Integrating Power Profile Optimization with Timetabling for Underground Train Networks

Deepthi Cherullil Padinjaroot* Tobias Kuen†

August 15, 2026

Abstract

We study energy-efficient operation of underground train networks, where energy from regenerative braking is usable only if another train in the same electrically isolated subnetwork accelerates simultaneously. Timetabling models for this setting typically fix one velocity profile per leg and running time, which limits the matching of braking and accelerating phases. We drop this assumption and optimize the power profile of every leg jointly with departure and running times in a single mixed-integer program. After discretizing time, distance, and velocity, we develop two formulations together with algorithms that make them tractable at practical instance sizes. The first builds a multi-leg trajectory graph whose arcs are enabled or blocked by the timetable decisions, turning the problem into network design. We solve it by a tailored Benders decomposition with Pareto-optimal cuts. The second formulation replaces the explicit graph by binary state variables and links leg-level and network-level energy through multipartite implications, whose polytope we exploit for separation. A computational study on real-world data from the underground train network in Nuremberg compares both formulations and quantifies the effects of decomposition and separation. Within the discretized model, optimized solutions save about 16% energy over an unoptimized reference and 7% over a timetable optimized without power profiles.

Keywords: Railway Timetabling, Multipartite Implication Polytope, Benders Decomposition, Power Profile Optimization.

1 Introduction

Urban rail systems are the backbone of modern public transportation, offering high-capacity and reliable mobility for millions of passengers daily. However, these networks are among the largest energy consumers in the municipal sector. The operation of an underground train network, in particular, presents a fundamental conflict between service quality, which demands high frequency and strict schedule adherence, and operational efficiency, which necessitates a reduction in energy consumption and costs. A significant portion of this energy is consumed during the acceleration phases of trains. Consequently, the optimization of train operations for energy efficiency has become a critical field of research for both transport authorities and the academic community.

This work was part of the Project EKSSE, funded by the Bundesministerium für Bildung und Forschung (BMBF).

*Technical University of Munich (TUM), Germany. deepthi.cherullil@tum.de

†Fraunhofer-Institut für Integrierte Schaltungen IIS, Gruppe Optimization, Nordostpark 84, 90411 Nürnberg, Germany. tobias.kuen@iis.fraunhofer.de

There are two primary levers for reducing energy consumption in train operations (Scheepmaker et al. (2017)). The first is the optimization of train velocity profiles, or trajectories. By extending the running time allocated for a leg (the journey between two adjacent stations), a train can employ longer phases of coasting, which drastically reduces tractive effort (Albrecht et al. (2023)). The second, and more complex, lever is the use of recuperative braking. Modern trains are equipped with regenerative brakes that convert kinetic energy back into electrical energy during deceleration. This recovered energy is fed back into the power grid. If there is no storage device installed in the network, it must be consumed at the exact moment it is produced by another train accelerating within the same electrically isolated power subnetwork. If no train is present to consume this power, the recuperation energy is dissipated as heat through onboard resistors.

This characteristic creates a complex synchronization problem. The energy efficiency of the entire network no longer depends on individual trains in isolation, but on the precise spatio-temporal coordination of all train movements. This challenge has given rise to the field of energy-efficient timetabling, which aims to schedule train arrivals and departures to maximize the overlap between braking and accelerating events (Peña-Alcaraz et al. (2012)). Several optimization models have been proposed to this end, including robust and stochastic approaches that account for the inherent uncertainty in operations. For instance, the work of Gemander et al. Gemander et al. (2023) provides a stochastic mixed-integer programming (MIP) model that minimizes energy consumption under uncertain dwell and running times.

A critical limitation of these existing timetabling models is the assumption of a fixed velocity profile for any given leg running time. That is, the model may decide on a certain running time, and this choice implies a single, pre-calculated energy profile. This rigid assumption severely curtails the potential for energy savings. If a train is braking and recuperating energy in the middle of a leg, another train in the same subnetwork cannot utilize this energy unless it happens to be at its pre-defined acceleration phase, which typically only occurs at the very start of its own leg. This paper addresses this specific gap. We propose to break this rigid coupling by integrating the optimization of the velocity profile directly into the timetabling model. By allowing a train to strategically accelerate or coast at different points along its journey, it can dynamically adapt its power consumption to utilize available recuperative energy from other trains, leading to a truly system-wide optimal solution.

The difficulty of this integration is computational. A timetable for a metro network already leads to a large binary problem, and attaching a velocity profile decision to every leg multiplies the state space. Our focus is therefore on formulations and algorithms that keep this integrated problem solvable for instances taken from real operation, and on keeping the physical modeling separated from the combinatorial structure, so that the level of detail of the energy model can be chosen independently of the solution method. To achieve this integration, we first discretize time, space, and velocity, creating a high-dimensional state space for train operations. We then develop two distinct, novel MIP formulations to solve this integrated problem. The first approach models the trajectory optimization as a shortest path problem on a multi-leg grid graph, analogous to methods used in airline trajectory optimization Hoch and Liers (2023). The integration of timetabling decisions (which define available running times) with this shortest path problem results in a complex network design model. Our second formulation is a binary logic-based model. We define binary variables for each discrete train state (position, velocity, time) and model the energy consumption through a series of logical implications. These dependencies form complex polyhedral structures known as multipartite implication polytopes, a concept recently introduced

in the optimization literature Braun et al. (2026).

Related Literature The research pertinent to this paper spans several domains of optimization. First is the extensive literature on train timetabling. Foundational work in this area, such as that by Cacchiani and Toth (2012), focused primarily on feasibility, robustness, and passenger service quality. More recently, the focus has shifted to include energy consumption. This has led to a class of energy-efficient timetabling problems Scheepmaker et al. (2017). Many of these works, however, primarily focus on energy savings from coasting by optimizing running time supplements.

A more advanced stream of research explicitly models recuperative braking. Among others, Zhu et al. (2025) and Peña-Alcaraz et al. (2012), developed models to synchronize timetables for better energy exchange. These models typically rely on simplified, aggregated energy calculations. The work by Gemander et al. (2023), which forms the basis of our timetabling component, extends this by incorporating stochasticity, but still relies on fixed, pre-computed power profiles.

Separate from timetabling, the domain of train trajectory optimization or energy-efficient train control has been studied in detail. Works such as Albrecht et al. (2023) focus on finding the optimal velocity profile for a single train given a fixed schedule, typically using optimal control or dynamic programming. Wang and Goverde (2019) consider multiple trains and optimize their trajectories for a given timetable.

Previously conducted studies have attempted to combine the fields of energy-efficient timetabling and train control by utilizing heuristic methods, see Ran et al. (2020), Yang et al. (2018). Our work differs fundamentally by proposing a single, integrated mixed-integer programming formulation that solves both problems simultaneously, enabling a true global optimization. Methodologically, our first model draws from the network design literature, and we adapt Benders decomposition techniques tailored for such problems by Magnanti et al. (1986). They introduce methods for generating Benders cuts for a similarly structured uncapacitated network design problem. The work by Magnanti and Wong (1981) provides the theoretical foundation for generating Pareto-optimal cuts. Other works, such as Pearce and Forbes (2018), also use a similar approach to yield strong benders optimality cuts for their master problem, which selects network components and has a continuous-flow subproblem. Our second model builds on novel polyhedral theory. We utilize the structural properties of the bipartite implication polytope, introduced by Burlacu et al. (2024a), and its extension to the multipartite case Braun et al. (2026), applying new theoretical results on constraint separation to solve our binary logic formulation efficiently.

Contribution of this Paper The contribution of this paper is methodological. We show how power profile optimization can be embedded into a network-wide timetabling model for an underground network, and we develop the formulations and algorithms that are needed to solve the resulting problem for instances taken from real operation. We make this concrete in three steps.

First, we introduce two MIP formulations of the integrated problem. The network design formulation couples the timetable variables with a multi-leg trajectory graph, where the timetable decides which arcs of the graph may be used. For this formulation we develop a Benders decomposition in which the master problem selects the timetable and the subproblem determines the energy-optimal trajectories. The subproblem is a shortest path problem, which is the reason why the Pareto-optimal cuts of Magnanti et al. (1986) become affordable here: the reference objective value can be obtained by Dijkstra's algorithm instead of a second linear program. The logic-based formulation avoids the multi-leg graph, whose size

grows exponentially in the number of legs per cluster. It represents discrete train states by binary variables and connects leg-level and network-level energy consumption through multipartite implications. For this formulation we adapt the separation of facet-defining inequalities of the multipartite implication polytope Braun et al. (2026) to the structures that arise in our model.

Second, we report a computational study on real operational data of the two automated metro lines of our application partner VAG in Nuremberg. The study compares the two formulations, shows for which instance characteristics each of them is preferable, and quantifies how much the Benders decomposition and the cut separation contribute. It also reports the energy consumption of the computed solutions relative to two reference timetables. These values are model-based estimates obtained with the discretized energy model; they are not measurements from operation, and verifying them in the field is beyond the scope of this paper.

Third, we keep the physical modeling separated from the combinatorial structure. In both formulations, the train and network physics enter only through the cost of a state transition, so the formulations and the algorithms remain unchanged if a more detailed energy model is used. We make this interface explicit and describe what a continuous trajectory optimization model has to provide in order to serve as the cost oracle of the discrete models. Implementing such a continuous model is not part of this paper; we treat it as the main next step and discuss the consequences for the reported energy figures.

Structure of the Paper The remainder of this paper is organized as follows. In Section 2, we restate the timetabling model and introduce two discrete power profile optimization models. First, the network design formulation and Benders decomposition as an algorithmic enhancement. Second, the binary logic-based model, and its connection to multipartite implication polytopes. Section 3 makes the interface between the discrete models and a continuous power profile optimization model explicit. In Section 4, we present our computational study, detailing the real-world data from VAG Nuremberg and comparing the performance of the two proposed models. Finally, Section 5 concludes the paper. A lookup table for the notation used throughout the paper can be found in the appendix.

2 Integrating Timetabling and Discrete Power Profile Optimization

This section presents our integrated approach to jointly optimizing train timetables and energy-efficient velocity profiles. We first introduce the timetabling model, then describe the trajectory graph structures used to model energy consumption. Building on these components, we develop two solution methods: a network design formulation and a logic-based formulation, each with tailored algorithmic enhancements.

2.1 Timetabling Model

The timetabling model seeks to solve

$$\min\{\xi(\mathbf{x}) \,:\, \mathbf{x} \in X\}, \tag{1}$$

where ξ maps the timetable encoded in $\mathbf{x}$ to its overall energy consumption, and X represents the set of all feasible timetables. The set X is defined via a conflict graph-based stable set formulation, see Gemander et al. (2023) for details. Each leg $k \in \mathcal{K}$ has a set $\mathcal{D}_k$ of feasible departure configurations (σ, ρ), where σ

is a discrete departure time and ρ a running time. The binary variable $x_{k\sigma\rho}$ indicates whether leg k departs at time σ with running time ρ. A timetable $\mathbf{x} \in X$ must satisfy two types of constraints: (i) multiple-choice constraints requiring exactly one configuration per leg, and (ii) pairwise conflict constraints forbidding combinations of configurations that violate operational requirements. These conflicts encode headway times on shared track sections, minimum dwell times at intermediate stations, turnaround times at terminals, and passenger connection time windows at transfer stations.

The key challenge lies in evaluating the objective function ξ: given a fixed timetable, we must determine the energy-optimal velocity profiles for all trains. The following subsection introduces the graph-based structures that enable this evaluation.

2.2 Trajectory Graphs for Single-Leg and Multi-Leg Models

To evaluate and optimize energy consumption for a given timetable, we model the velocity profiles of trains as paths through discretized state-space graphs. We begin by grouping the timetable's legs into clusters, denoted by C, such that any two legs within the same cluster are electrically coupled via power flow. The time horizon of a cluster C is termed $T(C)$ and $T'(C)$ if we exclude the final timestep. Within each cluster, we optimize the velocity profiles of all legs simultaneously, synchronizing their braking and acceleration phases to improve overall energy efficiency.

To achieve this, we discretize each leg's state along three dimensions, time, distance, and velocity, and then model energy consumption at two levels. The first level considers the energy consumed or recovered by a single train leg traveling along its route. The second level considers the combined power interactions among all legs in the cluster, including regenerative braking exchanges. Both levels are represented using multipartite graphs, as described below.

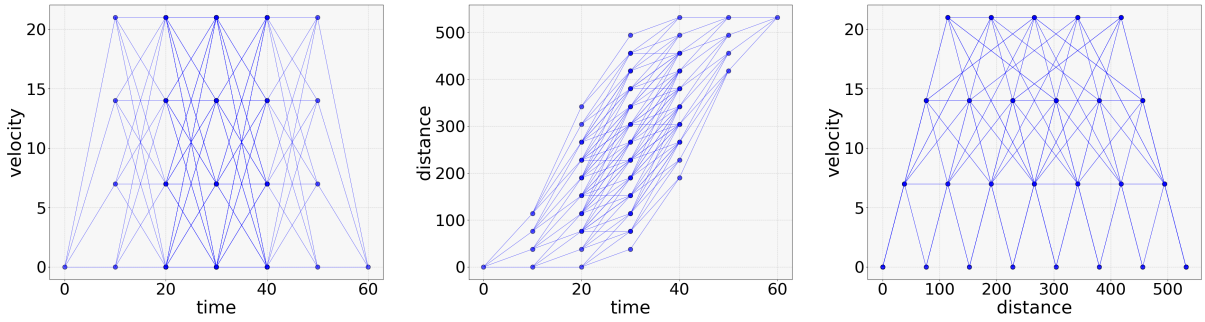

Figure 1: Time distance velocity relation in single-leg trajectory graph.

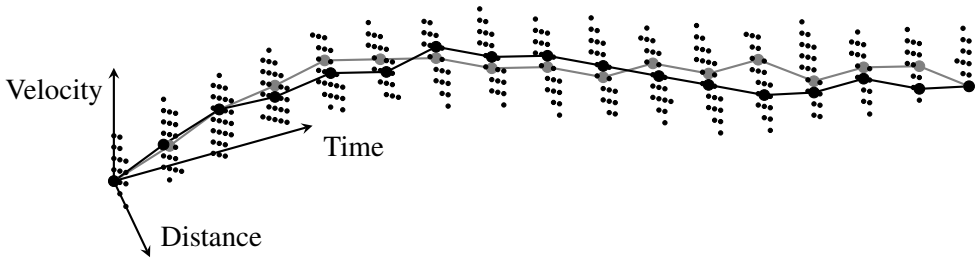

Figure 2: Two feasible paths in a single-leg trajectory graph.

Single-Leg Model For each leg in the cluster, we construct a layered (3D) directed graph. Each node represents a discretized state, characterized by a timestamp t, a distance coordinate d along the route, and a velocity level v. We denote $V_t^k(C)$ for all nodes for a leg $k \in C$ at time $t \in T(C)$. Directed arcs $A_t^k(C)$ connect feasible transitions between nodes in successive time layers. Each arc's weight represents the net power required (or recovered) when moving from state (t, d, v) to state $(t + 1, d', v')$. Figures 1 (3D view) and 2 (2D projections) illustrate this trajectory grid and its associated state graph. We combine the directed trees for all legs in an arborescence $G = (V, A)$

Arc Feasibility and Arc Costs The trajectory graph of a leg is fully determined by the three grids and by two conditions that decide which transitions between two consecutive time layers are admissible.

The first condition is kinematic consistency. Within one time interval we assume constant acceleration, so the distance covered between the states (t, d, v) and $(t + \Delta t, d', v')$ is $\frac{1}{2}(v + v')\Delta t$, where Δt is the step size of the time grid. Since the endpoint of this movement does not in general lie on the distance grid, we accept the transition only if the covered distance, projected onto the nearest grid point, is exactly d'. All other pairs of states are not connected. This projection introduces a distance error of at most half the local grid spacing per time step, which is the price of working on a fixed grid; it is controlled by the number of distance grid points and by Δt.

The second condition bounds the power. For an admissible transition, the energy exchanged with the grid is

$$\Delta E\big((d, v), (d', v')\big) \;=\; \tfrac{1}{2}\tau\left((v')^2 - v^2\right) \;+\; \tau\, g\, r\, v\, \Delta t, \tag{2}$$

that is the change of kinetic energy plus the work against running resistance, evaluated at the velocity at the beginning of the interval. Here τ is the train mass, g the gravitational constant and r the running resistance coefficient. A transition is discarded if $|\Delta E| > P^{\max}\Delta t$, where $P^{\max}$ is the available traction and braking power. This condition removes transitions with unrealistic acceleration or deceleration and, at the same time, keeps the out-degree of the nodes small.

The cost of an arc follows from (2). If $\Delta E \geq 0$, the train draws ΔE from the network and we set $p_{uw} = \Delta E$. If $\Delta E < 0$, the train brakes and feeds energy back, and we set $p_{uw} = \eta\, \Delta E \leq 0$ with a recuperation coefficient $\eta \in (0, 1)$ that accounts for the losses of the conversion chain.

Equation (2) is a point-mass model with constant running resistance and does not contain track gradients or curve resistances. It is important to note, however, that the physics enters both formulations of this section only through the numbers p_{uw}. A refined cost function changes these numbers, but neither the structure of the formulations nor the algorithms. Section 3 returns to this point.

Multi-Leg Model At the cluster level, we consider all legs $k \in C$ over the cluster's planning horizon. Again we construct an arborescence $\mathcal{G} = (\mathcal{V}, \mathcal{A})$. Each node in the cluster graph represents the joint state of all legs at a discrete timestamp $t \in T(C)$, characterized by each leg's distance d and velocity v. The joint state thus comprises $1 + 2|C|$ components: the timestamp together with each leg's (d, v) pair. The set of all nodes at time $t \in T(C)$ is noted $\mathcal{V}_t(C)$

Transitions between successive cluster states from time t to $t + 1$ are represented by arcs $\mathcal{A}_t(C)$ in the multipartite cluster graph. Each arc's weight equals the total power that has to be injected into the network to serve all legs during that interval. To explicitly capture simultaneous acceleration or braking events, we define two arc sets at each time t: $\mathcal{A}_t'(C)$ collects all arcs where a leg transitions from a standing position

to a positive velocity, and $\mathcal{A}_t^{\downarrow}(C)$ collects all arcs where a leg transitions from a positive velocity to rest. These sets allow us to model regenerative energy feedback when one or more legs brake, and concurrent power demands when other legs accelerate. If an arc is assigned both an accelerating and a braking leg, we split this arc as displayed in Figure 3. Specifically, the single arc from u to w is decomposed into a

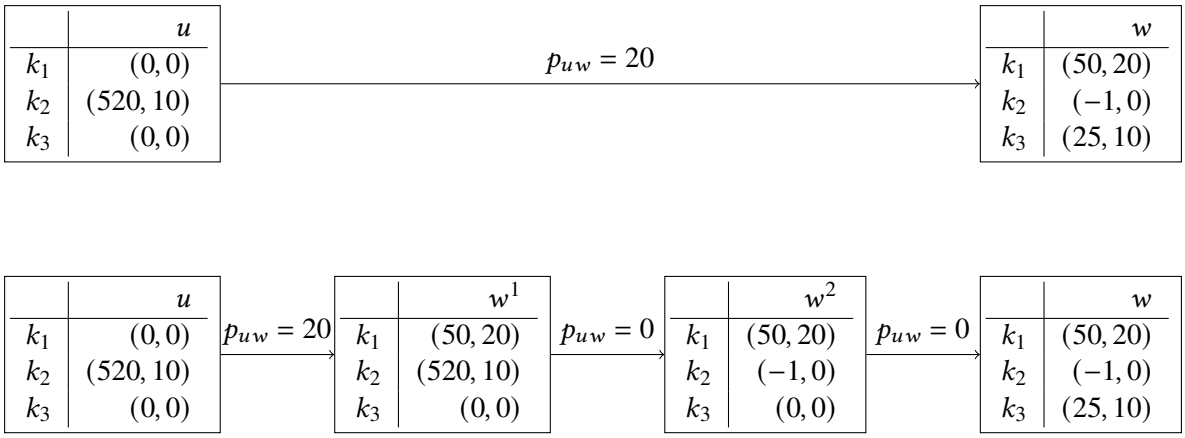

Figure 3: Arc-splitting in a multi-leg trajectory graph. An arc from node u to node w (top) and its splitting into three arcs: $u \to w^1$, $w^1 \to w^2$, $w^2 \to w$ (bottom).

sequence of sub-arcs $u \to w^1 \to w^2 \to w$, where each intermediate node w^i represents a partial state in which only one leg's transition has been applied. The first sub-arc $u \to w^1$ captures the acceleration event of one leg, the second sub-arc $w^1 \to w^2$ captures the braking event of another leg, and the third sub-arc $w^2 \to w$ captures any remaining acceleration. By sequencing the transitions in this way, the energy released during the braking phase on the intermediate arc becomes available to offset the power demand on subsequent acceleration arcs. This decomposition preserves the original arc's total transition semantics while enabling the optimization model to explicitly account for regenerative energy reuse between simultaneously braking and accelerating legs within the same time interval. It allows us to define a function $\zeta() : \mathcal{A} \to C$ that maps each arc to the unique leg that either accelerates from or brakes to a standing position at this arc.

By optimizing over this cluster-level graph, we coordinate multiple legs to minimize overall energy consumption while respecting the timetable constraints.

Having established both the timetabling model and the trajectory graph structures, we now turn to their integration. We propose two formulations that couple timetable selection with trajectory optimization: a network design approach (Method 1) and a logic-based approach (Method 2).

2.3 Method 1: Network Design Formulation

The first method directly integrates the timetabling constraints with the multi-leg trajectory graph by formulating the problem as a network design model. The timetable decisions selectively block or enable nodes in the trajectory graph, and the optimal velocity profiles correspond to shortest paths in the resulting restricted graph.

2.3.1 Base Model

Timetabling Model Model Master is a concrete realization of the abstract formulation (1). The overall energy consumption $\xi(\mathbf{x})$ is decomposed by cluster: for each cluster $C \in \mathcal{C}$, a continuous variable c_C represents the cluster's energy cost, bounded below by the value of a subproblem $Sub_C(\mathbf{x})$ that optimizes the velocity profiles of all legs in the cluster given the timetable $\mathbf{x}$. The total objective (Master.1) then sums these cluster-level costs.

$$\min_{\mathbf{c},\mathbf{x}} \sum_{C \in \mathcal{C}} c_C \tag{Master.1}$$

$$\text{s.t. } c_C \geq Sub_C(\mathbf{x}), \quad \forall C \in \mathcal{C}, \tag{Master.2}$$

$$\mathbf{x} \in X. \tag{Master.3}$$

Flow Formulation

$$\min_{\mathbf{Y}} \sum_{t \in T'(C)} \sum_{uw \in \mathcal{A}_t(C)} p_{uw} Y_{uw} \tag{Sub.1}$$

$$\text{s.t.} \sum_{w \mid wu \in \mathcal{A}_{t-1}(C)} Y_{wu} - \sum_{w \mid uw \in \mathcal{A}_t(C)} Y_{uw} = 0, \quad \forall u \in \mathcal{V}_t(C),\ \forall t \in T(C) \setminus \{0, \bar{t}\}, \tag{Sub.2}$$

$$\sum_{uw \in \mathcal{A}_0(C)} Y_{uw} = 1, \tag{Sub.3}$$

$$\sum_{uw \in \mathcal{A}_{\bar{t}-1}(C)} Y_{uw} = 1, \tag{Sub.4}$$

$$Y_{uw} \leq \sum_{\rho \mid (t,\rho) \in \mathcal{D}_{\zeta(uw)}} \bar{x}_{\zeta(uw) t \rho}, \quad \forall uw \in \mathcal{A}^{\downarrow}_t(C),\ \forall t \in T'(C) \tag{Sub.5}$$

$$Y_{uw} \leq \sum_{\sigma \mid (\sigma, t-\sigma) \in \mathcal{D}_{\zeta(uw)}} \bar{x}_{\zeta(uw) \sigma (t-\sigma)}, \quad \forall uw \in \mathcal{A}^{\uparrow}_t(C),\ \forall t \in T'(C) \tag{Sub.6}$$

$$Y_{uw} \in \mathbb{R}^+, \quad \forall uw \in \mathcal{A}_t(C),\ \forall t \in T'(C). \tag{Sub.7}$$

Model Sub presents a flow-based formulation for determining the shortest path in the multi-leg trajectory graph, which models the transition of train states in the underground network for a set of train legs collected in one cluster. Certain nodes are blocked based on the binary decision variables modeling the timetable, i.e., the departure and arrival times of each individual train. The objective function minimizes the total cost of the selected path, the total energy consumption of the cluster, by summing the weighted flows over all arcs.

The constraints enforce a flow-conservation principle (Sub.2), ensuring that for each intermediate node, the incoming and outgoing flows remain balanced. The source (Sub.3) and sink (Sub.4) constraints guarantee that exactly one unit of flow enters and exits the network. Additionally, blocking constraints (Sub.5) and (Sub.6) restrict the usage of specific nodes depending on the selected timetable, thereby modifying the structure of the feasible path set. Constraint (Sub.5) excludes network state arcs at time t where leg k accelerates from a standing position if no departure configuration is selected for leg k starting at time t. Constraint (Sub.6) analogously excludes network state arcs at time t where leg k brakes to a standing position if no arrival configuration is selected for leg k arriving at time t.

Although the flow variables Y_{uw} are defined continuously over $\mathbb{R}^+$, the optimal solutions are guaranteed

to be naturally binary due to the total unimodularity property of network flow formulations Sifaleras (2013). The node-arc incidence matrix underlying the flow conservation constraints, combined with the standard variable upper bounds introduced by the blocking constraints, forms a totally unimodular constraint matrix structure. Because the boundary conditions (source supply and sink demand) are equal to one and the right-hand side of the blocking constraints is determined by the fixed binary master timetable parameters $\bar{\mathbf{x}}$, all basic feasible solutions and consequently the optimal solution found via linear programming are strictly integral. Given the unit flow limits, these variables naturally take values in $\{0, 1\}$, rendering explicit binary constraints unnecessary.

2.3.2 Algorithmic Enhancement: Benders Decomposition

The choice of timetable $\mathbf{x} \in X$ determines when the trains accelerate from and brake to a standing position. The train accelerates over multiple time intervals. This shapes the feasible set of train trajectories and their associated energy consumption. However, the problem structure allows this complexity to be handled using a Benders decomposition framework. Once the timetable decisions are fixed, the trajectory optimization problem simplifies significantly. The base model in section 2.3.1 can be viewed as a shortest-path problem over a graph whose arcs can be removed or added, as in a network design problem. This makes our problem structurally similar to the uncapacitated network design problem considered by Magnanti et al. Magnanti et al. (1986). So, we can adapt their approach to generate Bender's cuts. The binary timetable design problem is the master problem that gets tightened iteratively using cuts derived from the subproblem, which is the continuous trajectory optimization part. The master problem selects departure and arrival configurations through the binary variables $\mathbf{x}$ and maintains an energy cost variable c_C for each cluster $C \in \mathcal{C}$. For a fixed master solution $\mathbf{x}$, the corresponding subproblem is a linear network design (shortest path) problem whose optimal objective value equals the minimum achievable energy consumption of the cluster.

First, we establish the structural similarity between the two problems. Magnanti et al. (1986) formulated the uncapacitated network design problem with multiple commodities. These commodities might represent distinct physical goods, or the same physical good but with different points of origin and destination. They use a set of binary decision variables indicating whether an arc is part of the network design, and a set of continuous flow variables that determine the flow of a commodity on an arc. These variables are associated with two cost parameters. One is a fixed cost for any given arc to be part of the network, and the other is the cost of routing a commodity via an arc. We can draw parallels to our model as follows: The binary variables $\mathbf{x}$ in our model is similar to their binary decision variables that indicate whether an arc is included in the network's design, and this decision enables or disables all flow across that arc via coupling constraints linking the design and flow. In our case, selecting a timetable variable implies adding or removing arcs in the trajectory graph (those that accelerate from or brake to a standing position). The variables Y, on the other hand, is a continuous flow variable, indicating that at the given timestep, the state of the trains in the cluster changes from u to w. Magnanti et al. (1986) uses a continuous variable similar to this to represent the continuous flow of a commodity on an arc, constrained by flow conservation equations and bounded above by the design variable.

The total energy consumption by cluster C for the chosen timetable is given by c_C, and this corresponds to the total cost of choosing the network. The energy required by a train to transition from state u to w is given by p_{uw}, which corresponds to the routing costs in the model formulation of Magnanti et al. Magnanti

et al. (1986). Both models also have the same types of constraints: flow conservation and arc activation or blocking constraints that prevent the use of arcs incompatible with the chosen network/timetable design.

Having established the structural similarity, we can exploit it to generate Benders cuts. Magnanti et al. (1986) adapt and enhance ideas from Magnanti and Wong (1981) to accelerate the Benders decomposition. Normally, the dual of the subproblem is solved to obtain the Benders cut. But network design problems are typically degenerate, meaning the dual subproblems often have alternate optimal solutions. This gives rise to multiple possible Benders cuts. Magnanti and Wong (1981) showed that among the optimal solutions, the one that is most distant from an interior point of the convex hull of the set of feasible solutions generates the best Benders cut. They prove that by maximizing the cut value at such an interior point, the resulting cut is guaranteed to be Pareto-optimal, meaning it is not dominated by any other possible cut over the entire feasible region. These Pareto-optimal Benders cuts are generated by solving an auxiliary linear program instead of the dual of the subproblem. This auxiliary problem is as follows:

$$\max_{\pi,\alpha,\beta,\mu} \ \alpha + \beta + \Gamma(\mathbf{x}^0) \tag{Aux.1}$$

$$\text{s.t. } \alpha + \beta + \Gamma(\bar{\mathbf{x}}) \leq Sub_C(\bar{\mathbf{x}}) \tag{Aux.2}$$

$$\alpha - \pi_w \leq p_{uw}, \quad \forall uw \in \mathcal{A}_0(C), \tag{Aux.3}$$

$$\pi_u + \beta \leq p_{uw}, \quad \forall uw \in \mathcal{A}_{\bar{t}-1}(C), \tag{Aux.4}$$

$$\pi_u - \pi_w + \mu_{uw} \leq p_{uw}, \quad \forall uw \in \mathcal{A}_t^{\nearrow}(C) \cup \mathcal{A}_t^{\searrow}(C),\ t \in T'(C), \tag{Aux.5}$$

$$\pi_u - \pi_w \leq p_{uw}, \quad \forall uw \in \mathcal{A}_t(C) \setminus \left(\mathcal{A}_t^{\nearrow}(C) \cup \mathcal{A}_t^{\searrow}(C)\right),\ t \in T'(C), \tag{Aux.6}$$

$$\mu_{uw} \geq 0, \quad \forall uw \in \mathcal{A}_t^{\nearrow}(C) \cup \mathcal{A}_t^{\searrow}(C),\ t \in T'(C), \tag{Aux.7}$$

$$\pi, \alpha, \beta \text{ free}, \tag{Aux.8}$$

where

$$\Gamma(\mathbf{x}) = \sum_{t \in T'(C)} \sum_{uw \in \mathcal{A}_t^{\nearrow}(C) \cup \mathcal{A}_t^{\searrow}(C)} \mu_{uw} \left(\sum_{\rho \mid (t,\rho) \in \mathcal{D}_{\zeta(uw)}} x_{\zeta(uw)t\rho} + \sum_{\sigma \mid (\sigma, t-\sigma) \in \mathcal{D}_{\zeta(uw)}} x_{\zeta(uw)\sigma(t-\sigma)} \right). \tag{Aux.9}$$

This auxiliary problem, as described in Magnanti et al. (1986), is similar to the dual of the original subproblem. The variables π_u, α, β, and μ_{uw} are the dual variables corresponding to the flow-conservation constraints (Sub.2), the source and sink constraints (Sub.3 and Sub.4), and the blocking constraints (Sub.5 and Sub.6) respectively.

How the problem differs from the dual is that the objective function uses $\mathbf{x}^0$, which is an interior point of the convex hull of X, the set of all feasible timetables, and has an additional constraint (Aux.2) that restricts the feasible region to the already computed optimal solution of the original subproblem. The interior point $\mathbf{x}^0$ in the objective function acts as a reference point from which the cut value can be maximized, and the constraint Aux.2 ensures that the objective function of the original dual subproblem does not decrease beyond $Sub_C(\bar{\mathbf{x}})$. Where $Sub_C(\bar{\mathbf{x}})$ is the objective value of the original subproblem for the timetable solution $\bar{\mathbf{x}}$ of the master problem.

Normally, this acceleration is costly, since the dual subproblem must be solved first to obtain the optimal solution value, and then it must be solved again with the additional constraint and an altered

objective. So at every step, this leads to solving the linear program twice. However, this is not the case in our problem. Since the subproblem is a shortest-path problem with arc weights given by the energy required to transition from one state node to another, we can use Dijkstra's algorithm to compute $Sub_C(\bar{\mathbf{x}})$. Dijkstra's algorithm is fast and allows us to use the acceleration procedure without solving the linear program twice. This is the reason we use this method and why it performs so well.

The Pareto-Optimal benders cut thus obtained $\forall C \in \mathcal{C}$ is as follows:

$$c_C \geq \alpha + \beta + \sum_{t \in T'(C)} \sum_{uw \in \mathcal{A}_t^{\nearrow}(C) \cup \mathcal{A}_t^{\searrow}(C)} \mu_{uw} \left(\sum_{\rho \mid (t,\rho) \in \mathcal{D}_{\zeta(uw)}} x_{\zeta(uw)t\rho} + \sum_{\sigma \mid (\sigma, t-\sigma) \in \mathcal{D}_{\zeta(uw)}} x_{\zeta(uw)\sigma(t-\sigma)} \right). \quad (6)$$

These cuts are expected to accelerate convergence of the master problem and reduce overall computation time in the Benders decomposition, compared to the usual cut algorithms.

2.4 Method 2: Logic-Based Formulation

While the network design formulation provides a natural integration of timetabling and trajectory optimization, the multi-leg trajectory graph suffers from an exponential growth in the number of arcs as the number of trains in the cluster increases. To address this scalability limitation, we introduce an alternative formulation that decomposes energy consumption modeling into two levels, the individual leg-level and the network-level, and links them through logical implications.

2.4.1 Base Model

This formulation leverages the theory of bipartite and multipartite implication polytopes to couple decisions across levels. The authors Burlacu et al. (2024a) study the bipartite implication polytope, which describes the relations for three sets of binary variables. The selection of exactly one element in each of two implying sets implies the selection of exactly one element in the implied set. The theory was later extended to the case of multiple (> 2) implying sets in Braun et al. (2026).

Train Trajectory Bipartite Implications In the shortest path in the single-leg trajectory graph for each train leg k, we select exactly one node for each timestamp. The selection of two nodes in successive timestamps implies the selection of one arc, the one incident to both nodes. We are not interested in which arc was chosen exactly. We are only interested in the energy consumption assigned to the arc. Therefore, we group the arcs at time t by their energy consumption and add one binary variable for each arc group $v \in W_t^k(C)$. The groups are obtained by agglomerative clustering with complete linkage on the one-dimensional set of arc energies, with a given distance threshold. This yields a bipartite implication relationship for each cluster $C \in \mathcal{C}$, leg $k \in C$, and timestamp $t \in T'(C)$.

$$S^{TRAIN}(C,k,t) \coloneqq \{\boldsymbol{\psi} \in \{0,1\}^{|V_t^k(C)|}, \mathbf{y} \in \{0,1\}^{|W_t^k(C)|} \ : \ (7a), (7b), (7c), (7d)\}$$

$$\sum_{u \in V_t^k(C)} \psi_u = 1, \tag{7a}$$

$$\sum_{u \in V_{t+1}^k(C)} \psi_u = 1, \tag{7b}$$

$$\sum_{v \in W_t^k(C)} y_v = 1, \tag{7c}$$

$$\psi_u \psi_w \leq y_{\Xi_C^{k,t}(uw)}, \quad \forall (u, w) \in A_t^k(C). \tag{7d}$$

The binary variables ψ_u indicate whether a particular node u is selected in the path, while the binary variables y_v indicate which leg-level energy consumption group v out of all possible groups W is active at timestamp t. Ξ maps each arc to its energy consumption group.

Power Grid Multipartite Implications For each leg and timestamp, we select a binary variable representing one energy consumption group. Together, the decisions across all legs determine the overall energy consumption of the network. This relationship can be captured using any model for the power network of the underground train system, such as the one in Burlacu et al. (2024b). To model this, we again group similar network energy consumption patterns and assign one binary variable to each group. As a result, the convex hull of the feasible points is an instance of the multipartite implication polytope for every cluster $C \in \mathcal{C}$ and timestamp $t \in T'(C)$ (see 8c).

$$S^{POWER}(C, t) \coloneqq \{\mathbf{y} \in \{0,1\}^{\sum_{k \in C} |W_t^k(C)|}, \mathbf{z} \in \{0,1\}^{|Z_t(C)|} \ : \ (8a), (8b), (8c)\}$$

$$\sum_{v \in W_t^k(C)} y_v = 1, \forall k \in C \tag{8a}$$

$$\sum_{\ell \in Z_t(C)} z_\ell = 1, \tag{8b}$$

$$\prod_{v \in \boldsymbol{\lambda}} y_v \leq z_{\Theta_C^t(\boldsymbol{\lambda})}, \quad \forall \boldsymbol{\lambda} \in \Omega_t(C). \tag{8c}$$

Here, the binary variables z_ℓ indicate which network-level energy consumption group ℓ out of all possible groups Z is active at timestamp t. Θ maps each combination of train-level energy consumption groups $\boldsymbol{\lambda} \in \Omega_t(C)$ to its network-level energy consumption group.

Integration into Timetabling Model As in the network design formulation, we couple the leg trajectory optimization with the timetabling model by blocking nodes in the trajectory optimization model based on the chosen timetable. If a leg is scheduled to depart at time t, we enforce the selection of the node representing velocity 0 and distance 0 at time t and disallow the selection of this node at time $t + 1$. Similarly, we block nodes that have velocity 0 at the final distance of the leg Δ_k before and enforce them at the scheduled arrival time for each leg.

$$\min_{\mathbf{x},\boldsymbol{\psi},\mathbf{y},\mathbf{z}} \sum_{C\in\mathcal{C}} \sum_{t\in T'(C)} \sum_{\ell\in Z_t(C)} q_\ell z_\ell \tag{9a}$$

$$\text{s.t. } \mathbf{x} \in X, \tag{9b}$$

$$\boldsymbol{\psi}, \mathbf{y} \in S^{TRAIN}(C,k,t), \quad \forall C \in \mathcal{C},\ \forall k \in C,\ \forall t \in T'(C), \tag{9c}$$

$$\mathbf{y}, \mathbf{z} \in S^{POWER}(C,t), \quad \forall C \in \mathcal{C},\ \forall t \in T'(C), \tag{9d}$$

$$\psi_{(\sigma,0,0)} \geq x_{k\sigma\rho}, \quad \forall C \in \mathcal{C},\ \forall k \in C,\ \forall (\sigma,\rho) \in \mathcal{D}_k, \tag{9e}$$

$$\psi_{(\sigma+1,0,0)} \leq 1 - x_{k\sigma\rho}, \quad \forall C \in \mathcal{C},\ \forall k \in C,\ \forall (\sigma,\rho) \in \mathcal{D}_k, \tag{9f}$$

$$\psi_{(\sigma+\rho,\Delta_k,0)} \geq x_{k\sigma\rho}, \quad \forall C \in \mathcal{C},\ \forall k \in C,\ \forall (\sigma,\rho) \in \mathcal{D}_k, \tag{9g}$$

$$\psi_{(\sigma+\rho-1,\Delta_k,0)} \leq 1 - x_{k\sigma\rho}, \quad \forall C \in \mathcal{C},\ \forall k \in C,\ \forall (\sigma,\rho) \in \mathcal{D}_k. \tag{9h}$$

2.4.2 Algorithmic Enhancement: Generalized MPIP Separation

The logic-based formulation introduces multipartite implications, but the LP relaxations may be weak. To tighten them, we exploit the polyhedral structure of the MPIP to separate valid cutting planes. We first recall the relevant definitions and results from Braun et al. (2026), then discuss their application to our setting.

The multipartite implication polytope (MPIP), introduced in Braun et al. (2026), is defined as follows.

We consider an instance involving $n \in \mathbb{N}$ implying sets of binary variables, where, without loss of generality, each set has cardinality $m \in \mathbb{N}$. From each implying set, a single element must be chosen. For the i-th implying set, the choice of the element is denoted by an index $j_i \in [m]$, where $[m]$ denotes $\{1,\dots,m\}$.

The choices across all implying sets are represented by an index vector $\mathbf{j} = (j_1,\dots,j_n) \in [m]^n$. The corresponding implied set has cardinality $\kappa \in \mathbb{N}$. The choices in the implying sets imply a corresponding element $l \in [\kappa]$ from the implied set.

The implication relationship between a potential choice in each implying set and the result in the implied set is defined through a mapping

$$\varphi : [m]^n \to [\kappa],\ \mathbf{j} = (j_1,\dots,j_n) \mapsto l,$$

where $\varphi(\mathbf{j}) = l$ determines the index l of the corresponding element in the implied set. We speak of a multipartite implication instance if parameters m, n, κ, and mapping φ are given. The requirement to select exactly one element from the i-th implying set is modeled as a multiple-choice constraint $\sum_{j\in[m]} \chi^i_j = 1$. The binary points fulfilling the multiple-choice constraints for all implying sets and the implied set are captured by the set

$$M := \left\{ (\chi^1,\dots,\chi^n,\theta) \in \{0,1\}^{mn+\kappa} \;\middle|\; \sum_{j\in[m]} \chi^i_j = 1,\ \forall\, i \in [n],\ \sum_{l\in[\kappa]} \theta_l = 1 \right\}.$$

Using the mapping φ, we restrict the set M to ensure only feasible relations by making use of a

constraint involving a product. This is formalized in the set

$$S(\varphi) := \left\{ (\chi^1, \ldots, \chi^n, \theta) \in M \;\middle|\; \prod_{i \in [n]} \chi^i_{j_i} \leq \theta_{\varphi(\mathbf{j})},\ \forall\, \mathbf{j} \in [m]^n \right\}.$$

The multipartite implication polytope is defined as its convex hull, i.e.,

$$MPIP(\varphi) := conv(S(\varphi)).$$

Braun et al. (2026) develop a separation routine for valid cuts they term general MPIP inequalities.

Definition 1 (Braun et al. (2026), Definition 2)**.** We call an inequality of the form

$$\sum_{i \in [n]} \sum_{j \in [m]} a^i_j \chi^i_j \leq \sum_{l \in [\kappa]} b_l \theta_l + \omega, \tag{10}$$

a general MPIP inequality if it satisfies the two conditions

$$b_l = \max_{\mathbf{j} \in [m]^n, l = \varphi(\mathbf{j})} \left\{ \sum_{i \in [n]} a^i_{j_i} - \omega \right\} \text{ for all } l \in [\kappa], \text{ and} \tag{aSet}$$

$$a^i_{\max} - a^i_{\min} \leq b_{\max} - b_{\min} \text{ for all } i \in [n]. \tag{bSet}$$

They show that the general MPIP inequalities together with lower bounds are sufficient to completely describe the convex hull of MPIP.

Theorem 1 (Braun et al. (2026), Theorem 3)**.** *Let F be a facet of $MPIP(\varphi)$. Then, either F is induced by a lower bound or by an inequality satisfying* (aSet) *and* (bSet)*.*

Burlacu et al. (2024a) analyze the bipartite implication polytope (BIP), which corresponds to an MPIP with $n = 2$. They define the dependency graph as an undirected graph where each node represents either a set of binary variables that is part of a BIP instance or the BIP instance itself. Each set of implying and implied variables that is part of a BIP instance is connected to the node representing the BIP instance.

Figure 4 shows the dependency graphs for chained multipartite implications within a single cluster.

A key result from Burlacu et al. (2024a) states that if the dependency graph is acyclic, then the convex hull of the resulting chained polytope is exactly the intersection of the convex hulls of the individual bipartite implication polytopes. This result can easily be generalized to the multipartite case.

In our case, however, the dependency graph contains cycles. As a result, separating all valid inequalities from the convex hull descriptions of the individual polytopes does not yield a complete description of the convex hull of the chained polytope. Nevertheless, these inequalities are still valid for the overall polytope and can significantly strengthen the relaxation, thus improving solver performance.

3 Combining Continuous Trajectory Optimization and Discrete Power Profile Optimization

This section presents a concise hybrid framework that integrates a continuous power profile optimization model of choice (CPPO) with the proposed discrete power profile optimization models (DPPO) to align

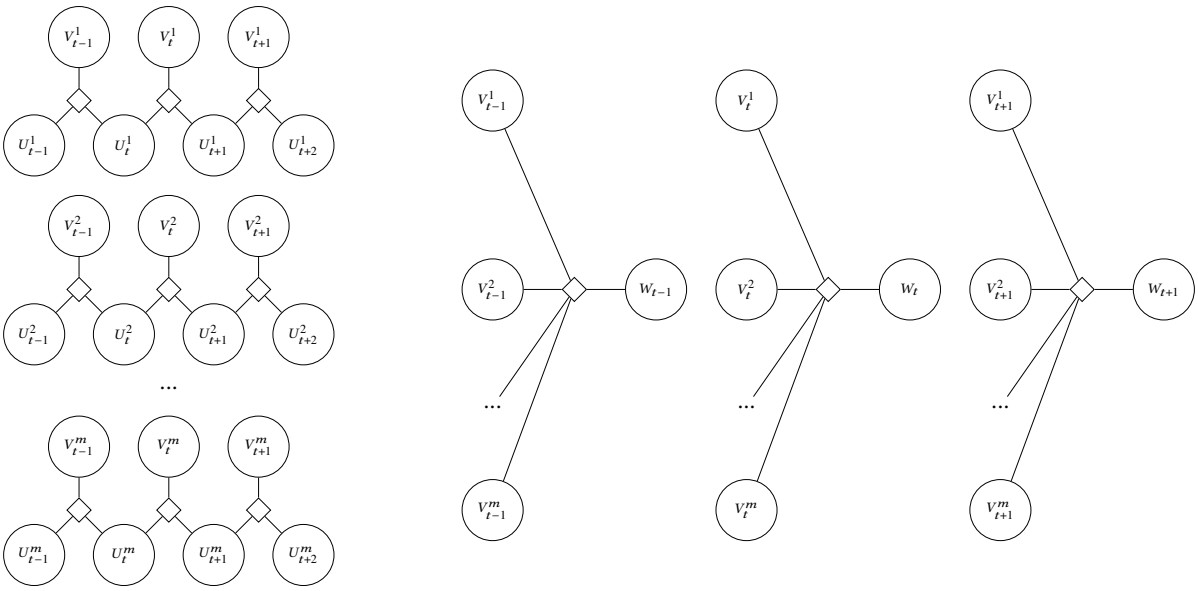

Figure 4: Dependency graphs for the chaining of multipartite implication polytopes across multiple trains.

energy-accurate train dynamics with combinatorial timetabling. While DPPO enables tight integration with the discrete timetable by operating on a trajectory graph, its abstraction of physics limits accuracy. CPPO, in contrast, resolves the nonlinearities of train and network physics at high fidelity. The proposed scheme couples the two: CPPO supplies physics-consistent costs during model construction and validates DPPO solutions during refinement, thereby reconciling discrete scheduling decisions with continuous energy optimization.

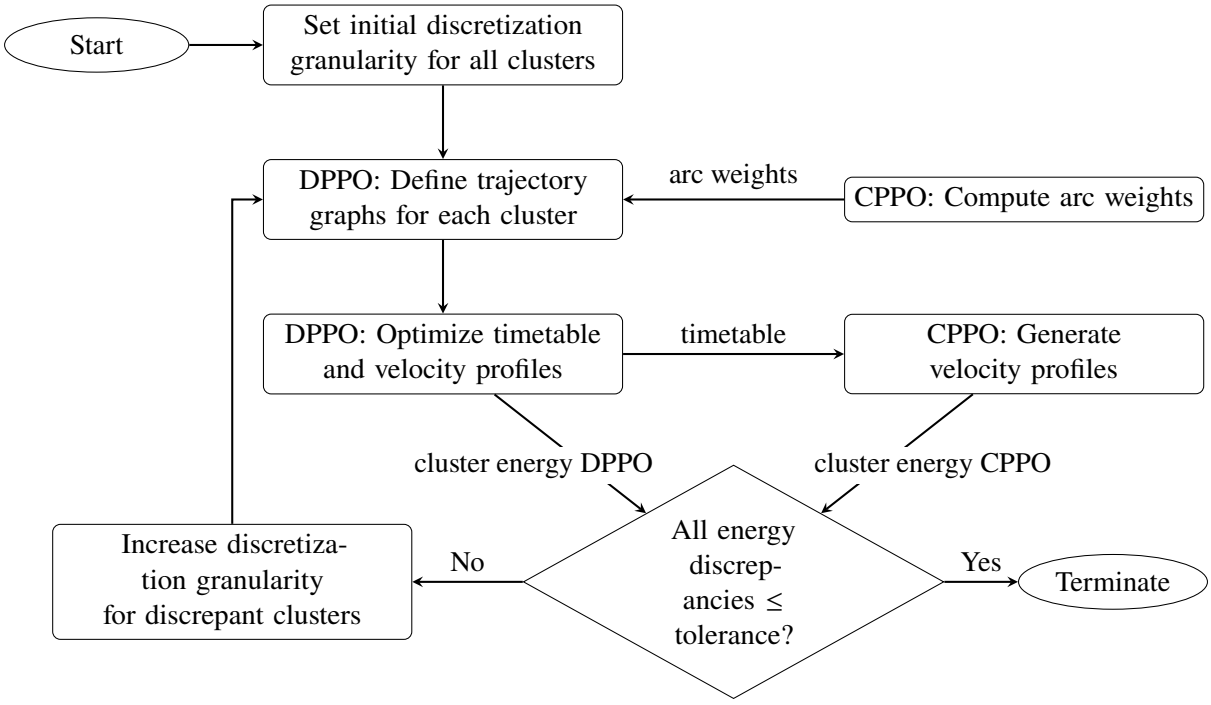

Figure 5: Sequenced operations in adaptive hybrid optimization.

We still assume the network is partitioned into operational clusters that group legs, each leg being one run between two adjacent stations, whenever they share substation areas and overlap in time. Within

a cluster, DPPO states represent global system states, i.e., the joint temporal positions and spatial configurations of all active legs. The DPPO trajectory graph discretizes time, space, and velocity; arcs encode feasible transitions between global states. During model construction, CPPO computes the arc weights by solving continuous boundary-value problems between any pair of DPPO states in the cluster, with boundary times and positions determined by those states. The continuous model can include standard train kinematics, traction and braking characteristics, resistive forces, gradient and curvature effects, substation power-flow and voltage limits, and regenerative power exchange. CPPO accounts for multitrain interactions within the cluster via the shared power network, so that the net electrical energy for each transition is computed in a way that is consistent across both single-leg and multi-leg transitions. These energies define the arc costs in the DPPO trajectory graphs and establish the physical cost structure used in the discrete optimization.

In the solution phase, DPPO is solved with a pure energy-minimization objective to produce a timetable: for each cluster, a mapping from leg identifiers to optimized departure and arrival times, together with the selected track metadata (identification, length, and elevation/curvature profiles). Given this timetable, CPPO is invoked again per cluster to compute a continuous velocity profile solution. For each leg, CPPO returns time-indexed velocity and position trajectories that satisfy the boundary times and positions from the timetable while enforcing the cluster's power-network constraints and allowing regenerative sharing among simultaneous legs. Let $E^{DPPO}(C)$ denote the DPPO energy in cluster C and $E^{CPPO}(C)$ the corresponding CPPO energy under the validated continuous trajectories. The two are compared via a relative discrepancy test, $\left|\frac{E^{CPPO}(C)-E^{DPPO}(C)}{E^{CPPO}(C)}\right| \leq \epsilon$, where $\epsilon > 0$ is a prescribed tolerance. When a cluster violates this test, its DPPO discretization is locally refined by tightening the resolutions Δt, Δd, and Δv used to construct the cluster's trajectory graph. The affected arc weights are then recomputed by CPPO under the refined grid, DPPO is re-optimized globally with the updated costs, and CPPO revalidates the new timetable. This adaptive loop is applied only to clusters that exceed the tolerance and terminates once all clusters satisfy the energy consistency test or an iteration cap is reached. In practice, this procedure substantially reduces the discretization-induced modeling error while limiting computational effort by focusing refinement where it is most needed. The workflow is illustrated in Figure 5.

Because clusters are power-wise decoupled by construction, CPPO solves are conducted per cluster with the DPPO timetable imposing boundary conditions; nonetheless, within each cluster, the multitrain coupling through the shared substation is fully enforced. Overall, the hybrid approach ensures that the discrete timetable is informed by and consistent with the underlying continuous physics, and that the resulting energy estimates are aligned across modeling layers.

4 Computational Study

In this section we evaluate the performance of four model formulations: NDBE, ND, LOGIC, and MPIP. All four address the railway energy management problem through integrated timetabling and power profile optimization, but they differ in the way the problem is modeled and solved. The complete mathematical formulations are given in Section 2.

4.1 Implementation Environment

The computational experiments are based on real-world timetabling instances provided by VAG (Verkehrs-Aktiengesellschaft Nürnberg), the operator of the underground railway network in Nuremberg, Germany. Each instance represents a planning horizon of 300 seconds during which trains operate across the network. Within this time window we distinguish two types of train legs. Legs that are already in operation at the start of the planning horizon keep their trajectories fixed in order to ensure operational continuity, while legs that have not yet commenced may be shifted in time by up to ±10 seconds relative to their scheduled departure times.

We evaluate the models on 12 configurations. Each configuration is described by a triple `(d, v, n)`, where $\mathtt{d} \in \mathbb{N}$ is the number of discretization points along the spatial dimension of the trajectory graph, $\mathtt{v} \in \mathbb{N}$ is the number of discretization points in the velocity dimension, and $\mathtt{n} \in \mathbb{N}$ is the maximum number of legs that are permitted to exchange energy simultaneously within a single cluster. The parameter `n` controls the problem complexity: whenever a cluster would contain more than `n` legs exchanging energy at the same time, it is partitioned into several smaller clusters, each respecting the cardinality constraint. The temporal discretization is fixed at 5-second intervals in all configurations so that the comparison remains fair. For each of the 12 configurations we generate 10 independent problem instances, which results in 120 instance-configuration pairs per model.

Model performance is measured by three metrics. First, we report computational time, separated into the model construction time, that is the time required to build the mathematical programming formulation, and the solver runtime, that is the time spent by the optimization solver. Second, we report the optimality gap at termination, defined as the relative difference between the best incumbent solution value UB and the dual bound LB, gap $= \frac{UB-LB}{UB}$. Third, we report the relative improvement, that is the percentage reduction in objective function value compared to a baseline non-optimized reference timetable. A computational limit of 3600 seconds is enforced for the model construction and for the solving process separately. If the model construction alone exceeds this limit for a given model-configuration pair, the corresponding data point is excluded from the analysis.

All experiments were carried out on a virtual machine with an Intel Xeon processor with 28 cores at 2.70 GHz base frequency and 64 GB of RAM. We used Gurobi Optimizer version 12.0.1 as mixed-integer programming solver with default parameter settings and the number of parallel threads limited to 4. All models and the data processing were implemented in Python 3.10.16.

4.2 Results Analysis

Figure 6 shows the shifted geometric mean of the total runtime over all instances for each model and configuration. The observed runtime patterns reflect the structural and algorithmic characteristics of the individual modeling approaches.

If energy exchange is limited to at most two legs simultaneously, the formulations based on the multi-leg trajectory graph are clearly faster. For configuration `20.4.2`, the ND model attains the shortest geometric mean runtime with 28.40 seconds, and the NDBE decomposition follows closely with 29.13 seconds, which shows that the decomposition introduces only minimal overhead in this regime. The logic-based formulations need considerably more time: LOGIC requires 198.10 seconds and MPIP requires 832.89 seconds. The reason for this gap is that with limited coupling the multi-leg trajectory

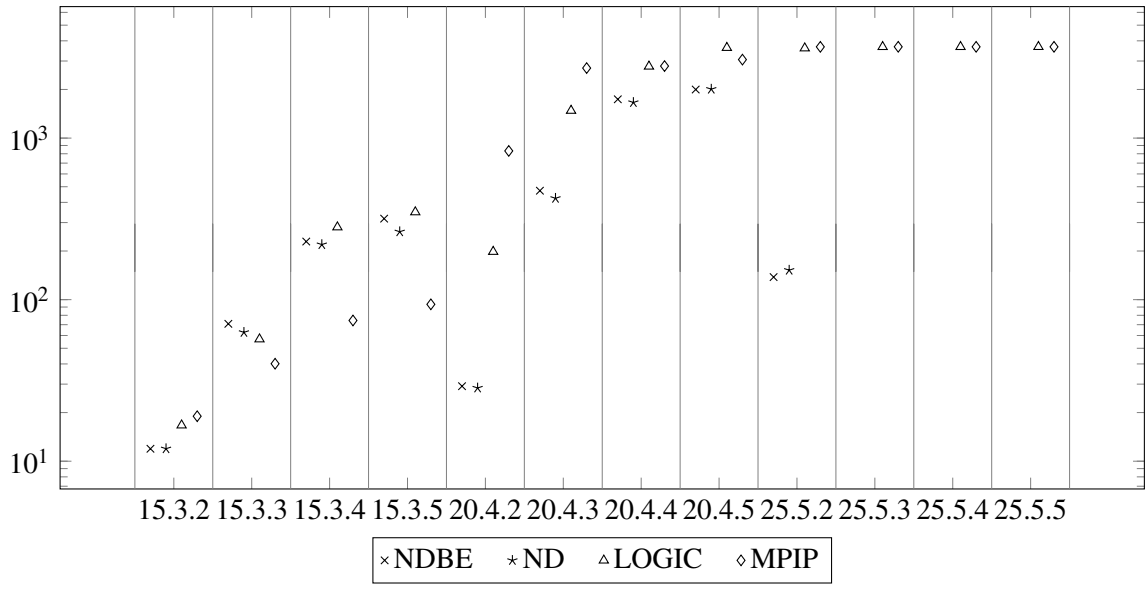

Figure 6: Shifted Geometric Mean of Total Time (seconds) by Model and Configuration

graph stays small enough for the shortest path computations to be very efficient, whereas the logic-based models have to handle a large number of binary variables and logical constraints even when the underlying combinatorial structure is simple.

When the coupling parameter increases to $\mathtt{n} = 3$, the relative performance of the models changes qualitatively. The scalability of the logic-based formulations becomes visible, and MPIP performs particularly well. In configuration `15.3.5`, for example, LOGIC, NDBE, and ND all have geometric mean runtimes above 260 seconds, while MPIP finishes in approximately 95 seconds. This reversal is explained by the growth of the multi-leg trajectory graph with increasing coupling: the number of nodes and edges scales as $O((\mathtt{d} \cdot \mathtt{v})^{\mathtt{n}})$ for $\mathtt{n}$ interacting legs, which quickly exhausts memory and computing resources. The logic-based models scale more moderately because they represent interactions through logical constraints instead of explicit graph enumeration.

For large and tightly coupled configurations such as `25.5.3`, `25.5.4`, and `25.5.5`, the multi-leg trajectory graph becomes intractable. We report no results for NDBE and ND on these configurations, since the model construction phase alone exceeded the one-hour time limit in all instances. Only LOGIC and MPIP remain applicable here, and MPIP is consistently faster than LOGIC because of its separation-based cutting plane procedure.

Figure 7 shows the model construction time alone and identifies the main computational bottleneck of the trajectory graph-based approaches. The construction of the multi-leg trajectory graph is both memory-intensive and time-consuming, since every additional leg in the coupling structure multiplies the graph size and the computation of the edge weights requires the solution of a large number of subproblems that determine the feasible transitions between trajectory states.

These results suggest a regime-dependent model selection. For small and medium-scale instances with low leg coupling, that is $\mathtt{n} \leq 2$, NDBE and ND are computationally more efficient and should be preferred. Since they produce high-quality solutions quickly, they are well suited for operational planning scenarios with short turnaround times. For large instances with high coupling, that is $\mathtt{n} \geq 3$, the logic-based formulations LOGIC and MPIP are the only tractable option. The separated multipartite

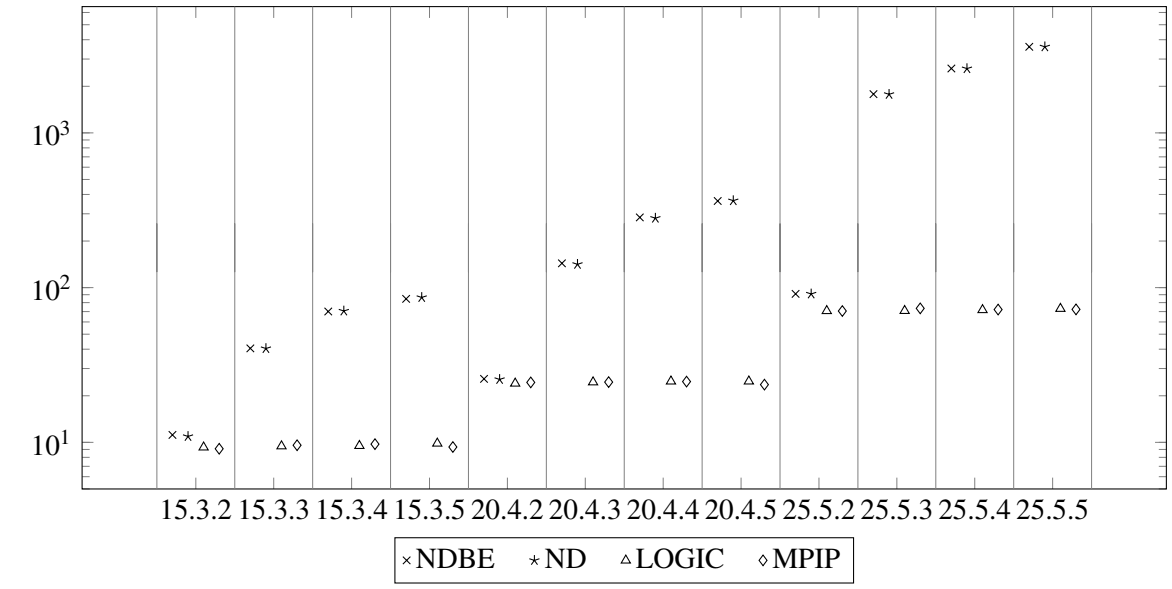

Figure 7: Shifted Geometric Mean of Buildup Time (seconds) by Model and Configuration

implication cuts in MPIP become more valuable as the coupling complexity grows, since they approximate the convex hull of the feasible power exchange patterns without requiring an explicit enumeration of the whole trajectory space.

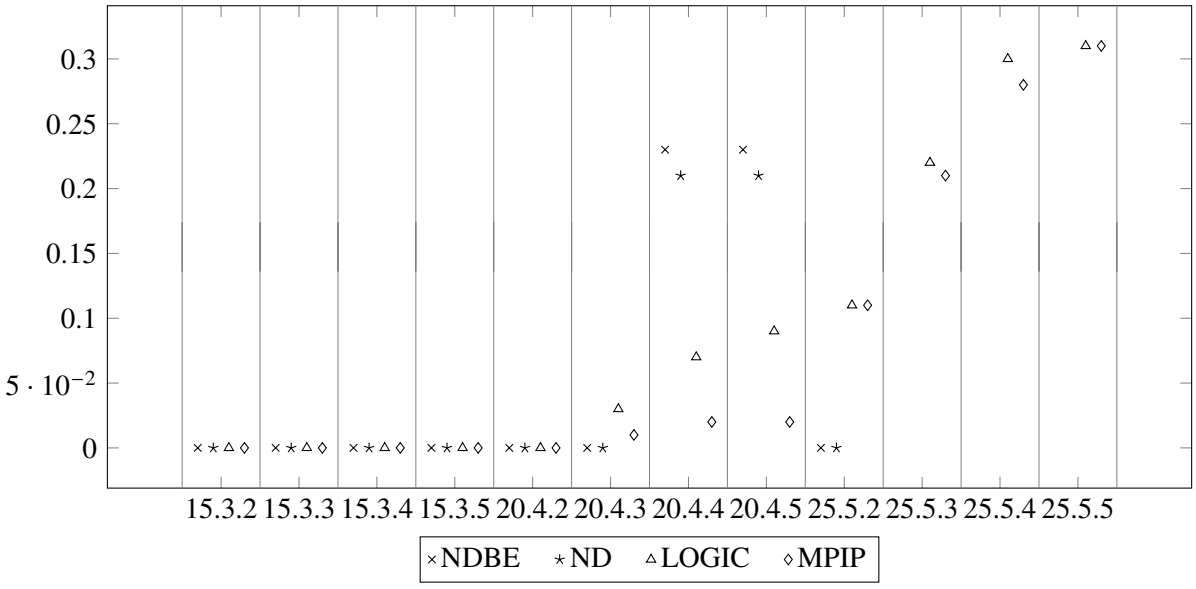

Figure 8: Mean Optimality Gap by Model and Configuration

Figure 8 reports the optimality gaps of all models over all configurations. This metric describes the solution quality in particular for those instances in which the time limit is reached before optimality can be proven. In configurations in which none of the models proves optimality within the time limit, MPIP consistently attains the smallest optimality gaps, which indicates better dual bounds. In configuration `20.4.5`, LOGIC terminates with an optimality gap of 9.34%, whereas the separation routine in MPIP reduces the gap to 2.08% on the same configuration. This shows that the cutting plane algorithm tightens the linear programming relaxation by cutting off fractional solutions that violate the multipartite implication structure. The trajectory graph-based models NDBE and ND have clearly weaker dual bounds

on these configurations. On `20.4.5`, NDBE reports a gap of 22.96% and ND a gap of 21.19%.

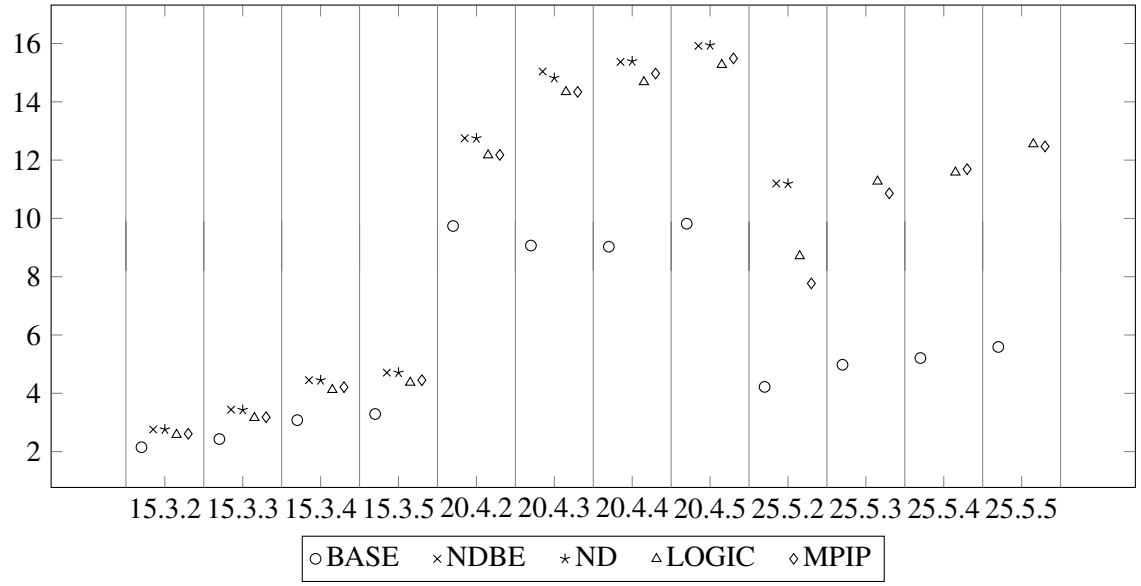

Figure 9: Mean Improvement by Model and Configuration

Figure 9 quantifies the practical value of the models in terms of the energy savings achieved relative to the non-optimized baseline timetable, which translates directly into operational cost savings and reduced environmental impact. We use two reference points for the comparison: a completely non-optimized timetable that represents the current operational practice without any optimization, and BASE, a scheduling-only optimization model that optimizes departure times without adjusting power profiles. The comparison with BASE isolates the additional benefit of the integrated power profile optimization.

A further observation concerns the solution quality across model types. Whenever optimal solutions are found within the time limit, the trajectory graph-based models ND and NDBE yield solutions with larger energy savings than the logic-based models LOGIC and MPIP. This is caused by the discretization approximation in the logic-based models. Both approaches discretize the state space, but the trajectory graph optimizes explicitly over feasible continuous trajectories within each discrete cell, while the logic-based models use a coarser aggregation of the energy consumption within each time-space cell. In configuration `15.3.5`, where all models find high-quality solutions, ND attains an improvement of 4.71% over the baseline, while MPIP attains a slightly lower improvement of 4.45%.

The results also show a positive correlation between the fineness of the discretization and the solution quality, as long as the model remains tractable. Increasing `d` and `v` consistently leads to larger energy savings whenever the resulting model can still be solved.

The largest energy savings are obtained by ND in configuration `20.4.5` with a reduction of 15.94% compared to the completely non-optimized baseline timetable, which corresponds to substantial operational cost savings and a reduction of carbon emissions. Relative to the reference timetable that is already optimized for scheduling, the largest improvement in a realistic setting with $\mathtt{n} = 5$ is obtained by LOGIC in configuration `25.5.5` with 7.37% on average.

The computational evaluation shows that each formulation occupies a distinct niche, with clear trade-offs between solution quality, computational efficiency, and scalability. The trajectory graph-based models NDBE and ND perform well on small and medium-scale problems with limited leg interaction,

that is $\mathtt{n} \leq 2$. In this regime they deliver better solution quality, due to their precise representation of the trajectory dynamics, and they are considerably faster than the logic-based alternatives. The ND model is the integrated formulation without decomposition and results in a large but structured mixed-integer program. The NDBE variant applies Benders decomposition in order to separate the timetabling decisions from the power profile optimization, which allows a parallel solution of the subproblems and an integration into larger hierarchical planning frameworks. Both trajectory graph-based models, however, share one critical limitation, namely that their model construction phase scales exponentially with the coupling complexity. As a consequence, they do not produce any solution for high-coupling configurations.

The logic-based models LOGIC and MPIP trade solution accuracy for robustness and scalability. By aggregating the energy dynamics within discretized space-time-velocity cells and by representing interactions through logical constraints instead of an explicit trajectory enumeration, they scale much more favorably in terms of memory and construction time. The model size grows approximately linearly in $\mathtt{n}$, which makes instances solvable that are completely intractable for the trajectory graph methods. The MPIP model extends LOGIC by the dynamic separation of strengthened inequalities derived from the multipartite implication polytope. These cuts approximate faces of the convex hull of the feasible power exchange patterns and thereby tighten the formulation considerably. In high-coupling configurations, where second-level multipartite implications dominate the problem complexity, the separation algorithm is very effective. This benefit is not universal, though. In configurations with sparse first-level implication structures, that is with simple train dynamics and low coupling, the cutting plane algorithm may cause more computational overhead than it saves.

From the computational study we derive the following practical guidance for the model selection. For instances with coupling parameter $\mathtt{n} \leq 2$ and discretization parameters satisfying $\mathtt{d} \cdot \mathtt{v} \leq 100$, ND or NDBE should be used in order to obtain the best solution quality together with short computation times, where NDBE is preferable if an integration into a larger hierarchical planning system is intended. For instances with $3 \leq \mathtt{n} \leq 4$, preliminary tests of the model construction time should be carried out: if ND or NDBE can be built within the given time limit, they should be used because of their better solution quality, and otherwise MPIP should be used. For instances with $\mathtt{n} \geq 5$ or $\mathtt{d} \cdot \mathtt{v} > 100$, only MPIP should be used, since the trajectory graph methods are intractable and the approximation error relative to the exact trajectory optimization is acceptable given that the alternative is no solution at all. Finally, if the solution time is severely restricted, for example in real-time operational settings that require solutions within minutes, the logic-based models should be preferred independently of the problem size, since their runtime is more predictable.

4.3 Algorithmic Implications of the Discrete-Continuous Approach

The computational results above show a distinct trade-off between the multi-leg trajectory graph-based models NDBE and ND and the logic-based formulations LOGIC and MPIP. In order to combine the strengths of both approaches in a hybrid setting as described in Section 3, we propose an iterative refinement algorithm that selects the model dynamically according to the precision required in the different parts of the problem.

The algorithm starts from the NDBE decomposition framework and initializes the problem with a coarsely discretized trajectory grid for all leg clusters, which exploits the short solution times of the graph-based models on less complex scenarios. In each iteration the Benders master problem is solved in

order to obtain a candidate timetable. For each cluster the power profiles are then optimized for this fixed timetable, and two methods are executed in parallel: the discrete subproblem DPPO is solved as a shortest path problem in the multi-leg trajectory graph of the cluster $C \in \mathcal{C}$, which yields $E^{DPPO}(C)$, and the continuous power profile optimization method is used to compute the more accurate energy consumption $E^{CPPO}(C)$.

The results of these two computations are then compared. If the discrepancy lies within a predefined tolerance, the coarse discretization is a sufficiently accurate approximation of the dynamics of that cluster under the given timetable, and a standard Benders optimality cut is generated from the energy value of the discrete subproblem and added to the master problem.

If the two energy values differ significantly, the coarse grid does not capture the energy dynamics of the cluster accurately enough. A simple refinement of the grid would lead to a combinatorial explosion of the multi-leg trajectory graph and, as the computational study shows, would most likely cause the NDBE model to exceed the memory limits. Therefore the algorithm changes the modeling strategy for this particular cluster and replaces its Benders subproblem by the more memory-efficient logic-based formulation MPIP, which can handle the finer and more complex grid. Since this formulation does not support Benders decomposition in the same way, the power profile optimization of this cluster is integrated directly into the master problem. The master problem therefore becomes a hybrid model that contains Benders cuts for the well-approximated clusters and the full logic-based model for those clusters that require a higher fidelity. This process of solving, validating, and selectively refining is repeated until convergence and thus balances the required accuracy against computational tractability. The overall workflow is given in Algorithm 1.

Algorithm 1 Hybrid Iterative Refinement Algorithm

Require: Set of clusters $\mathcal{C}$, master problem Model Master, tolerance ϵ
Ensure: An optimized timetable DC

$\mathcal{C}^B \leftarrow \mathcal{C}$
while not Benders algorithm converged for each $C \in \mathcal{C}^B$ **do**
    $DC \leftarrow$ Solve master problem
    for each $C \in \mathcal{C}^B$ **do**
        $E^{DPPO}(C) \leftarrow$ Solve Model Sub for C
        $E^{CPPO}(C) \leftarrow$ Solve continuous optimization
        if $|E^{DPPO}(C) - E^{CPPO}(C)| < \epsilon$ **then**
            Add Benders cut
        else
            $\mathcal{C}^B = \mathcal{C}^B \setminus C$
            Refine trajectory grid for C
            Integrate Model 9 for C into master problem
        end if
    end for
end while
    return DC

5 Conclusion

We presented mixed-integer formulations and algorithms that optimize railway timetabling and vehicle power profiles at the same time, with the aim of improving the energy efficiency of an underground network.

We developed a discrete power profile optimization (DPPO) framework and proposed two solution methods for it. The first method is graph-based. It uses a multi-leg state-trajectory graph and was implemented in two ways: as a single network flow model (ND) and as a Benders decomposition scheme (NDBE). The second method uses logic-based integer programming formulations (LOGIC and MPIP). These formulations represent the system state transitions through multipartite implication constraints. In this way, they avoid building the state graph explicitly, which can become very large.

We evaluated the proposed models in a computational study based on real operational data from the Nuremberg underground network. The results show that the graph-based models (ND and NDBE) perform best on instances with low energy coupling between train legs. However, their memory requirements become too high for larger instances with many simultaneous vehicle interactions. The logic-based models (LOGIC and MPIP) scale much better. They solved highly coupled instances for which the graph-based models ran out of memory. Adding separation cuts based on the facial structure of the multipartite implication polytopes to the MPIP model was especially useful. These cuts strengthen the linear programming relaxation and speed up convergence in the difficult instances. Within the discretized energy model, the computed solutions consume up to 7% less energy than a timetable that is already optimized but does not consider power profiles, and up to 16% less than a non-optimized baseline timetable. These numbers quantify what the additional degrees of freedom are worth inside the model. They are not measured savings, since the energy of a solution is evaluated with the same point-mass model that is used to build the arc costs.

This is the main limitation of the present work, and it defines the two next steps. The first is the energy model itself. Both formulations depend on the physics only through the cost of a state transition, and we described in Section 3 which quantities a continuous trajectory optimization model would have to supply to replace our point-mass costs, and how it could be used to re-evaluate a discrete solution. Implementing such a model, including track gradients, speed limits and a power network model with voltage-dependent losses, and repeating the computational study with these costs, will show how the reported savings translates into a more detailed representation. The second step is validation in operation. The formulations are set up so that they can be used on the planning data of any underground train system with no storages, and the natural continuation is a comparison of measured traction energy for an optimized and a reference timetable in a real-world setting.

A Notation

Symbol	Description
Sets and Indices	

Continued on next page

Table 1 continued from previous page

Symbol	Description
$\mathcal{K}$	Set of all train legs
k	Leg identifier/index
$\mathcal{D}_k$	Set of feasible departure configurations for leg k
$\mathcal{C}$	Set of clusters (groups of electrically coupled legs)
C	Cluster of legs
$T(C)$	Time horizon of cluster C
$T'(C)$	Time horizon of cluster C excluding the final timestep
t	Discrete time index
d	Discretized distance coordinate along a route
v	Discretized velocity level
u, w	Nodes in the trajectory graph
$G = (V, A)$	Single-leg arborescence (node set, arc set)
$\mathcal{G} = (\mathcal{V}, \mathcal{A})$	Multi-leg (cluster) arborescence
$V_t^k(C)$	Nodes for leg k at time t in cluster C
$A_t^k(C)$	Arcs for leg k at time t in cluster C
$\mathcal{V}_t(C)$	Cluster-level nodes at time t
$\mathcal{A}_t(C)$	Cluster-level arcs at time t
$\mathcal{A}_t^{\nearrow}(C)$	Arcs at time t where a leg accelerates from rest
$\mathcal{A}_t^{\searrow}(C)$	Arcs at time t where a leg brakes to rest
$W_t^k(C)$	Leg-level energy consumption groups for leg k at time t
W	Set of all leg-level energy consumption groups
$Z_t(C)$	Network-level energy consumption groups at time t
Z	Set of all network-level energy consumption groups
$\Omega_t(C)$	Set of leg-level group combinations at time t
Timetabling	
$x_{k\sigma\rho}$	Binary: leg k departs at time σ with running time ρ
X	Set of all feasible timetables
(σ, ρ)	Departure configuration (departure time, running time)
$\xi(\mathbf{x})$	Total energy consumption of timetable $\mathbf{x}$
c_C	Energy cost of cluster C
$Sub_C(\mathbf{x})$	Energy-optimization subproblem for cluster C given timetable $\mathbf{x}$
Δ_k	Final distance (route length) of leg k
Trajectory Graph and Flow Model	
$\Delta t,\ \Delta d,\ \Delta v$	Discretization resolutions (time, distance, velocity)
$\zeta(\cdot)$	Function mapping each arc to the leg that accelerates from or brakes to rest
ψ_u	Binary: node u is selected in the path
Y_{uw}	Flow variable on arc (u, w)
p_{uw}	Energy required to transition from state u to state w

Continued on next page

Table 1 continued from previous page

Symbol	Description
$0,\ \bar{t}$	Source and sink time indices
Benders Decomposition	
π_u	Dual variable of the flow-conservation constraint at node u
$\alpha,\ \beta$	Dual variables of the source / sink constraints
μ_{uw}	Dual variable of the blocking constraint on arc (u, w)
$\Gamma(\mathbf{x})$	Cut term aggregating the blocking-constraint duals
$\mathbf{x}^0$	Interior point of the convex hull of X
$\bar{\mathbf{x}}$	Fixed master timetable solution
Logic-Based Model (Bipartite/Multipartite Implications)	
v	Index of a leg-level energy consumption group
ℓ	Index of a network-level energy consumption group
y_v	Binary: leg-level energy group v is active
z_ℓ	Binary: network-level energy group ℓ is active
q_ℓ	Energy cost of network-level group ℓ
$\Xi(\cdot)$	Mapping from arcs to leg-level energy consumption groups
$\Theta(\cdot)$	Mapping from combinations of leg-level groups to network-level groups
$\boldsymbol{\lambda}$	Combination of leg-level energy consumption groups
$S^{TRAIN}(C, k, t)$	Feasible set of the train-trajectory bipartite implications
$S^{POWER}(C, t)$	Feasible set of the power-grid multipartite implications
Multipartite Implication Polytope (MPIP)	
n	Number of implying sets
m	Cardinality of each implying set
κ	Cardinality of the implied set
$[m]$	Index set $\{1, \ldots, m\}$
$\mathbf{j} = (j_1, \ldots, j_n)$	Index vector over implying sets
χ^i_j	Binary: element j is chosen in the i-th implying set
θ_l	Binary: element l is chosen in the implied set
$\varphi(\cdot)$	Implication mapping
M	Set of binary points satisfying all multiple-choice constraints
$S(\varphi)$	Feasible set restricted by the implication mapping
$MPIP(\varphi)$	Multipartite implication polytope: $\mathrm{conv}(S(\varphi))$
a^i_j	Coefficients for implying-set variables in general MPIP inequalities
b_l	Coefficients for implied-set variables in general MPIP inequalities
ω	Constant term in general MPIP inequalities
F	A facet of the polytope
Hybrid Framework	

Continued on next page

Table 1 continued from previous page

Symbol	Description
$E^{DPPO}(C)$	Energy of cluster C from the discrete model
$E^{CPPO}(C)$	Energy of cluster C from the continuous model
ϵ	Relative discrepancy tolerance
Computational Study	
`(d, v, n)`	Configuration triple: spatial points, velocity points, max coupled legs
UB	Best incumbent objective value
LB	Dual lower bound
Model Names	
DPPO	Discrete power profile optimization
CPPO	Continuous power profile optimization
ND	Network design formulation
NDBE	Network design with Benders decomposition
LOGIC	Logic-based formulation
MPIP	Logic-based formulation with MPIP separation
BASE	Scheduling-only optimization baseline

References

T. Albrecht, P. Howlett, P. Pudney, and A. Albrecht. An optimal train journey with bounds on energy consumption during specified intermediate time intervals. *Journal of Rail Transport Planning & Management*, 27:100391, 2023. doi: 10.1016/j.jrtpm.2023.100391.

Kristin Braun, Robert Burlacu, Tobias Kuen, and Kristina Rolsing. Modeling binary relations in piecewise-linear approximations, 2026. URL `https://optimization-online.org/?p=33773`. Optimization Online preprint.

Robert Burlacu, Patrick Gemander, and Tobias Kuen. The Bipartite Implication Polytope: Conditional relations over multiple sets of binary variables, 2024a. URL `https://optimization-online.org/?p=26208`. Optimization Online preprint.

Robert Burlacu, Patrick Gemander, Tobias Kuen, and Jorge Weston. Energy-efficient timetables for railway traffic: Incorporating DC power models, 2024b. URL `https://optimization-online.org/?p=27585`. Optimization Online preprint.

V. Cacchiani and P. Toth. Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737, 2012. doi: 10.1016/j.ejor.2011.11.003.

Patrick Gemander, Andreas Bärmann, and Alexander Martin. A stochastic optimization approach to energy-efficient underground timetabling under uncertain dwell and running times. *Transportation Science*, 57(6):1627–1645, 2023. doi: 10.1287/trsc.2022.0269.

Benno Hoch and Frauke Liers. An integrated rolling horizon and adaptive-refinement approach for disjoint trajectories optimization. *Optimization and Engineering*, 24:1017–1055, 2023. doi: 10.1007/s11081-022-09719-2.

Thomas L. Magnanti and Richard T. Wong. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981. doi: 10.1287/opre.29.3.464.

Thomas L. Magnanti, P. Mireault, and Richard T. Wong. Tailoring Benders decomposition for uncapacitated network design. In Giorgio Gallo and Claudio Sandi, editors, *Netflow at Pisa*, pages 112–154. Springer, Berlin, Heidelberg, 1986. doi: 10.1007/BFb0121090.

Robin H. Pearce and Michael Forbes. Disaggregated Benders decomposition and branch-and-cut for solving the budget-constrained dynamic uncapacitated facility location and network design problem. *European Journal of Operational Research*, 270(1):78–88, 2018. doi: 10.1016/j.ejor.2018.03.021.

Maite Peña-Alcaraz, Antonio Fernández, Asunción Paloma Cucala, Andrés Ramos, and Ramón R. Pecharromán. Optimal underground timetable design based on power flow for maximizing the use of regenerative-braking energy. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, 226(4):397–408, 2012. doi: 10.1177/0954409711429411.

Xin-Chen Ran, Shao-Kuan Chen, Ge-Hui Liu, and Yun Bai. Energy-efficient approach combining train speed profile and timetable optimisations for metro operations. *IET Intelligent Transport Systems*, 14(14):1967–1977, 2020. doi: 10.1049/iet-its.2020.0346.

G. M. Scheepmaker, R. M. P. Goverde, and L. G. Kroon. Review of energy-efficient train control and timetabling. *European Journal of Operational Research*, 257(2):355–376, 2017. doi: 10.1016/j.ejor.2016.09.044.

Angelo Sifaleras. Minimum cost network flows: Problems, algorithms, and software. *Yugoslav Journal of Operations Research*, 23(1):3–17, 2013. doi: 10.2298/YJOR121120001S.

Pengling Wang and Rob M. P. Goverde. Multi-train trajectory optimization for energy-efficient timetabling. *European Journal of Operational Research*, 272(2):621–635, 2019. doi: 10.1016/j.ejor.2018.06.034.

Songpo Yang, Jianjun Wu, Xin Yang, Huijun Sun, and Ziyou Gao. Energy-efficient timetable and speed profile optimization with multi-phase speed limits: Theoretical analysis and application. *Applied Mathematical Modelling*, 56:32–50, 2018. doi: 10.1016/j.apm.2017.11.017.

Songwei Zhu, Yihui Wang, Guodong Wei, Yi Zheng, Datian Zhou, and Nikola Bešinović. Integrated optimization of energy-efficient train timetable and rolling stock circulation plan with regenerative energy utilization. *Journal of Rail Transport Planning & Management*, 33:100499, 2025. doi: 10.1016/j.jrtpm.2024.100499.