The first one is the *Rouen control area*, illustrated in figure 4. The infrastructure has a length of approximately 27 km and encompasses the region around the city of Rouen, where both freight and passenger trains operate. The infrastructure primarily consists of double-track line segments connecting several intermediate stations. In total, over 11,300 different routes are available, and a train traversing the entire area can choose among roughly 400 alternative routes. A typical daily timetable includes slightly more than 180 trains.
2. The second one is the *Lille control area*, also shown in figure 4. It covers the Lille Flandres station area, a terminal station with 17 platforms used by intercity and high-speed trains. The infrastructure has a length of approximately 12 km and can be traversed using one of its 2,409 routes. Each train can have up to about 200 alternative routes, and a typical daily timetable includes more than 500 trains.

Our TSRSP instances are derived from a real one-day timetable and generated through a controlled perturbation process which simulates train delays.

For the Rouen control area, we randomly select 20% of the trains in the timetable and introduce a random delay between 5 and 15 minutes at their entry point into the designated area. We then randomly select a starting time and define a one-hour time window: all trains entering the control area within this horizon, whether delayed or on time, are included in the instance. In total, we consider five different perturbation scenarios, each with ten distinct starting times, chosen during either peak hours (7:30–9:00 and 18:30–20:00) or off-peak hours. This procedure yields 100 base TSRSP instances, evenly split between peak and off-peak periods. As these instances feature layer-wise complete compatibility graphs, we introduce additional instances by limiting the number of compatible route pairs. Specifically, for each instance, we randomly remove 10% and 20% of the edges of the compatibility graph, generating two additional versions of the instance. This results in a total of 300 instances, categorized by the edge density parameter $\varepsilon \in \{1.0, 0.9, 0.8\}$. The number of trains per instance ranges from 6 to 17, reflecting various levels of network congestion. The number of routes per

Figure 4: The Rouen and Lille Flandres control areas.



instance ranges from 383 to 983, while the number of compatible route pairs varies between approximately 39,000 and over 424,000, depending on the number of trains and the edge density.

For the Lille control area, we consider a 30-minute time window and generate five perturbation scenarios at two different starting times—one during peak hours and one during off-peak hours—resulting in 20 instances. These instances are characterized by very large compatibility graphs and very high congestion levels, with 8 to 25 trains per instance and a broad range of route alternatives. The average edge density in this set is 0.98. The Lille instances are significantly larger and denser than those from Rouen. The number of routes per instance in the Lille dataset ranges from 1,294 to 5,540, while the number of compatible route pairs varies between approximately 600,000 and over 14 million, highlighting the extremely high density of the corresponding compatibility graphs.

In all these TSRSP instances, the route and pairing costs represent potential delays in seconds. Following the definition in Samà et al. [40], we compute the potential delay as the nonnegative difference between the actual travel time of a train along a selected route and its scheduled travel time in the timetable.

An online supplement is available at `https://github.com/AnnaLivia/TSRSP-Instances/tree/main/data`. It includes all benchmark instances used in our computational experiments, along with a detailed description of the input data formats. The repository also provides summary statistics for each instance—including the number of trains, routes, compatible pairs of routes, and the optimal objective value (when available).

7.2. Results for the Rouen control area instances

For the Rouen control area instances with edge density $\varepsilon = 1$, whose compatibility graphs are layer-wise complete, the clique constraints are not necessary. For this reason, we test simplified variants of the formulations without these constraints, denoted Q, GW, G, and $\hat{G}$. For the instances with edge densities $\varepsilon = 0.9$ and $\varepsilon = 0.8$, where the clique constraints are required, we test two variants of each formulation: one using the non-edge constraints (10) and the other using the anti-neighborhood constraints (11), denoted Q_a, GW_a, G_a, $\hat{G}_a$ and Q_b, GW_b, G_b, $\hat{G}_b$, respectively. The tests of formulation $\tilde{G}$ are discussed separately in Section 7.2.2, as computing the clique-based values (14) can be extremely time-consuming for these instances. Formulation S does not include any clique constraints and is therefore always tested in a single variant.

Table 3 presents the results of the tests conducted on the 300 TSRSP instances derived from the Rouen control area. The table is divided into three parts, each corresponding to a set of 100 instances with a specific edge density: $\varepsilon = 1$, $\varepsilon = 0.9$, and $\varepsilon = 0.8$. For each tested formulation, the corresponding row reports the minimum, average, and maximum solution times (in seconds) across the 100 instances in the respective group (columns "CPU time (s)"). If an instance is not solved within the time limit, the symbol "*t.ℓ.*" is shown to indicate that the maximum allowed time of 300 seconds was reached. Average solution times include these time-limited instances. The last column ("#opt") reports the number of instances, out of 100, that were solved to proven optimality within the time limit for each formulation. For interested readers, additional information on computational times is provided in the online supplement at `https://github.com/AnnaLivia/TSRSP-Instances/tree/main/results`. The repository includes box plots of CPU times, box plots of optimality gaps, and performance profiles corresponding to the experiments reported in Table 3.

The results shown in table 3 demonstrate the superior performance of the G formulation compared to the benchmark formulations S, Q, and GW across all TSRSP instances derived from the Rouen control area. In particular, the improved variant $\hat{G}$ of the G formulation stands out across all edge densities. For $\varepsilon = 1$, it solves all 100 instances to optimality with an average computational time of just 2.4 seconds. For $\varepsilon = 0.9$ and $\varepsilon = 0.8$, formulations $\hat{G}_a$ and $\hat{G}_b$ also solve all instances to optimality, with average solution times below 20 seconds—and often under 10 seconds. The excellent performance of the $\hat{G}$ formulation is primarily due to the strength of its LP relaxation (see Section 7.2.1) and the fact that it involves fewer variables and constraints than the other formulations.

The basic formulation G also performs well for $\varepsilon = 1$, solving 92 out of 100 instances with a significantly lower average time than the other benchmark formulations. For $\varepsilon = 0.9$ and $\varepsilon = 0.8$, formulations G_a and G_b consistently solve more than 76 instances to optimality, with significantly lower computational times than Q_a, Q_b, GW_a, and GW_b. However, the difference in performance between the G and $\hat{G}$ formulations highlights the benefit of using the layer-based values (15) in these instances. Specifically, when the G formulation is compared to its improved variants $\hat{G}_a$ and $\hat{G}_b$, it exhibits a noticeable increase in computational time and a smaller number of solved instances.

Regarding the benchmark formulations, the results confirm that S, Q, and GW are significantly less efficient than the G formulations, both in terms of computational time and number of instances solved to optimality. In particular, formulation S struggles with all three edge densities: for $\varepsilon = 1$, it solves only 19 instances to optimality, and for $\varepsilon = 0.8$ and $\varepsilon = 0.9$, it solves just 5 and 12 instances, respectively, with average solution times close to the time limit. Moreover, formulation S fails to provide even a feasible (integer) solution for 9 instances with $\varepsilon = 0.9$ and 44 instances with $\varepsilon = 0.8$ out of 100. Formulation Q performs slightly better but

Table 3: Performance comparison of the different formulations for the TSRSP instances derived from the Rouen control area with $\varepsilon = 1$, $\varepsilon = 0.9$, and $\varepsilon = 0.8$. Time limit: 300 seconds.

	$\varepsilon = 1$			
	CPU time (s)			
Formulation	*min*	*avg*	*max*	#opt
S	1.5	251.3	*t.ℓ.*	19/100
Q	30.8	254.3	*t.ℓ.*	24/100
GW	3.8	234.8	*t.ℓ.*	33/100
G	0.1	43.2	*t.ℓ.*	92/100
$\hat{G}$	0.1	2.4	23.4	100/100

	$\varepsilon = 0.9$				$\varepsilon = 0.8$			
	CPU time (s)				CPU time (s)			
Formulation	*min*	*avg*	*max*	#opt	*min*	*avg*	*max*	#opt
S	1.4	269.5	*t.ℓ.*	12/100	1.7	278.4	*t.ℓ.*	5/100
Q_a	1.8	248.4	*t.ℓ.*	26/100	0.4	247.8	*t.ℓ.*	26/100
Q_b	2.7	251.5	*t.ℓ.*	27/100	0.7	252.8	*t.ℓ.*	24/100
GW_a	2.0	227.0	*t.ℓ.*	38/100	0.6	189.9	*t.ℓ.*	50/100
GW_b	3.0	225.6	*t.ℓ.*	39/100	0.7	195.3	*t.ℓ.*	51/100
G_a	0.5	121.6	*t.ℓ.*	76/100	0.2	103.6	*t.ℓ.*	83/100
G_b	0.6	112.9	*t.ℓ.*	83/100	0.4	100.6	*t.ℓ.*	85/100
$\hat{G}_a$	0.3	10.0	105.9	100/100	0.1	15.2	143.0	100/100
$\hat{G}_b$	0.3	8.2	54.5	100/100	0.2	13.9	185.2	100/100

remains limited: the variants Q_a and Q_b never solve more than 27 instances in the denser test sets and often exceed 240 seconds on average. Formulation GW, although slightly more competitive, also fails to match the performance of G. For $\varepsilon = 0.9$, GW_a and GW_b solve at most 39 out of 100 instances with an average CPU time below 230 seconds; while for $\varepsilon = 0.8$, formulations GW_a and GW_b solve at most 51 instances in an average time of 190 and 195 seconds, respectively.

The *percentage optimality gap* is defined as the relative difference between the best feasible solution value and the best lower bound computed within the time limit by a given formulation. The corresponding box plots are available in the online supplement at `https://github.com/AnnaLivia/TSRSP-Instances/tree/main/results`, and provide insight into the solution quality when optimality is not reached. Among all tested formulations, the G variants exhibit the best behavior, producing smaller gaps on unsolved instances and solving more instances to optimality. In contrast, formulations S, GW, and Q lead to more suboptimal solutions with significantly larger gaps—often exceeding 60%—especially at higher edge densities. The performance of the G variants degrades as edge density decreases, with G_a maintaining an advantage over G_b, confirming the effectiveness of non-edge constraints. These results underline the robustness of the G-based formulations both in terms of solution quality and number of instances solved across all instance groups.

7.2.1. Empirical comparison of the strength of the LP relaxations of the formulations

Figure 5 presents box plots of the percentage gap between the optimal solution value of the LP relaxation of a MILP formulation and the optimal solution value of the instance (denoted as LP gap (%)), for TSRSP instances derived from the Rouen control area. The results are grouped by edge density values $\varepsilon = 1$, $\varepsilon = 0.9$, and $\varepsilon = 0.8$. Each box shows the median, interquartile range, and outliers of the LP gaps over the 100 tested instances.

Across all three density levels, the results clearly confirm the strength of the LP relaxation provided by the $\hat{G}_a$ and $\hat{G}_b$ formulations, which consistently achieve the best performance. This is particularly evident for $\varepsilon = 1$, where the median LP gap drops to approximately 30%. For $\varepsilon = 0.9$ and $\varepsilon = 0.8$, the improvement remains noticeable, with an average below 40%, although the variability increases slightly as the edge density decreases. In contrast, the other formulations—namely S, GW, and G—exhibit significantly weaker LP relaxations, with gaps consistently close to 100% and low dispersion. As the edge density decreases, these formulations display a growing number of outliers, indicating that a subset of instances becomes easier to relax under sparser conditions.

Notably, the modeling choice regarding clique constraints also affects the quality of the LP relaxation. Formulations GW_a, G_a, and $\hat{G}_a$, which incorporate the non-edge constraints (10), tend to yield smaller LP gaps than their b-type counterparts including anti-neighborhood constraints (11). This behavior aligns with the theoretical results established in proposition 3 (see Section 6.2).

7.2.2. Comparison between the use of the layer-based values and the clique-based values

To conclude our analysis, we evaluate the efficiency of using the clique-based values (14) in place of the layer-based values (15). Since computing the clique-based values can be extremely time-consuming, we restrict the comparison to a subset of 10 Rouen instances with edge

Figure 5: Box plots of the linear programming gaps in percentage for instances from the Rouen control area, grouped by edge density values $\varepsilon = 1$, $\varepsilon = 0.9$, and $\varepsilon = 0.8$.

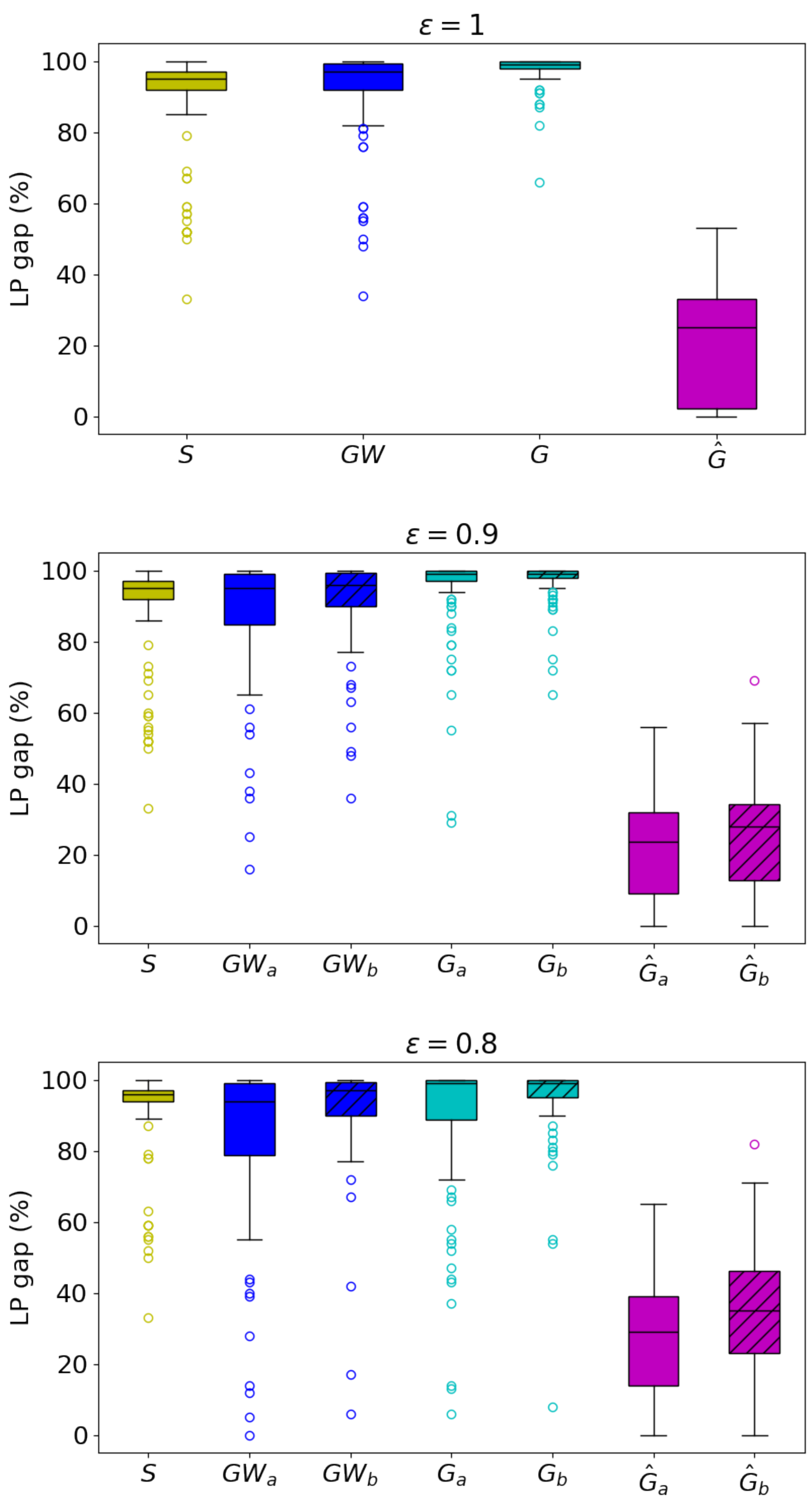


Table 4: Comparative analysis of computational results between formulation $\hat{G}_a$ and $\tilde{G}_a$ on a subset of the Rouen control area instances with $\varepsilon = 0.8$.

											B&B nodes	
Inst.	$\|\mathcal{K}\|$	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	*Obj*	$\downarrow \hat{D}_r^+$	$\uparrow \hat{D}_r^-$	*Time*	$LP(\hat{G}_a)$	$LP(\tilde{G}_a)$	% $\uparrow$	$\hat{G}_a$	$\tilde{G}_a$
R_1	6	421	48,508	418	1	0.00	26	255.46	255.46	0.00	2,528	2,528
R_4	9	555	59,083	438	51	2	4	437.14	437.51	0.08	0	0
R_5	9	701	156,133	202	89	37	1,058	198.50	200.00	0.76	0	3
R_7	10	542	98,658	233	117	21	802	191.86	191.86	0.00	43	35
R_8	10	542	98,658	505	115	38	855	384.08	384.48	0.10	1,025	1,099
R_9	10	542	98,658	666	175	150	834	506.44	513.94	1.48	1,163	1,151
R_{11}	12	383	39,403	365	136	42	13	242.50	310.75	28.14	0	0
R_{12}	12	644	130,042	523	247	117	1,805	296.45	333.05	12.35	499	349
R_{13}	12	673	149,900	152	142	94	2,541	94.39	97.14	2.91	99	110
R_{15}	14	478	68,802	206	93	65	62	176.30	185.25	5.00	0	0

density $\varepsilon = 0.8$. These instances were selected because they feature the smallest compatibility graphs and, for each of them, the optimal value of the LP relaxation of formulations $\hat{G}_a$ and $\hat{G}_b$ remains strictly below the optimal solution value.

For each instance, the clique-based values used in formulations $\tilde{G}_a$ and $\tilde{G}_b$ were computed using the state-of-the-art combinatorial branch-and-bound algorithm by San Segundo et al. [41] to solve problems (14). Table 4 reports in the first six columns the instance characteristics: the instance label, number of layers $|\mathcal{K}|$, number of vertices $|\mathcal{V}|$, number of edges $|\mathcal{E}|$ in the compatibility graph, optimal solution value (column *Obj*), and overall number of improvements identified when computing the clique-based values $\hat{D}_r^+$ and $\hat{D}_r^-$ for the vertices of the graph. Column "Time" reports the total time (in seconds) required to compute these bounds for all vertices in the instance. The next three columns report the LP relaxation values obtained using $\hat{G}_a$ and $\tilde{G}_a$ and the relative improvement in percentage (column % $\uparrow$). Finally, the last two columns show the number of branch-and-bound nodes explored when solving each formulation to optimality.

The results reveal that while the use of clique-based values (14) can strengthen the LP relaxation—as seen, for instance, in R_{11} with a 28.14% improvement—these gains are generally modest. In several cases (e.g., R_1, R_4, R_5, R_7, R_8, R_9, R_{13}), the LP values of $\hat{G}_a$ and $\tilde{G}_a$ are either identical or differ by less than 3%, despite substantial computation times for generating clique-based bounds (up to 2,541 seconds in R_{13}). Moreover, the number of branch-and-bound nodes remains almost unchanged between the two formulations. For some instances, including R_1, R_4, R_{11}, and R_{15}, the number of explored nodes is identical. In a few cases, such as R_7, R_9 and R_{13}, there is a slight reduction in node count when using $\tilde{G}_a$, suggesting that tighter LP relaxations may help reduce the search space. However, these reductions are not significant enough to justify the computational overhead in most instances.

7.3. Results for the Lille control area instances

We conclude our computational analysis with the results obtained on the dataset derived from the Lille control area. These instances involve larger and denser compatibility graphs, which make them particularly challenging from a computational perspective. As a performance metric, we consider the percentage optimality gap (denoted as Opt gap (%)).

Table 5 presents the Opt gap (%) values for all 20 Lille instances. When an instance is solved to optimality, the corresponding CPU time (in seconds) is reported in parentheses. An asterisk "*" indicates that the lower bound is equal to the trivial value (i.e., zero), while "–" denotes that no feasible solution was found within the time limit.

The aggregated results confirm the superior performance of the $\hat{G}_a$ and $\hat{G}_b$ formulations, which outperform all the benchmarks both in terms of optimality gap and number of solved instances. Both formulations solve 3 out of 20 instances to proven optimality. Their average optimality gaps are the lowest among all formulations: 8.9% for $\hat{G}_a$ and 10.2% for $\hat{G}_b$. They also exhibit the smallest maximum gaps (31.8% and 30.8%, respectively), and require minimal computational time when optimality is reached (typically under 20 seconds). Formulations G_a and G_b solve only one and two instances, respectively, with substantially higher average gaps of 56.1% and 68.0%. Their CPU times remain under five minutes when optimality is reached. In contrast, all other benchmark formulations—namely S, Q_a, Q_b, GW_a, and GW_b—perform significantly worse. None of them solve any instance to optimality, and their average gaps remain extremely high: 94.9% for Q_a, 97.5% for Q_b, 97.6% for GW_a, and 96.8% for GW_b. Their best gaps are always above 83%. Formulation S fails to compute any meaningful lower bound, consistently returning the trivial value of zero. Moreover, it is the only formulation that does not find a feasible integer solution within the time limit in 10 out of the 20 instances.

8. Conclusions

The TSRSP is an important problem in RTM that seeks to assign one route to each train, minimizing the total cost while ensuring pairwise compatibility among the selected routes. In this paper, we introduced and tested a novel MILP formulation for the TSRSP obtained by adapting the linearization technique of Glover [21] to a BQP formulation of the problem. By leveraging the specific structure of the TSRSP, we further strengthened the LP relaxation of the new MILP model. We conducted a comprehensive comparative analysis against existing formulations from the literature, evaluating them from both theoretical and computational perspectives. Extensive computational experiments were performed on a diverse set of benchmark instances derived from the French railway network, including a sensitivity analysis with respect to the number of compatible route pairs. The novel MILP formulation consistently demonstrated superior performance, primarily due to the quality of its LP relaxation. Within a five-minute time limit, it outperformed all other benchmark formulations, successfully solving instances with up to 1,500 routes and 1 million compatible route pairs. Moreover, for larger instances with approximately 5,000 routes and 10 million compatible pairs, the new MILP formulation achieved an average optimality gap of 10%, confirming its effectiveness and scalability for large-scale instances.

Future research could explore extending the proposed MILP formulation to the more general Train Routing Selection Problem, in which multiple routes are assigned to each train. Developing an exact model for this broader setting would require a substantial generalization of the linearization technique studied in this work—a direction that warrants further investigation. Additionally, heuristic approaches for the general problem could be designed by repeatedly

Table 5: Percentage optimality gap (Opt gap %) reached for the Lille control area instances within the time limit. When the instance is solved to optimality we report the computational times in seconds (CPU time). The symbol "*" denotes that the lower bound is equal to the trivial bound 0, while "-" denotes that no feasible solution has been found within the time limit of 300 seconds.

	Opt gap % (CPU time)								
Inst.	S	Q_a	Q_b	GW_a	GW_b	G_a	G_b	$\hat{G}_a$	$\hat{G}_b$
L_1	*	85.71	*	*	*	(219.1s)	(190.1s)	(3.5s)	(3.9s)
L_2	*	*	*	*	*	14.25	(286.3s)	(9.2s)	(9s)
L_3	*	99.1	99.74	99.1	99.1	1.18	1.14	(12.8s)	(19.9s)
L_4	*	99.81	99.71	99.71	99.96	2.08	1.85	0.6	0.68
L_5	*	96.67	96.15	97.86	95.45	22.05	23.05	10.8	12.6
L_6	*	*	*	*	*	61.76	64.6	3.12	12.5
L_7	*	83.41	84.47	85.77	83.06	41.06	39.61	31.75	27.57
L_8	*	*	*	*	*	51.81	85.43	7.15	7.33
L_9	-	98.94	99.18	98.99	98.6	43.02	84.03	0.4	0.56
L_{10}	*	*	*	*	*	96.3	95.72	1.3	1.13
L_{11}	-	97.86	97.48	97.92	96.13	4.78	87.95	3.41	2.39
L_{12}	-	98.98	99.23	99.19	98.77	82.96	83.81	10.66	10.69
L_{13}	-	*	*	*	*	64.49	47.25	12.32	25.5
L_{14}	-	99.61	99.71	99.71	99.83	95.19	95.65	5.77	5.91
L_{15}	-	84.16	99.85	*	*	83.84	85.06	11.87	10.06
L_{16}	-	*	*	*	*	98.99	99.19	2.08	2.14
L_{17}	*	*	*	*	*	*	*	5.43	5.13
L_{18}	-	99.87	99.87	99.91	99.9	99.42	99.08	7.87	7.89
L_{19}	-	*	*	*	*	91.05	94.35	30.37	30.81
L_{20}	-	*	*	*	*	*	*	6.72	9.83
#opt	0	0	0	0	0	1	2	3	3
Avg.	*	94.9	97.5	97.6	96.8	56.1	68.0	8.9	10.2
Min.	*	83.4	84.5	85.8	83.1	1.2	1.1	0.4	0.6
Max.	*	*	*	*	*	*	*	31.8	30.8

solving instances of the TSRSP. Both exact and heuristic methods could then be integrated into larger RTM frameworks, enhancing their effectiveness in complex real-world applications.

References

[1] W. P. Adams, R. J. Forrester, and F. W. Glover. Comparisons and enhancement strategies for linearizing mixed 0-1 quadratic programs. *Discrete Optimization*, 1(2):99–120, 2004.

[2] B. Alidaee, F. Glover, G. Kochenberger, and H. Wang. Solving the maximum edge weight clique problem via unconstrained quadratic programming. *European Journal of Operational Research*, 181(2):592–597, 2007.

[3] A. Billionnet. Different Formulations for Solving the Heaviest K-Subgraph Problem. *INFOR: Information Systems and Operational Research*, 43(3):171–186, 2005.

[4] A. Billionnet, S. Elloumi, and M.-C. Plateau. Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method. *Discrete Applied Mathematics*, 157(6):1185–1197, 2009.

[5] C. Bliek, P. Bonami, and A. Lodi. Solving Mixed-Integer Quadratic Programming problems with IBM-CPLEX: a progress report. In *26th RAMP Symposium*, 2014.

[6] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

[7] P. Bonami, A. Lodi, and G. Zarpellon. A Classifier to Decide on the Linearization of Mixed-Integer Quadratic Problems in CPLEX. *Operations Research*, 70(6):3303–3320, 2022.

[8] V. Cacchiani, D. Huisman, M. Kidd, L. Kroon, P. Toth, L. Veelenturf, and J. Wagenaar. An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, 63:15–37, 2014.

[9] A. Caprara. Constrained 0–1 quadratic programming: Basic approaches and extensions. *European Journal of Operational Research*, 187(3):1494–1503, 2008.

[10] F. Corman and L. Meng. A Review of Online Dynamic Models and Algorithms for Railway Traffic Management. *Intelligent Transportation Systems, IEEE Transactions on*, 16(3):1274–1284, 2015.

[11] F. Corman, A. D'Ariano, D. Pacciarelli, and M. Pranzo. A tabu search algorithm for rerouting trains during rail operations. *Transportation Research Part B*, 44(1):175–192, 2010.

[12] F. Corman, A. D'Ariano, D. Pacciarelli, and M. Pranzo. Dispatching and coordination in multi-area railway traffic management. *Computers & Operations Research*, 44:146–160, 2014.

[13] A. L. Croella. *Real-time Train Scheduling: reactive and proactive algorithms for safe and reliable railway networks.* PhD thesis, Sapienza University of Rome, Department of Computer, Control, and Management Engineering Antonio Ruberti (DIAG), Italy, 2022.

[14] A. L. Croella, V. Sasso, L. Lamorgese, C. Mannino, and P. Ventura. Disruption Management in Railway Systems by Safe Place Assignment. *Transportation Science*, 56(4): 938–952, 2022.

[15] A. L. Croella, B. Luteberget, C. Mannino, and P. Ventura. A MaxSAT approach for solving a new Dynamic Discretization Discovery model for train rescheduling problems. *Computers & Operations Research*, 167:106679, 2024.

[16] A. D'Ariano, D. Pacciarelli, and M. Pranzo. A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research*, 183(2):643–657, 2007.

[17] F. Della Croce and R. Tadei. A multi-KP modeling for the maximum-clique problem. *European Journal of Operational Research*, 73(3):555–561, 1994.

[18] W. Fang, S. Yang, and X. Yao. A survey on problem models and solution approaches to rescheduling in railway networks. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):2997–3016, 2015.

[19] M. Fischetti and M. Monaci. Using a general-purpose Mixed-Integer Linear Programming solver for the practical solution of real-time train rescheduling. *European Journal of Operational Research*, 263(1):258–264, 2017.

[20] F. Furini and E. Traversi. Theoretical and computational study of several linearisation techniques for binary quadratic problems. *Annals of Operations Research*, 279(1-2): 387–411, 2019.

[21] F. Glover. Improved Linear Integer Programming Formulations of Nonlinear Integer Problems. *Management Science*, 22(4):455–460, 1975.

[22] F. Glover and E. Woolsey. Technical note—converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, 22(1):180–182, 1974.

[23] N. Kumar and A. Mishra. Role of Artificial Intelligence in Railways: An Overview. In R. K. Phanden, K. Mathiyazhagan, R. Kumar, and J. Paulo Davim, editors, *Advances in Industrial and Production Engineering*, pages 323–330, Singapore, 2021. Springer Singapore.

[24] T. Lanciano, A. Miyauchi, A. Fazzone, and F. Bonchi. A Survey on the Densest Subgraph Problem and its Variants. *ACM Computing Surveys*, 56(8), 2024.

[25] A. N. Letchford. The Boolean Quadric Polytope. In *The Quadratic Unconstrained Binary Optimization Problem*, Springer Books, pages 97–120. Springer, 2022.

[26] E. M. Macambira and C. C. de Souza. The edge-weighted clique problem: Valid inequalities, facets and polyhedral computations. *European Journal of Operational Research*, 123(2):346–371, 2000.

[27] R. Martí, M. Gallego, and A. Duarte. A branch and bound algorithm for the maximum diversity problem. *European Journal of Operational Research*, 200(1):36–44, 2010.

[28] A. Mehrotra. Cardinality constrained boolean quadratic polytope. *Discrete Applied Mathematics*, 79(1):137–154, 1997.

[29] L. Meng and X. Zhou. Simultaneous train rerouting and rescheduling on an N-track network: A model reformulation with network-based cumulative flow variables. *Transportation Research Part B: Methodological*, 67:208–234, 2014.

[30] G. L. Nemhauser and L. E. Trotter Jr. Vertex packings: structural properties and algorithms. *Mathematical Programming*, 8(1):232–248, 1975.

[31] M. W. Padberg. The boolean quadric polytope: Some characteristics, facets and relatives. *Mathematical Programming*, 45(1):139–172, 1989.

[32] K. Park, K. Lee, and S. Park. An extended formulation approach to the edge-weighted maximal clique problem. *European Journal of Operational Research*, 95(3):671–682, 1996.

[33] B. Pascariu, M. Sama, P. Pellegrini, A. D'ariano, D. Pacciarelli, and J. Rodriguez. Train routing selection problem: Ant colony optimization versus integer linear programming. *IFAC-PapersOnLine*, 54(2):167–172, 2021. 16th IFAC Symposium on Control in Transportation Systems CTS 2021.

[34] B. Pascariu, M. Sama, P. Pellegrini, A. D'Ariano, J. Rodriguez, and D. Pacciarelli. Effective train routing selection for real-time traffic management: Improved model and ACO parallel computing. *Computers & Operations Research*, 145:105859, 2022.

[35] B. Pascariu, M. Sama, P. Pellegrini, A. d'Ariano, J. Rodriguez, and D. Pacciarelli. Formulation of train routing selection problem for different real-time traffic management objectives. *Journal of Rail Transport Planning & Management*, 31:100460, 2024.

[36] P. Pellegrini, G. Marlière, R. Pesenti, and J. Rodriguez. RECIFE-MILP: an effective MILP-based heuristic for the real-time railway traffic management problem. *Intelligent Transportation Systems, IEEE Transactions on*, 16(5):2609–2619, 2015.

[37] D. Pisinger. Upper bounds and exact algorithms for p-dispersion problems. *Computers & Operations Research*, 33(5):1380–1398, 2006.

[38] D. Pisinger. The quadratic knapsack problem—a survey. *Discrete Applied Mathematics*, 155(5):623–648, 2007.

[39] M. Samà, P. Pellegrini, A. D'Ariano, J. Rodriguez, and D. Pacciarelli. On the tactical and operational train routing selection problem. *Transportation Research Part C: Emerging Technologies*, 76:1–15, 2017.

[40] M. Samà, P. Pellegrini, A. D'Ariano, J. Rodriguez, and D. Pacciarelli. Ant colony optimization for the real-time train routing selection problem. *Transportation Research Part B: Methodological*, 85:89–108, 2016.

[41] P. San Segundo, F. Furini, and J. Artieda. A new branch-and-bound algorithm for the Maximum Weighted Clique Problem. *Computers & Operations Research*, 110:18–33, 2019.

[42] P. San Segundo, F. Furini, and R. León. A new branch-and-filter exact algorithm for binary constraint satisfaction problems. *European Journal of Operational Research*, 299 (2):448–467, 2022.

[43] B. Sharma, B. Pascariu, P. Pellegrini, J. Rodriguez, and N. Chaudhary. A Real-time Railway Traffic Management Approach Preserving Passenger Connections. *IEEE Access*, pages 79066–79081, 2024.

[44] C. Wen, P. Huang, Z. Li, J. Lessan, L. Fu, C. Jiang, and X. Xu. Train dispatching management with data-driven approaches: a comprehensive review and appraisal. *IEEE Access*, 7:114547–114571, 2019.

Acknowledgements

The work has been carried out within the MOST– Sustainable Mobility National Research Center and received funding from the European Union Next-Generation EU (PIANO NAZIONALE DI RIPRESA E RESILIENZA(PNRR) – MISSIONE 4. COMPONENTE 2, INVESTIMENTO 1.4– D.D. 1033 17/06/2022, CN00000023). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.