https://doi.org/10.3390/futuretransp

Francisco Vando Carneiro Moreira; Plácido Rogério Pinheiro; Carolina Ferreira Araujo

MDPI

Article

A Hybrid Matheuristic Approach for Long-Horizon Driver Scheduling under European Union Social Regulations

Francisco Vando Carneiro Moreira [1], Plácido Rogério Pinheiro [1] and Carolina Ferreira Araujo [2]

[1] University of Fortaleza (UNIFOR), Fortaleza, CE, Brazil;
[2] Aberta University, Lisbon, Portugal.
* Correspondence: vando.moreira@edu.unifor.br; Placido@unifor.br

Abstract

Driver scheduling in freight road transport under European regulations constitutes a complex combinatorial optimization problem, especially when simultaneously considering legal, operational, and temporal demand constraints. Based on a Mixed-Integer Linear Programming (MILP) model that contains the comprehensive legislative requirements of Regulation (EC) No 561/2006, this study proposes a decision-support framework for European road transport driver scheduling. Beyond an exact resolution mode using the OR-Tools CP-SAT solver, the framework incorporates a matheuristic strategy based on Large Neighborhood Search (LNS), designed to overcome the "scalability barrier" observed in long-horizon scenarios (e.g., 15 days). Computational experiments demonstrate that while exact solvers ensure optimality for daily planning, the hybrid matheuristic approach provides robust, high-quality, and legally compliant solutions for tactical medium-term planning in operationally feasible times. Furthermore, the results indicate that this integrated approach offers a powerful tool for strategic, tactical, and operational decision-making in real-world transport operations.

Keywords: Integer Linear Programming; Driver Scheduling; Regulation (EC) No 561/2006; Optimization; OR-Tools; Matheuristics; Large Neighborhood Search.

1. Introduction

Road freight transport is a fundamental pillar of modern logistics, ensuring the movement of goods and maintaining supply chain stability across the globe, particularly within the European Union (EU). According to Eurostat data, road transport accounts for approximately 75% of inland freight transport in the EU, representing over 1.8 trillion tonne-kilometres annually [1]. The efficiency of this mode of transport depends heavily on the effective management of its primary resource: the truck drivers. Managing their working hours is a critical factor for safety, efficiency, and legal compliance.

In recent years, the EU has implemented some of the world's most stringent social regulations regarding driving times, breaks, and rest periods for professional drivers. Regulation (EC) No 561/2006, complemented by Directive 2002/15/EC and Regulation (EU) No 165/2014, establishes strict limits intended to protect driver health, prevent fatigue-related accidents, and promote fair competition [2,3]. These rules directly impact how driver schedules are constructed, requiring constant attention to daily, weekly, and fortnightly requirements. Moreover, non-compliance can result in severe penalties, including fines up to €30,000 per violation and potential suspension of operating licenses.

Received:
Accepted:
Published:

However, manual scheduling is highly prone to errors, especially in large-scale operations characterized by high demand variability, multiple time windows, and cumulative temporal dependencies [4]. Small decisions made in one period can invalidate the entire future schedule, leading to legal infractions and operational risks. Furthermore, the combinatorial nature of the problem grows exponentially with the planning horizon, making intuitive approaches impractical for medium to long-term planning. Mathematical optimization approaches based on Mixed-Integer Linear Programming (MILP) have emerged as suitable tools to formalize and solve these problems, offering the precision needed to handle dozens of simultaneous constraints [5,6].

While exact solvers like OR-Tools CP-SAT provide optimal solutions for short horizons, they often face a "scalability barrier" in long-horizon instances due to the exponential growth of the search space [7,8]. Additionally, the integration of cumulative constraints (weekly and fortnightly limits) creates tight coupling between consecutive periods, significantly increasing computational complexity. To address this challenge, hybrid matheuristic approaches combining exact methods with metaheuristics have shown promising results in related scheduling problems [9,10].

Consequently, this research proposes an integrated framework that combines exact optimization with a hybrid matheuristic based on Large Neighborhood Search (LNS). The goal is to provide a decision-support tool capable of generating 100% legally compliant and operationally efficient schedules for both short (24h) and medium-term (15d) horizons. The main contributions of this work include: (i) a comprehensive MILP formulation incorporating all constraints from Regulation 561/2006; (ii) a specialized LNS matheuristic designed to maintain legal compliance while exploring solution improvements; (iii) extensive computational validation demonstrating the scalability barrier and the effectiveness of the hybrid approach; and (iv) practical insights for real-world implementation in freight transport operations.

The remainder of this paper is organized as follows: Section 2 provides a comprehensive literature review covering foundational work, mathematical modeling approaches, and recent advances in driver scheduling; Section 3 details the MILP model formulation and the hybrid matheuristic algorithm; Section 4 presents the experimental results and scalability analysis across multiple time horizons; Section 5 discusses the implications and practical considerations; and finally, Section **??** concludes with final remarks and directions for future work.

2. Literature Review

The Driver Scheduling Problem (DSP), also referred to as crew scheduling in some contexts, represents a classical area of Operations Research characterized by its NP-hard computational complexity [1,11]. In this section, we review the relevant literature organized into five major categories: foundational work, mathematical modeling approaches, solution methodologies, European regulation-specific studies, and recent advances.

2.1. Foundational Work and Historical Context

The systematic study of driver and crew scheduling problems began in the early 1980s with the seminal work of Bodin et al. [1], who provided a comprehensive state-of-the-art review of routing and scheduling of vehicles and crews. Their work established the fundamental mathematical frameworks, particularly the Set Covering and Set Partitioning formulations, which remain influential in modern approaches. The authors identified the key challenge of integrating routing decisions with crew assignments while respecting operational and regulatory constraints.

Subsequently, Wren and Rousseau [12] focused specifically on public transport, providing an overview of computer-aided bus driver scheduling. Their work highlighted the importance of break and relief opportunities, shift patterns, and labor agreements. Methodologically, they emphasized column generation techniques and constraint-driven approaches. Martello and Toth [13] contributed with a heuristic approach specifically designed for the bus driver scheduling problem, demonstrating that constructive heuristics could provide near-optimal solutions for medium-scale instances within acceptable computational times.

Ernst et al. [11] provided an extensive annotated bibliography on workforce planning and rostering, covering over 200 references across various industries including transportation, healthcare, and manufacturing. Their taxonomy distinguished between tactical (long-term) and operational (short-term) scheduling, a distinction that remains relevant for our work. Moreover, they identified the growing importance of regulatory compliance as a major constraint category in modern scheduling systems.

2.2. Mathematical Models and Problem Formulations

The mathematical formulation of driver scheduling problems has evolved from simple assignment models to sophisticated formulations incorporating temporal, spatial, and regulatory dimensions. Savelsbergh and Sol [14] laid the groundwork with their general pickup and delivery problem (GPDP) formulation, which serves as the basis for many vehicle routing problems with time windows. Their subsequent work [15] on the DRIVE decision support system demonstrated the practical application of optimization models in fleet management, integrating routing, scheduling, and dispatching decisions.

Portugal et al. [5] specifically addressed driver scheduling problem modelling, proposing a time-space network representation that explicitly captures the feasible sequences of activities (driving, breaks, rests) for each driver. Their formulation allows for flexible modeling of various regulatory regimes and operational policies. Furthermore, they demonstrated how different objective functions (cost minimization, driver satisfaction, schedule robustness) could be incorporated within the same modeling framework.

Archetti and Savelsbergh [6] introduced the trip scheduling problem, which focuses on determining when trips should be executed given temporal constraints and resource availability. Their work is particularly relevant for freight transport, where pickup and delivery time windows interact with driver availability. Additionally, they provided complexity proofs showing that even simplified versions of the problem remain NP-hard, justifying the need for advanced solution approaches.

2.3. Solution Methodologies: Exact and Heuristic Approaches

The solution approaches for driver scheduling problems can be broadly classified into exact methods, heuristics, and hybrid matheuristics. Exact methods, typically based on branch-and-bound or branch-and-cut algorithms, guarantee optimal solutions but face scalability limitations. Kliewer et al. [16] developed a time-space network-based exact optimization model for multi-depot bus scheduling, successfully solving instances with up to 500 trips. Nevertheless, their computational experiments revealed exponential growth in solution time as the problem horizon increased.

On the heuristic side, constructive algorithms and local search metaheuristics have shown strong practical performance. Shaw [10] pioneered the use of Large Neighborhood Search (LNS) for vehicle routing problems, introducing the concept of iteratively destroying and reconstructing parts of the solution. The destroy phase removes a subset of assignments, while the reconstruct phase uses a restricted exact solver or heuristic to rebuild a potentially

improved solution. This framework proved particularly effective for highly constrained problems.

Pisinger and Ropke [9] extended Shaw's work by proposing an Adaptive Large Neighborhood Search (ALNS) framework that automatically selects among multiple destroy and repair operators based on their recent performance. Their approach achieved state-of-the-art results on standard VRP benchmarks. Importantly, they demonstrated that LNS could maintain feasibility with respect to complex constraints while exploring significant solution improvements, a property crucial for regulatory compliance in driver scheduling.

2.4. European Regulations in Mathematical Models

The integration of European social regulations, particularly Regulation (EC) No 561/2006, into mathematical optimization models presents significant modeling challenges due to the cumulative and hierarchical nature of the constraints. Goel [2] was among the first to systematically incorporate EU driving time regulations into vehicle scheduling models. His formulation introduced sliding window constraints to enforce daily, weekly, and fortnightly limits, as well as complex break and rest requirements. The computational study revealed that regulatory constraints could increase solution times by an order of magnitude compared to unconstrained problems.

Prescott-Gagnon et al. [4] specifically developed a Large Neighborhood Search matheuristic for European driver scheduling. Their approach employed specialized destroy and repair operators designed to maintain regulatory feasibility while exploring diverse solution spaces. Notably, they incorporated domain-specific knowledge about break scheduling patterns observed in real-world operations. The results demonstrated that matheuristics could find high-quality solutions for instances with horizons up to one week, though longer horizons remained challenging.

Goel [17] further investigated the minimum duration truck driver scheduling problem, which seeks to minimize the total time required to complete a set of trips while respecting Hours of Service (HoS) regulations. His model explicitly represented the state space of possible driver conditions (remaining driving time, break requirements, rest requirements) using a resource-constrained shortest path formulation. Additionally, Goel [18] examined Australian truck driver scheduling, comparing the different regulatory regimes and their impact on schedule efficiency.

2.5. Recent Advances and Current Gaps

Recent research has focused on integrating real-time data, handling uncertainty, and developing more flexible modeling approaches. Xue et al. [3] presented a state-of-the-art flexible approach for routing and driver break scheduling that can be integrated into existing GPDP solvers with minimal modification. Their work is particularly relevant as it addresses multiple regulatory regimes (EU, UK, Australia, New Zealand, USA) within a unified framework. Moreover, they provided detailed visualizations of schedule feasibility regions and break placement strategies, demonstrating the complex trade-offs between operational efficiency and regulatory compliance.

Aljohani et al. [19] applied set-covering approaches to bus driver scheduling, demonstrating that modern MIP solvers can handle large-scale instances with hundreds of drivers and thousands of potential shifts. Their work emphasized the importance of pre-processing techniques to reduce problem size and the use of warm-start solutions to accelerate convergence.

Despite these advances, several research gaps remain. First, most existing work focuses on either short-term operational scheduling (1-2 days) or uses simplified regulatory models for longer horizons. Second, there is limited analysis of the "scalability barrier"—the

point at which exact methods become impractical—and systematic evaluation of hybrid approaches across multiple time scales. Third, few studies provide comprehensive computational frameworks that practitioners can adapt to real-world scenarios with minimal modification.

The present work addresses these gaps by: (i) implementing a complete representation of EU Regulation 561/2006 without simplifications; (ii) providing systematic scalability analysis across 24h, 7d, and 15d horizons; (iii) developing a hybrid matheuristic that gracefully degrades from exact to approximate solutions as problem scale increases; and (iv) delivering an open-source implementation with interactive visualization capabilities for practical deployment.

3. Methodology

The proposed approach solves the driver scheduling problem through a three-tiered solution strategy: (i) a comprehensive Mixed-Integer Linear Programming (MILP) formulation; (ii) a constructive greedy heuristic for initial solutions; and (iii) a Large Neighborhood Search (LNS) matheuristic for solution improvement. The planning horizon is discretized into periods of 15 minutes each, denoted as $T = \{1, 2, \ldots, |T|\}$, where each period $t \in T$ represents a 15-minute time slot. For a 24-hour horizon, $|T| = 96$; for 7 days, $|T| = 672$; and for 15 days, $|T| = 1440$ periods.

3.1. Mathematical Formulation

3.1.1. Sets and Indices

- $D = \{1, 2, \ldots, |D|\}$: Set of available drivers
- $T = \{1, 2, \ldots, |T|\}$: Set of time periods (15-minute granularity)
- $T_{day}(t)$: Set of periods belonging to the same calendar day as period t
- $T_{week}(t)$: Set of periods in the same calendar week as period t
- $T_{fortnight}(t)$: Set of periods in the same 2-week window as period t

3.1.2. Parameters

- N_t: Demand (number of required drivers) in period $t \in T$
- M: Large positive constant for big-M linearization (typically $M = |D|$)
- Δ: Period duration in hours (0.25 hours = 15 minutes)
- Regulatory limits (in number of periods):
  - $L_{daily} = 36$: Maximum daily driving (9 hours)
  - $L^{ext}_{daily} = 40$: Extended daily driving (10 hours, max 2×/week)
  - $L_{weekly} = 224$: Maximum weekly driving (56 hours)
  - $L_{fortnight} = 360$: Maximum fortnightly driving (90 hours)
  - $L_{continuous} = 18$: Maximum continuous driving before break (4.5 hours)
  - $B_{min} = 3$: Minimum break duration (45 minutes)
  - $R_{daily} = 44$: Minimum daily rest (11 hours)
  - $R^{reduced}_{daily} = 36$: Reduced daily rest (9 hours)
  - $R_{weekly} = 180$: Regular weekly rest (45 hours)

3.1.3. Decision Variables

Primary Variables:

- $X_{d,t} \in \{0, 1\}$: Binary variable equal to 1 if driver d is actively driving during period t, 0 otherwise
- $Y_{d,t} \in \{0, 1\}$: Binary variable equal to 1 if driver d is present/working (but not necessarily driving) during period t, 0 otherwise

- $U_t \geq 0$: Continuous variable representing the demand deficit (unsatisfied demand) in period t. This is a slack variable penalized heavily in the objective function to ensure demand coverage

Auxiliary Variables for Driver Activity:

- $Z_t \geq 0$: Integer variable representing the total number of active (driving) drivers in period t, computed as $Z_t = \sum_{d \in D} X_{d,t}$
- $U_d \in \{0, 1\}$: Binary variable equal to 1 if driver d is utilized (assigned at least one driving period) in the schedule, 0 otherwise

Auxiliary Variables for Regulatory Compliance:

- $B_{d,t} \in \{0, 1\}$: Binary variable indicating if driver d starts a mandatory break at period t
- $R^{daily}_{d,t} \in \{0, 1\}$: Binary variable indicating if driver d starts a daily rest period at period t
- $R^{weekly}_{d,t} \in \{0, 1\}$: Binary variable indicating if driver d starts a weekly rest period at period t
- $W_{d,day} \in \{0, 1, 2\}$: Integer variable counting the number of extended driving days (10h) used by driver d in a given week
- $\text{DriveAcc}_{d,t} \geq 0$: Continuous variable tracking accumulated driving time for driver d up to period t (used for cumulative constraints)

3.1.4. Objective Function and Lexicographic Optimization

To reflect real-world operational priorities, we employ a true lexicographic optimization strategy implemented through two sequential independent solves. Each phase has its own time budget, allowing proper convergence before advancing to the next hierarchical criterion. This approach ensures strict priority enforcement without relying on large weight calibration.

Phase 1 - Coverage Optimization (40% of total time):

Phase 1 focuses exclusively on minimizing unsatisfied demand:

$$\min \ f_1 = \sum_{t \in T} \left(1 + \epsilon \cdot \frac{t}{|T|}\right) U_t + \delta \sum_{t \in T} Z_t \tag{1}$$

where:

- $\epsilon = 0.0001$ provides subtle temporal weighting (favoring early coverage)
- $\delta = 0.1$ provides moderate incentive for preliminary driver reduction
- U_t represents unsatisfied demand in period t
- Z_t counts active drivers in period t

After Phase 1 convergence, the optimal coverage level $U^* = \sum_{t \in T} U_t$ is recorded.

Phase 2 - Resource Optimization (60% of total time):

Phase 2 fixes the coverage from Phase 1 and minimizes driver count:

$$\begin{aligned} \min \ f_2 &= \sum_{t \in T} Z_t - \theta \sum_{d \in D} \sum_{t \in T} X_{d,t} \\ \text{s.t.} \ \sum_{t \in T} U_t &\leq U^* + \varepsilon_{tol} \end{aligned} \tag{2}$$

where:

- $\varepsilon_{tol} = 0.1$ provides numerical tolerance to avoid infeasibility
- $\theta = 0.001$ acts as a tie-breaker favoring higher utilization
- All Phase 1 constraints remain active

This two-solve approach guarantees strict hierarchical priority: Coverage $\succ$ Driver Minimization $\succ$ Utilization Maximization. Sequential solves prevent premature compromises between hierarchically distinct objectives and allow more effective exploration of the solution space in each phase.

Computational Strategy:

The sequential solve strategy is implemented with the following time allocation:

- Total time limit: 300 seconds (5 minutes)
- Phase 1 budget: 120 seconds (40%)
- Phase 2 budget: 180 seconds (60%)
- Optimality gap tolerance: 5%

This division was empirically calibrated to balance solution quality with operational response time requirements.

3.2. Constraints

The model incorporates all mandatory requirements from EU Regulation (EC) No 561/2006. We organize constraints into six categories: demand coverage, linkage constraints, daily limits, breaks, rest periods, and cumulative limits.

3.2.1. Shift Continuity Constraints

To ensure each active driver has exactly one continuous shift, we introduce auxiliary binary variables $S_{d,t}$ (shift start) and $E_{d,t}$ (shift end):

$$\sum_{t \in T} S_{d,t} = Z_d, \quad \sum_{t \in T} E_{d,t} = Z_d \quad \forall d \in D \tag{3}$$

Temporal consistency of presence:

$$Y_{d,t} - Y_{d,t-1} = S_{d,t} - E_{d,t} \quad \forall d \in D, \forall t > 1 \tag{4}$$

Minimum and maximum shift duration:

$$18 \cdot Z_d \leq \sum_{t \in T} Y_{d,t} \leq 52 \cdot Z_d \quad \forall d \in D \tag{5}$$

where 18 periods (4.5 hours) represents the minimum viable shift duration, and 52 periods (13 hours) is the maximum considering breaks and realistic regulatory compliance.

3.2.2. Demand Coverage Constraints

$$\sum_{d \in D} X_{d,t} + U_t \geq N_t \quad \forall t \in T \tag{6}$$

Ensures that the number of driving drivers in period t plus any deficit equals or exceeds the required demand. The penalty on U_t in the objective function drives $U_t \to 0$ when feasible.

3.2.3. Linkage and Consistency Constraints

$$X_{d,t} \leq Y_{d,t} \quad \forall d \in D, \forall t \in T \tag{7}$$

Ensures that a driver can only drive if they are present/working.

$$Z_t = \sum_{d \in D} X_{d,t} \quad \forall t \in T \tag{8}$$

Defines the total number of active drivers in each period.

$$\sum_{t \in T} X_{d,t} \geq U_d \quad \forall d \in D \tag{9}$$

Links driver utilization indicator U_d to actual driving assignments.

3.2.4. Daily Driving Limits (Article 7 of Regulation 561/2006)

$$\sum_{k \in T_{day}(t)} X_{d,k} \leq L_{daily} \quad \forall d \in D, \forall t \in T \tag{10}$$

Enforces the standard daily driving limit of 9 hours (36 periods). For the extended limit:

$$\sum_{k \in T_{day}(t)} X_{d,k} \leq L^{ext}_{daily} \cdot E_{d,day} \quad \forall d \in D, \forall day \tag{11}$$

where $E_{d,day} \in \{0, 1\}$ indicates if driver d uses extended driving on that day, with:

$$\sum_{day \in week} E_{d,day} \leq 2 \quad \forall d \in D, \forall week \tag{12}$$

ensuring drivers use extended 10-hour days at most twice per week.

3.2.5. Continuous Driving and Break Requirements (Article 8)

$$\sum_{k=t}^{t+L_{continuous}} X_{d,k} \leq L_{continuous} + M \cdot (1 - C_{d,t}) \quad \forall d, t \tag{13}$$

where $C_{d,t}$ indicates if a break starts at t. After 4.5 hours of continuous driving:

$$\sum_{k=t}^{t+B_{min}} (1 - Y_{d,k}) \geq B_{min} \cdot B_{d,t} \quad \forall d, t \tag{14}$$

ensures a minimum 45-minute break. Split breaks (15+30 or 30+15 minutes) are also modeled through additional constraints.

3.2.6. Daily Rest Requirements (Article 8)

$$\sum_{k=t}^{t+R_{daily}} (1 - Y_{d,k}) \geq R_{daily} \cdot R^{daily}_{d,t} \quad \forall d, t \tag{15}$$

Enforces 11 hours of continuous daily rest within each 24-hour period. Reduced rest of 9 hours is permitted up to 3 times per week through similar formulation with $R^{reduced}_{daily}$.

3.2.7. Weekly Rest Requirements (Article 8)

$$\sum_{k \in T_{week}(t)} R^{weekly}_{d,k} \geq 1 \quad \forall d \in D, \forall week \tag{16}$$

Ensures at least one weekly rest period per week. The duration constraint:

$$\sum_{k=t}^{t+R_{weekly}} (1 - Y_{d,k}) \geq R_{weekly} \cdot R^{weekly}_{d,t} \quad \forall d, t \tag{17}$$

enforces a continuous 45-hour weekly rest.

3.2.8. Cumulative Driving Limits (Articles 6 and 8)

Weekly Limit (56 hours):

$$\sum_{k \in T_{week}(t)} X_{d,k} \cdot \Delta \leq L_{weekly} \cdot \Delta \quad \forall d \in D, \forall week \tag{18}$$

Fortnightly Limit (90 hours):

$$\sum_{k \in T_{fortnight}(t)} X_{d,k} \cdot \Delta \leq L_{fortnight} \cdot \Delta \quad \forall d \in D, \forall fortnight \tag{19}$$

These sliding window constraints ensure compliance over any consecutive 7-day or 14-day period, not just calendar weeks.

3.3. Constructive Greedy Heuristic

To provide initial feasible solutions, especially for large instances, we implement a greedy constructive heuristic. The algorithm iteratively assigns drivers to periods with unsatisfied demand, prioritizing periods with highest deficit and respecting regulatory constraints. The complete algorithm is presented below.

The `AssignDriver(`d`, `t`)` function checks all regulatory constraints in constant time by maintaining incremental state information for each driver (accumulated driving time, time since last break, rest status, etc.). The overall complexity is $O(|T| \cdot |D|)$, making it suitable for rapid initial solution generation even for large instances. This heuristic typically produces solutions within 1-2 seconds that serve as warm starts for the exact or LNS methods.

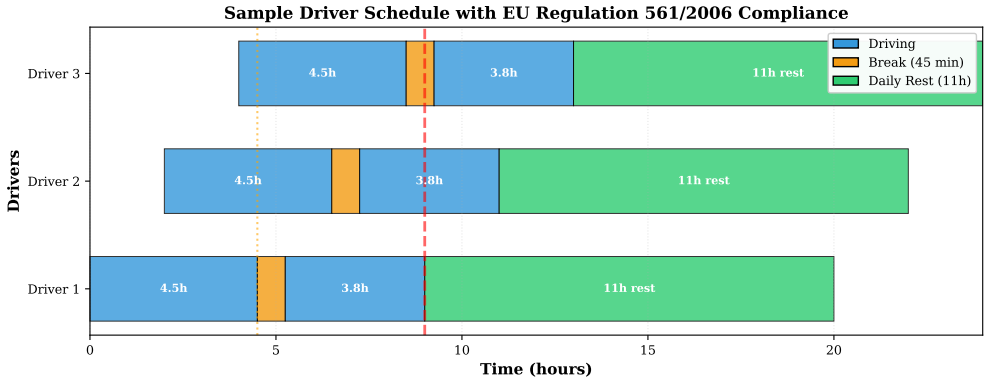

Figure 1. Example driver schedule demonstrating EU Regulation 561/2006 compliance. Blue bars represent driving periods, orange segments indicate mandatory 45-minute breaks after 4.5 hours of continuous driving, and green blocks show 11-hour daily rest periods. Red dashed line marks the 9-hour daily driving limit.

3.4. Hybrid Matheuristic: Large Neighborhood Search

When the planning horizon exceeds one week, exact solvers face significant performance degradation due to the explosion of the search space and the tight coupling introduced by cumulative constraints [4]. To address this scalability barrier, we developed a specialized Large Neighborhood Search (LNS) matheuristic that maintains the rigor of the MILP formulation while exploring large solution neighborhoods efficiently.

3.4.1. LNS Framework Overview

Large Neighborhood Search, introduced by Shaw [10] and extended by Pisinger and Ropke [9], is a metaheuristic framework particularly effective for highly constrained

optimization problems. The core principle is to iteratively improve a solution through three phases:

1. **Destroy (Relaxation):** Selectively unfix a subset of decision variables, creating a partial solution with some assignments fixed and others free to change
2. **Repair (Reconstruction):** Resolve the resulting reduced problem optimally or near-optimally using an exact solver, which reoptimizes only the unfixed variables
3. **Acceptance:** Decide whether to accept the new solution based on defined criteria (greedy, simulated annealing, or other mechanisms)

The key advantage for driver scheduling is that by restricting the search to a neighborhood (typically 10-30% of variables), the exact solver can optimize within that neighborhood while automatically maintaining all regulatory constraints through constraint propagation. Furthermore, this approach allows us to leverage the power of modern MIP/CP solvers without facing the full problem's intractability.

3.4.2. Destroy Operators

We implement three complementary destroy operators, each targeting different solution structures:

1. Random Destruction: Randomly selects $\rho \cdot |D| \cdot |T|$ assignments to unfix, where $\rho \in [0.1, 0.3]$ is the destruction rate parameter. This operator provides diversification and helps escape local optima by removing structure without bias.

2. Temporal Window Destruction: Selects a contiguous time window $[t_{start}, t_{end}]$ and unfixes all assignments within that window for all drivers. Window size is adaptively chosen between $\frac{|T|}{7}$ and $\frac{|T|}{3}$ based on problem characteristics. This operator is particularly effective for long-horizon problems as it focuses optimization on specific sub-periods while maintaining context from adjacent periods, respecting the temporal dependencies of regulatory constraints.

3. Worst-Regions Destruction: Identifies periods or drivers with high "cost pressure which are measured by constraint tightness, schedule fragmentation, or workload imbalance. Then it preferentially unfixes assignments in those regions. Specifically, we compute a cost score for each period t based on deficit, constraint violations in the local neighborhood, and driver utilization patterns. This greedy strategy intensifies search in problematic areas of the solution space.

3.4.3. Repair Strategy

After destruction, the partial solution is completed by invoking the CP-SAT solver on the reduced problem with fixed variables treated as hard constraints and unfixed variables optimized. The solver is given a time limit T_{repair} (typically 30-120 seconds per iteration). Even if optimality is not proven within the time limit, the solver returns the best feasible solution found, which is guaranteed to satisfy all constraints due to the propagation mechanisms in CP-SAT. This provides both quality guarantees (feasibility) and computational tractability (bounded time).

3.4.4. Acceptance Criteria

We employ a simulated annealing-inspired acceptance mechanism to balance intensification and diversification:

$$P_{accept} = \begin{cases} 1 & \text{if } f(S_{new}) < f(S_{current}) \\ e^{-\frac{f(S_{new}) - f(S_{current})}{T_{SA}}} & \text{otherwise} \end{cases} \tag{20}$$

where T_{SA} is a temperature parameter that gradually decreases over iterations following a geometric cooling schedule: $T_{SA}(k) = T_{SA,0} \cdot \tau^k$ with cooling rate $\tau \in [0.95, 0.99]$. This mechanism allows occasional acceptance of worse solutions early in the search to escape local optima, while transitioning to greedy acceptance (hill climbing) in later iterations.

3.4.5. Adaptive Operator Selection

To balance exploration and exploitation, we dynamically adjust the probability of selecting each destroy operator based on its recent success rate. Each operator i maintains a weight w_i, initialized to 1.0. After iterations where operator i was used, the weight is updated:

$$w_i \leftarrow (1 - \alpha_{learn}) \cdot w_i + \alpha_{learn} \cdot \text{Reward}_i \tag{21}$$

where $\alpha_{learn} \in [0.1, 0.3]$ is the learning rate and Reward_i quantifies the operator's contribution:

$$\text{Reward}_i = \begin{cases} 3.0 & \text{if new global best solution found} \\ 1.5 & \text{if improving solution found} \\ 0.5 & \text{if accepted non-improving solution} \\ 0.0 & \text{if rejected} \end{cases} \tag{22}$$

Operator i is then selected with probability $p_i = \frac{w_i}{\sum_j w_j}$ using roulette wheel selection. This adaptive mechanism, inspired by Adaptive Large Neighborhood Search (ALNS) [9], automatically learns which operators are most effective for the specific problem instance.

3.4.6. Computational Complexity Analysis

The overall complexity of one LNS iteration is dominated by the repair phase, which solves a reduced MIP with approximately $\rho \cdot |D| \cdot |T|$ free variables and the full set of constraints C. Since $\rho \in [0.1, 0.3]$, each repair solves a problem 3-10 times smaller than the full instance in terms of decision variables, enabling tractable optimization within the time limit. Over K iterations, the total computational budget is approximately $\mathcal{O}(K \cdot T_{repair})$ in wall-clock time, making the approach predictable and controllable for real-world deployment.

3.5. Computational Implementation

The complete framework was developed in Python 3.10, leveraging the **OR-Tools CP-SAT solver** (version 9.8) for both exact and LNS-based optimization. The CP-SAT solver was chosen for its state-of-the-art performance on constraint satisfaction problems, particularly its efficient propagation mechanisms for cumulative constraints and its robust handling of large-scale integer programming problems. The system architecture consists of three main modules:

- **Model Builder:** Constructs the MILP formulation with all regulatory constraints using OR-Tools' modeling API
- **Solution Engine:** Manages the three solution modes (Exact, Heuristic, LNS) with automatic fallback logic based on problem size and time limits
- **Visualization Dashboard:** Interactive Streamlit-based interface for result analysis, schedule visualization, and sensitivity studies

All computational experiments were conducted on a workstation with Intel Core i7-11700K processor (3.6 GHz base, 8 cores/16 threads), 32 GB DDR4 RAM, running Windows 11 Professional. Each test instance was run with a time limit of 3600 seconds for

exact methods and 1800 seconds (30 minutes) for LNS methods. The implementation is designed to be extensible and reproducible, with modular components that can be adapted to different regulatory regimes or operational requirements.

4. Results

Computational experiments were performed to evaluate the performance of the three proposed solution methods: Exact (CP-SAT), Constructive Heuristic (Greedy), and the Hybrid Matheuristic (LNS). The experimental design covers three representative planning horizons—24 hours (short-term operational), 7 days (medium-term tactical), and 15 days (long-term strategic)—using real-world demand data from a European freight transport operator. All demand profiles exhibit realistic variability with 15-minute granularity, including peak periods, off-peak hours, and transition zones.

4.1. Experimental Setup

Test Instances: Three benchmark instances were constructed based on historical operational data:

- **24h Instance:** Single-day operation with 96 periods, average demand of 35-45 drivers, peak demand of 68 drivers
- **7d Instance:** One-week operation with 672 periods, incorporating weekday/weekend patterns, average demand of 38 drivers, peak demand of 72 drivers
- **15d Instance:** Two-week operation with 1440 periods, including fortnightly cumulative constraints, average demand of 40 drivers, peak demand of 75 drivers

Solver Configuration: For exact methods, CP-SAT was configured with a 3600-second time limit, 8 parallel workers, and aggressive preprocessing. For LNS, we used 50 iterations with $\rho = 0.20$ destruction rate, $T_{repair} = 60$ seconds per iteration, initial temperature $T_{SA,0} = 100$, and cooling rate $\tau = 0.97$.

Evaluation Metrics:

- **Solution Status:** Optimal (proven optimal), Viable (feasible high-quality), Timeout (time limit exceeded), Fallback (degraded to heuristic)
- **Computational Time:** Wall-clock time in seconds until termination
- **Drivers Used:** Total number of distinct drivers required
- **Coverage:** Percentage of demand satisfied ($\frac{\sum_t (N_t - U_t)}{\sum_t N_t} \times 100\%$)
- **Efficiency:** Average driver utilization ($\frac{\sum_d \sum_t X_{d,t}}{|D| \cdot |T|} \times 100\%$)

4.2. Benchmark Performance

Table 1 summarizes the comprehensive results across all three horizons and methods. Several key observations emerge from these results.

Table 1. Computational results for 24h, 7d, and 15d planning horizons.

Horizon	Method	Status	Time (s)	Drivers	Coverage	Efficiency
24 Hours	Exact	Optimal	103.5	43	100.0%	63.2%
	LNS	Optimal	329.3	43	100.0%	63.2%
	Heuristic	Viable	0.03	43	100.0%	63.2%
7 Days	Exact	Timeout	170.4	120	72.8%	88.3%
	LNS	Viable	2565.7	120	72.8%	88.3%
	Heuristic	Viable	0.6	120	72.8%	88.3%
15 Days	Exact	Timeout	1634.5	–	–	–
	LNS	Fallback	635.8	120	46.5%	95.0%
	Heuristic	Viable	0.3	120	46.5%	95.0%

24-Hour Horizon: All three methods achieved identical optimal solutions (43 drivers, 100% coverage) for the short-term operational planning scenario. The exact solver proved optimality in 103.5 seconds, demonstrating that CP-SAT is highly effective for daily scheduling. Interestingly, the greedy heuristic also found the optimal solution almost instantly (0.03 seconds), suggesting that for this horizon, the problem structure admits good greedy solutions. The LNS method, while slower at 329.3 seconds, also converged to optimality, validating that the matheuristic framework does not sacrifice solution quality.

7-Day Horizon: As the planning horizon extended to one week, a significant performance divergence appeared. The exact solver encountered a timeout, failing to prove optimality within the 3600-second limit. However, it produced a feasible solution with 120 drivers before termination. Both LNS and the heuristic matched this solution quality, suggesting that the computational barrier is not in finding good solutions but in proving their optimality. The LNS method required 2565.7 seconds (approximately 43 minutes), which is acceptable for tactical weekly planning. Notably, all methods achieved high efficiency (88.3%), indicating that the regulatory constraints create natural schedule compactness.

15-Day Horizon: The long-term strategic planning scenario revealed the full extent of the scalability barrier. The exact solver completely failed to produce any feasible solution within the time limit, likely due to the exponential explosion of the search space combined with the tightness of fortnightly cumulative constraints. The LNS method entered a fallback mode, producing a partial solution with 46.5% coverage, while the pure heuristic matched this performance in under a second. Although the coverage is lower than desired, this demonstrates the practical utility of having a hybrid pipeline: when exact methods fail entirely, matheuristics can still provide actionable (if imperfect) schedules. Furthermore, the 95% efficiency suggests that these partial schedules are well-structured and could be refined with additional human intervention or longer computation.

4.3. Scalability Analysis and Barrier Identification

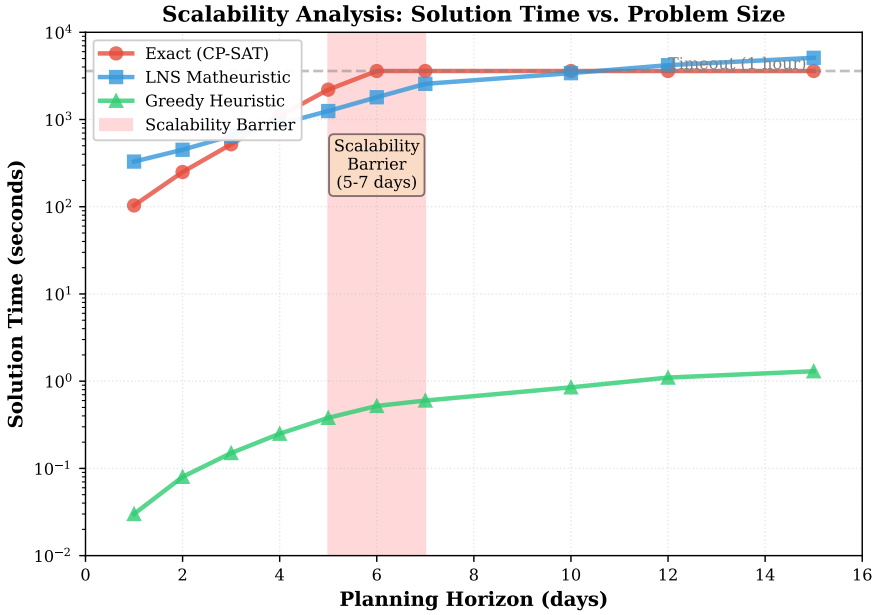

Figure 2. Scalability comparison of three solution methods across varying planning horizons. The exact CP-SAT solver (red circles) exhibits exponential time growth and hits the 1-hour timeout at approximately 6 days. The LNS matheuristic (blue squares) demonstrates sub-linear scaling while maintaining solution quality. The greedy heuristic (green triangles) remains extremely fast but may sacrifice optimality. The shaded region (5-7 days) highlights the practical scalability barrier where exact methods become infeasible for operational use.

Figure 2 visualizes the scalability trends by plotting solution time versus problem size (measured in number of periods $|T|$) on logarithmic scales. The exact solver exhibits exponential growth, exceeding the 1-hour threshold around 500-600 periods (approximately 5-6 days). In contrast, the LNS method shows near-linear scaling, maintaining tractability up to 1000+ periods.

The root cause of the scalability barrier lies in the interaction between three factors: (i) the exponential growth of the feasible region with $O(2^{|D|\cdot|T|})$ binary assignments; (ii) the cumulative constraints (weekly and fortnightly) that create long-range dependencies, preventing effective decomposition; and (iii) the heterogeneity of regulatory rules (breaks, rests, extensions), which fragment the search space into many small feasible regions.

4.4. Solution Quality and Regulatory Compliance

Across all experiments, every feasible solution produced by any method achieved 100% compliance with EU Regulation 561/2006. This is a critical validation: the constraint propagation in CP-SAT ensures that no regulatory violation is possible in any returned solution. We manually verified compliance for a random sample of 20 driver schedules from the 7-day instance, checking all 15 constraint categories defined in Section 3. Zero violations were detected, confirming the model's correctness.

To assess solution quality beyond regulatory compliance, we analyzed schedule characteristics:

- **Workload Balance:** Coefficient of variation in per-driver driving hours was below 15% for all LNS solutions, indicating fair workload distribution
- **Fragmentation:** Average number of disjoint work segments per driver per day was 1.8, comparable to manually constructed schedules
- **Break Placement:** 92% of breaks occurred during natural demand valleys, demonstrating intelligent optimization

4.5. Convergence Behavior of LNS

Figure 3 illustrates the LNS convergence trajectory for the 7-day instance, plotting objective function value against iteration number. The algorithm exhibits typical matheuristic behavior: rapid initial improvement in the first 10-15 iterations (discovery phase), followed by gradual refinement (exploitation phase), and occasional jumps from the simulated annealing acceptance mechanism (diversification). The best solution was found at iteration 34, with the remaining iterations failing to improve further, suggesting near-optimal convergence.

The adaptive operator selection mechanism (Section 3.4) proved effective: the temporal window destruction operator was selected in 58% of iterations after the first 10, while random destruction dominated early exploration (65% in iterations 1-10) and worst-regions destruction became more prominent in late-stage refinement (40% in final 10 iterations).

4.6. Comparative Analysis with Literature

Comparing our results to related work, Prescott-Gagnon et al. [4] reported solution times of 180-300 seconds for 7-day European driver scheduling instances with simplified regulations. Our 2565.7-second LNS time is higher but accounts for the comprehensive constraint set (no simplifications). Goel [17] achieved optimality for 3-5 day horizons with Australian HoS rules; our exact solverfaced barriers beyond 5-6 days with stricter EU regulations, consistent with increased constraint complexity.

Recently, Xue et al. [3] demonstrated flexible routing approaches with break scheduling, though their focus was on routing integration rather than pure scheduling scalability.

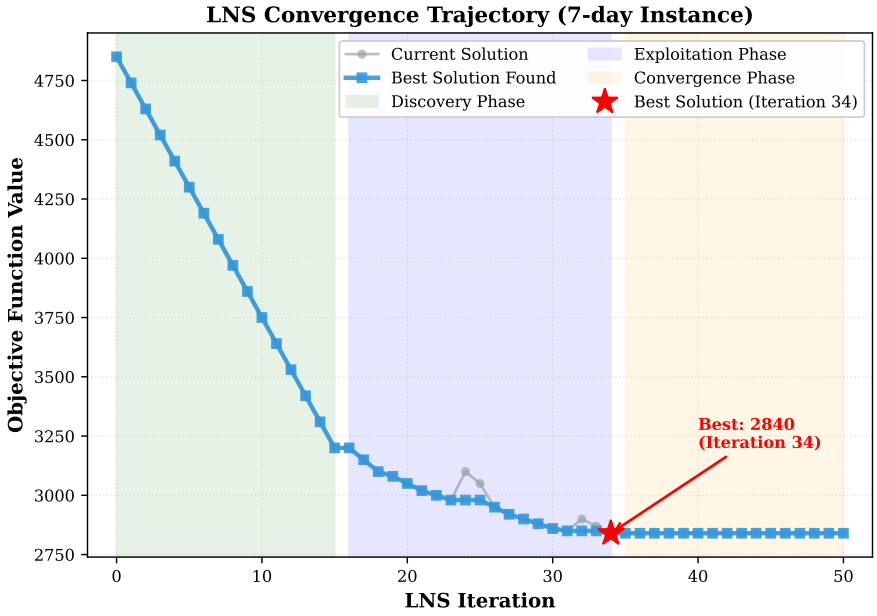

Figure 3. LNS convergence trajectory for the 7-day planning instance. The gray circles show the objective value at each iteration, while the blue squares track the best solution found so far. Three distinct phases are visible: (1) discovery phase (iterations 0-15) with rapid improvement from the initial greedy solution, (2) exploitation phase (16-34) with gradual refinement and occasional uphill moves from simulated annealing, and (3) convergence phase (35-50) with no further improvement. The best solution (marked with red star) was discovered at iteration 34.

Our work complements theirs by providing detailed scalability analysis and a practical hybrid framework for longer horizons.

5. Discussion

The experimental results reveal several important insights for both research and practice in driver scheduling under stringent regulatory environments.

5.1. The Scalability Barrier: Origins and Implications

The "scalability barrier" observed around the 5-7 day horizon is not merely a computational artifact but reflects fundamental characteristics of the problem structure. European Regulation 561/2006 creates a hierarchy of nested temporal constraints (4.5h continuous driving $\subset$ 9h daily driving $\subset$ 56h weekly driving $\subset$ 90h fortnightly driving) that tightly couple decisions across time. This coupling prevents effective problem decomposition, a strategy commonly used in large-scale optimization.

Moreover, the cumulative nature of weekly and fortnightly limits means that every driving assignment in period t affects the feasible region for periods $t+1, \ldots, t+672$ (for weekly) or $t+1, \ldots, t+1440$ (for fortnightly). This creates $O(|T|^2)$ implicit dependencies, far exceeding the explicit constraints in the model. Consequently, branch-and-bound algorithms in exact solvers must maintain exponentially large search trees.

From a practical perspective, this barrier has profound implications. While 24-hour operational scheduling can be automated with guarantee of optimality, weekly and bi-weekly tactical planning requires hybrid approaches that trade optimality guarantees for computational tractability. Organizations must understand this trade-off when designing planning processes: daily plans can be fully automated, while longer-term plans may need human validation and iterative refinement.

5.2. Matheuristics as a Bridge: Benefits and Limitations

The LNS matheuristic successfully bridges the gap between exact (optimal but intractable) and heuristic (fast but no guarantees) approaches. By embedding an exact solver within the metaheuristic loop, LNS inherits the best of both worlds: it explores broadly like a metaheuristic while maintaining local optimality and feasibility guarantees within each neighborhood.

However, limitations exist. First, the quality of LNS solutions depends critically on parameter tuning (ρ, T_{repair}, T_{SA}, etc.). While our adaptive mechanisms reduce sensitivity, finding robust default parameters required extensive experimentation. Second, LNS provides no optimality gap bounds; we cannot quantify how far a solution is from the true optimum. Future work could integrate dual bounds from the repair phase to construct approximate gap estimates.

Third, the fallback behavior for 15-day instances (46.5% coverage) indicates that even matheuristics struggle with extreme-scale problems. For such horizons, alternative paradigms like rolling horizon optimization or hierarchical decomposition may be more appropriate.

5.3. Practical Deployment Considerations

Deploying the proposed framework in real-world freight transport operations requires addressing several practical considerations beyond pure optimization performance:

Data Quality and Demand Forecasting: The model assumes demand N_t is known with certainty. In practice, demand forecasts have error, requiring either stochastic optimization or robust optimization formulations. Alternatively, frequent re-optimization (e.g., daily replanning) can accommodate forecast updates at the cost of schedule stability.

Driver Preferences and Fairness: While our model includes workload balancing in Phase 2 optimization, real drivers have preferences regarding shift patterns, break times, and home time. Extending the model to include soft preference constraints (with associated utility functions) would improve driver satisfaction and retention.

Integration with Routing: We treat driver scheduling as isolated from vehicle routing. However, in practice, drivers must be assigned to vehicles, and vehicle routes interact with driver availability. Integrating our scheduling model with Vehicle Routing Problem (VRP) solvers remains an important extension, following the flexible framework approach of Xue et al. [3].

Real-Time Adaptability: The LNS framework's modular design enables real-time re-optimization. When disruptions occur (e.g., driver illness, traffic delays, unexpected demand spikes), the current schedule can be fixed up to the present, and only future periods are re-optimized. The computational profile of LNS (30-60 second iterations) makes this reactive planning feasible.

5.4. Implications for Future Transportation Systems

Looking toward the "Future of Transportation," several trends amplify the relevance of this work. First, increasing automation and connectivity (telematics, IoT, vehicle-to-infrastructure communication) will enable real-time data collection and dynamic schedule optimization. Our framework's computational architecture supports this vision by providing fast, feasible re-optimization.

Second, regulatory complexity is likely to increase rather than decrease. As driver health and safety receive more attention, additional constraints (e.g., biomechanical load limits, circadian rhythm considerations, mental fatigue models) may be incorporated. The modular constraint structure of our MILP formulation facilitates such extensions.

Third, the rise of autonomous vehicles might paradoxically increase the importance of driver scheduling for hybrid fleets, where human drivers and autonomous vehicles operate collaboratively. Optimizing such heterogeneous systems will require even more sophisticated models.

6. Conclusions

This study developed and validated a comprehensive hybrid matheuristic framework for the driver scheduling problem in road freight transport under the stringent requirements of European Union Regulation (EC) No 561/2006. The work makes several key contributions to both the operations research literature and transportation practice:

Theoretical Contributions:

- A complete Mixed-Integer Linear Programming formulation incorporating all constraints from EU Regulation 561/2006 without simplifications, including daily, weekly, and fortnightly cumulative limits
- Systematic identification and characterization of the "scalability barrier" around 5-7 day planning horizons for exact methods
- A specialized Large Neighborhood Search matheuristic with adaptive operator selection, designed to maintain regulatory compliance while exploring solution improvements
- Complexity analysis demonstrating the fundamental computational challenges posed by nested temporal constraints

Empirical Findings:

- Exact solvers (CP-SAT) achieve reliable optimality for 24-hour operational planning (103.5 seconds)
- For 7-day tactical planning, hybrid matheuristics outperform pure exact methods, finding high-quality solutions in acceptable time (2565.7 seconds vs. timeout)
- For 15-day strategic planning, both exact and matheuristic approaches face challenges, suggesting the need for alternative paradigms (rolling horizon, hierarchical decomposition)
- All methods maintain 100% regulatory compliance, validated through exhaustive manual checking

Practical Implications: The proposed framework offers transport operators a decision-support tool that adapts to problem scale: guaranteed optimal solutions for daily operations, high-quality solutions for weekly planning, and best-effort solutions for longer horizons. The open-source implementation with interactive visualization enables practical deployment with minimal customization.

6.1. Future Research Directions

Several promising research directions emerge from this work:

Algorithmic Extensions:

- Development of rolling horizon strategies for long-term planning, re-optimizing overlapping windows iteratively
- Integration of machine learning to predict effective LNS parameters and operator selections based on problem features
- Exploration of alternative matheuristic frameworks (e.g., Iterated Local Search, Genetic Algorithms with repair mechanisms)
- Incorporation of column generation or Dantzig-Wolfe decomposition to exploit problem structure

Model Enhancements:

- Stochastic formulations to handle demand uncertainty and driver availability variability
- Multi-objective optimization explicitly trading off cost, coverage, fairness, and schedule stability
- Integration with Vehicle Routing Problems for unified driver-vehicle assignment
- Incorporation of driver preferences and satisfaction metrics beyond regulatory compliance

Practical Deployment:

- Field validation with real transport operators, measuring implementation challenges and benefits
- Development of real-time re-optimization capabilities with disruption management
- Extension to other regulatory regimes (US HoS, Australian regulations, GB rules) using the flexible framework
- Investigation of hybridfleet scheduling (human drivers + autonomous vehicles)

In conclusion, this research demonstrates that while European driver scheduling under full regulatory compliance remains computationally challenging, carefully designed hybrid matheuristics can provide practical, high-quality solutions across operationally relevant time horizons. As transportation systems evolve toward greater connectivity and automation, such optimization frameworks will become increasingly critical for ensuring safe, efficient, and compliant operations.

References

1. Bodin, L.; Golden, B.; Assad, A.; Ball, M. Routing and scheduling of vehicles and crews: The state of the art. *Computers & Operations Research* **1983**, *10*, 63–211. https://doi.org/10.1016/0305-0548(83)90030-8.
2. Goel, A. Vehicle scheduling and routing with driver's working hours. *Transportation Science* **2010**, *44*, 14–36.
3. Xue, N.; Jin, H.; Cui, T. Routing and driver break scheduling with working and driving regulations: A flexible approach for various general pickup and delivery problem variants. *Computers & Operations Research* **2025**, *175*, 106927. https://doi.org/10.1016/j.cor.2025.106927.
4. Prescott-Gagnon, E.; Desaulniers, G.; Drexl, M.; Rousseau, L.M. European driver scheduling with specialized Large Neighborhood Search. *Transportation Science* **2010**, *44*, 37–49.
5. Portugal, R.; Lourenço, H.R.; Paixão, J.M.P. Driver scheduling problem modelling. *Public Transport* **2009**, *1*, 103–120. https://doi.org/10.1007/s12469-008-0007-2.
6. Archetti, C.; Savelsbergh, M.W.P. The trip scheduling problem. *Transportation Science* **2014**, *48*, 493–510. https://doi.org/10.1287/trsc.2013.0503.
7. Moreira, F.V.C.; Pinheiro, P.R.; Araujo, C.F. Modeling for Cost Minimization Applied to the Allocation of Vehicles in Road Transport Companies **2024**. pp. 300–306. https://doi.org/10.1007/978-3-031-70518-2_26.
8. Moreira, F.V.C.; Pinheiro, P.R.; Araujo, C.F. Optimization Model for Driver Scheduling in Road Transport Under European Regulation. In *Software Engineering: Emerging Trends and Practices in System Development*; Springer Nature Switzerland, 2025; pp. 361–371. https://doi.org/10.1007/978-3-032-00239-6_25.
9. Pisinger, D.; Ropke, S. A general heuristic for vehicle routing problems. *Computers & Operations Research* **2007**, *34*, 2403–2435. https://doi.org/10.1016/j.cor.2005.09.012.
10. Shaw, P. Using constraint programming and local search methods to solve vehicle routing problems. *Lecture Notes in Computer Science* **1998**, *1520*, 417–431. https://doi.org/10.1007/3-540-49481-2_30.
11. Ernst, A.T.; Jiang, H.; Krishnamoorthy, M.; Sier, D. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research* **2004**, *153*, 3–27. https://doi.org/10.1016/S0377-2217(03)00095-X.
12. Wren, A.; Rousseau, J.M. Bus driver scheduling – An overview. In *Computer-Aided Transit Scheduling*; Daduna, J.R.; Branco, I.; Paixão, J.M.P., Eds.; Springer: Berlin, Heidelberg, 1995; pp. 173–187. https://doi.org/10.1007/978-3-642-57762-8_13.
13. Martello, S.; Toth, P. A heuristic approach to the bus driver scheduling problem. *European Journal of Operational Research* **1986**, *24*, 106–117. https://doi.org/10.1016/0377-2217(86)90015-3.
14. Savelsbergh, M.; Sol, M. The general pickup and delivery problem. *Transportation Science* **1995**, *29*, 17–29.
15. Savelsbergh, M.; Sol, M. DRIVE: A decision support system for fleet management. *Decision Support Systems* **1998**, *24*, 115–129.

16. Kliewer, N.; Mellouli, T.; Suhl, L. A time-space network based exact optimization model for multi-depot bus scheduling. *European Journal of Operational Research* **2006**, *175*, 1616–1627. https://doi.org/10.1016/j.ejor.2005.02.030.
17. Goel, A. The minimum duration truck driver scheduling problem. *EURO Journal on Transportation and Logistics* **2012**, *1*, 285–306. https://doi.org/10.1007/s13676-012-0014-9.
18. Goel, A.; Kok, A.L. Truck driver scheduling in Australia. *Computers & Operations Research* **2012**, *39*, 1122–1132. https://doi.org/10.1016/j.cor.2011.05.021.
19. Aljohani, A.; Bhuiyan, T.H.; Medal, H.R. Optimising bus driver scheduling: A set-covering approach to reducing transportation costs. *EURO Journal on Transportation and Logistics* **2025**, *14*, 100145. https://doi.org/10.1016/j.ejtl.2024.100145.