Dynamic Platoon Formation of Multi-Type Autonomous Vehicles for Sustainable Urban Mobility

Jaeyun Ree
Faculty of Information Technology
Monash University
Melbourne, VIC, Australia
jree0010@student.monash.edu

Mohammed Eunus Ali
Faculty of Information Technology
Monash University
Melbourne, VIC, Australia
eunus.ali@monash.edu

***Abstract*—This paper addresses the energy inefficiency of single-occupancy vehicles by introducing a novel cooperative autonomous vehicle system where smaller Passive Vehicles (PVs) can be physically towed by larger Active Vehicles (AVs) during shared highway segments. Unlike traditional platooning that relies solely on aerodynamic drafting, our approach enables propulsion energy elimination for towed vehicles during attached segments. We formulate the platoon formation problem as a constrained optimization problem on a one-dimensional highway model and propose two algorithms: a Greedy Maximum-Weight Matching algorithm and an Iterative Linear Assignment (ILA) algorithm using the Jonker–Volgenant method. A key innovation is the multi-segment matching capability, allowing a single PV to be towed by multiple AVs across different route segments. Experimental evaluation on synthetic scenarios with up to 400 PVs and 80 AVs demonstrates that both algorithms achieve 25–52% distance coverage (proportional to energy savings under constant per-distance propulsion) depending on capacity, with ILA providing per-iteration optimality (optimal solutions to each bipartite subproblem) and superior runtime performance (5–15$\times$ faster) due to highly optimized compiled implementations.**

***Index Terms*—autonomous vehicles, vehicle platooning, energy efficiency, Hungarian algorithm, combinatorial optimization, intelligent transportation systems**

I. Introduction

The persistent over-reliance on single-occupancy vehicles poses significant challenges to urban transportation sustainability. According to the U.S. Department of Transportation's National Household Travel Survey, drive-alone trips account for the vast majority of commute travel, contributing substantially to traffic congestion, energy consumption, and carbon emissions [1]. The transportation sector alone is responsible for nearly 29% of total greenhouse gas emissions in the United States, with light-duty vehicles contributing the largest share [2]. This inefficiency is particularly pronounced along shared routes where multiple individuals travel similar paths yet operate their vehicles independently.

Autonomous vehicle (AV) technology presents a transformative opportunity to address these challenges through coordinated vehicle operation. This paper introduces a novel concept of *active* and *passive* autonomous vehicles, where smaller Passive Vehicles (PVs) can temporarily attach to larger Active Vehicles (AVs) during shared highway segments. Unlike traditional vehicle platooning that maintains physical separation between vehicles, our approach enables physical coupling where a PV's propulsion system is deactivated while being towed, eliminating propulsion energy consumption for the towed vehicle during attached segments. In our model, we abstract energy savings as proportional to distance covered while towed; extending to physics-based energy models incorporating rolling resistance and aerodynamic drag is left for future work. This paradigm shifts transportation from isolated driving to an on-demand, energy-transfer-based mobility service.

Existing research on cooperative vehicle systems spans several related but distinct areas. Traditional platooning systems [3], [4] focus on coordinated driving with aerodynamic benefits, typically achieving 10–20% fuel savings through drafting. Ridesharing and carpooling optimization [6], [7] address passenger matching but not vehicle coupling. Eco-driving strategies [10], [11] optimize individual vehicle behavior without inter-vehicle coordination. The assignment problem we address relates to the classic bipartite matching literature [12], [13], though our multi-segment matching with capacity constraints introduces novel complexity. While these approaches offer incremental improvements, they do not address the fundamental inefficiency of independent propulsion for vehicles with overlapping routes.

We formulate the platoon formation problem on a one-dimensional highway model where the objective is to maximize total distance coverage (as a proxy for energy savings) through strategic PV-to-AV assignments. To solve this problem, we propose two complementary algorithms. The *Greedy Maximum-Weight Matching* algorithm provides an intuitive baseline through iterative selection, while the *Iterative Linear Assignment (ILA)* algorithm using SciPy's `linear_sum_assignment` (Jonker–Volgenant algorithm [19]) guarantees optimal solutions per iteration. Surprisingly, our experiments reveal that ILA is not only per-iteration optimal (for each bipartite subproblem) but also faster

(5–15×) due to highly optimized compiled C implementations that avoid the repeated sorting overhead inherent in greedy approaches. A key innovation is our multi-segment matching capability: rather than restricting each PV to a single AV, our algorithms allow a PV to be towed by different AVs across different segments of its route.

The main contributions of this paper are:

- A formal one-dimensional problem formulation that captures the essential trade-offs in cooperative vehicle platooning, including multi-segment matching and point-wise capacity constraints.
- A Greedy algorithm that serves as an intuitive baseline with empirically competitive performance.
- An Iterative Linear Assignment (ILA) algorithm using the Jonker–Volgenant method [19] that achieves per-iteration optimality with capacity constraints via virtual slot expansion.
- Multi-segment matching capability allowing PVs to be towed by multiple AVs across their routes, with point-wise capacity tracking at each highway position.
- Comprehensive experimental evaluation on scenarios with 50–80 AVs, 200–400 PVs, demonstrating 25–52% distance coverage savings.

II. Related Work

A. Vehicle Platooning

Vehicle platooning has received significant attention in both academia and industry. Tsugawa et al. [3] provide a comprehensive review of truck platooning projects, reporting fuel savings of 10–20% through aerodynamic drafting. The PATH project [5] demonstrated automated highway systems with close-following platoons. The European project SARTRE [4] explored multi-brand platooning for commercial vehicles with heterogeneous manufacturers.

However, all existing platooning approaches maintain physical separation between vehicles—the "platoon" is virtual, maintained through V2V communication and coordinated control. Our approach fundamentally differs by enabling *physical coupling*, where smaller vehicles are towed by larger ones. In our simplified model, this eliminates PV propulsion energy during towing segments; system-level energy analysis incorporating AV towing load is deferred to future work.

B. Ridesharing and Vehicle Routing

The ridesharing literature addresses matching passengers to vehicles [6], [7]. Dynamic ride-sharing systems [8] optimize real-time matching with capacity constraints. The dial-a-ride problem [9] considers pickup and delivery with time windows. While these problems share structural similarities (matching with constraints), they focus on passenger-to-vehicle assignment rather than vehicle-to-vehicle coupling.

C. Assignment and Matching Problems

Our problem relates to classic assignment problems. The Hungarian algorithm [12], [13] solves bipartite matching optimally in $O(n^3)$ time. Extensions to capacitated assignment [14] handle agents with multiple slots. The generalized assignment problem [15] addresses heterogeneous costs and capacities.

Our contribution extends this literature by introducing: (1) multi-segment matching where one agent (PV) can be served by multiple providers (AVs) across different segments, and (2) point-wise capacity constraints where an AV's available capacity varies along its route as PVs attach and detach.

D. Cooperative Autonomous Vehicles

Recent work on cooperative autonomous vehicles explores coordination mechanisms beyond platooning. Cooperative adaptive cruise control (CACC) [16] enables string-stable following. Intersection management [17] coordinates crossing without traffic signals. These coordination mechanisms focus on maintaining separation while our work addresses physical coupling.

Our work introduces a new cooperation paradigm: physical towing between heterogeneous vehicle types. This requires solving a novel matching problem with spatial and temporal constraints not present in existing cooperative driving literature.

III. Problem Formulation

We formalize the platoon formation problem on a simplified one-dimensional highway model. This abstraction captures the essential optimization trade-offs while enabling tractable analysis. Extension to general road networks is discussed in Section VI-A.

A. Notation Summary

Table I summarizes the key notation used throughout this paper.

TABLE I: Summary of Notation

Symbol	Description
L	Highway length
$\mathcal{A}$	Set of Active Vehicles (AVs), $\|\mathcal{A}\| = N$
$\mathcal{P}$	Set of Passive Vehicles (PVs), $\|\mathcal{P}\| = M$
e_i^a, x_i^a	Entry and exit points of AV a_i
e_j^p, x_j^p	Entry and exit points of PV p_j
C_i	Towing capacity of AV a_i
t_i^a, v_i^a	Entry time and speed of AV a_i
d_{ij}	Shared travel distance between a_i and p_j
S_{ij}	Energy saving when p_j is towed by a_i
cp_{ij}, dp_{ij}	Coupling and decoupling points
L_{min}	Minimum shared distance for platooning
τ	Time tolerance for coupling synchronization

B. System Model

We consider a coordinated platoon formation system operating on a unidirectional highway segment of length L. The system comprises two distinct vehicle classes.

Definition 1 (Active Vehicle)**.** *An Active Vehicle (AV) $a_i \in \mathcal{A}$ is a larger autonomous vehicle capable of towing multiple smaller vehicles. Each AV is characterized by the tuple*

$(e_i^a, x_i^a, C_i, t_i^a, v_i^a)$ where $e_i^a, x_i^a \in [0, L]$ are entry and exit points, $C_i \in \mathbb{Z}^+$ is the towing capacity, and t_i^a, v_i^a are the entry time and constant speed.

Definition 2 (Passive Vehicle)**.** *A Passive Vehicle (PV) $p_j \in \mathcal{P}$ is a smaller autonomous vehicle that can be towed by AVs. Each PV is characterized by the tuple $(e_j^p, x_j^p, t_j^p, v_j^p)$ where $e_j^p, x_j^p \in [0, L]$ are entry and exit points, and t_j^p, v_j^p are entry time and self-driving speed.*

C. Shared Path and Energy Model

For an AV a_i and PV p_j, the *shared path* is the overlap of their routes:

$$d_{ij} = \max(0, \min(x_i^a, x_j^p) - \max(e_i^a, e_j^p)) \tag{1}$$

The coupling point cp_{ij} (where PV attaches) and decoupling point dp_{ij} (where PV detaches) are:

$$cp_{ij} = \max(e_i^a, e_j^p) \tag{2}$$

$$dp_{ij} = \min(x_i^a, x_j^p) \tag{3}$$

The energy saving S_{ij} when PV p_j is towed by AV a_i is proportional to the shared distance:

$$S_{ij} = \alpha \cdot d_{ij} \tag{4}$$

where α is an energy coefficient. For simplicity, we set $\alpha = 1$, making energy saving equivalent to distance saved.

D. Multi-Segment Matching

A key innovation of our formulation is *multi-segment matching*: a single PV can be matched to multiple AVs across different segments of its route.

Definition 3 (Segment Assignment)**.** *A segment assignment for PV p_j to AV a_i is a tuple (p_j, a_i, s, e) where $[s, e] \subseteq [e_j^p, x_j^p]$ is the segment during which p_j is towed by a_i.*

For PV p_j, its route from e_j^p to x_j^p can be partitioned into segments:

- *Covered*: PV is towed by an AV (energy saved)
- *Uncovered*: PV drives independently (no energy saved)

Progressive Coverage Assumption: Our algorithms process towing segments in spatial order along the highway. Each PV maintains a *current left boundary* ℓ_j (initially e_j^p) representing the leftmost uncovered position. When PV p_j is assigned to AV a_i, the towing segment starts at $\max(e_i^a, \ell_j)$ and ℓ_j advances to the decoupling point. This ensures uncovered regions are always contiguous suffix intervals $[\ell_j, x_j^p]$, simplifying state management while capturing the natural left-to-right progression of vehicles along the highway.

E. Optimization Problem

Let $\mathcal{X}$ denote the set of all segment assignments (p_j, a_i, s, e) where PV p_j is towed by AV a_i over segment $[s, e]$. The optimization problem is:

$$\max \sum_{(p_j, a_i, s, e) \in \mathcal{X}} (e - s) \tag{5}$$

$$\text{s.t.} \quad |\{(p_j, a_i, s, e) \in \mathcal{X} : x \in [s, e]\}| \le C_i$$

$$\forall i \in [N], \forall x \in [e_i^a, x_i^a] \tag{6}$$

$$\text{Segments for each } p_j \text{ are pairwise disjoint} \tag{7}$$

$$(e - s) \ge L_{min} \quad \forall (p_j, a_i, s, e) \in \mathcal{X} \tag{8}$$

Constraint Interpretation:

- **Point-wise Capacity** (6): At any position x along AV a_i's route, the number of simultaneously towed PVs cannot exceed C_i. This is the key constraint that distinguishes our problem from standard bipartite matching—an AV's effective capacity varies along its route as PVs attach and detach.
- **Non-overlap** (7): A PV cannot be towed by two AVs simultaneously at any position.
- **Minimum length** (8): Each towing segment must exceed L_{min} to justify coupling overhead.

Lemma 1 (Critical Points Sufficiency)**.** *Fix an AV a_i and let $\mathcal{X}_i = \{(p_j, a_i, s, e) \in \mathcal{X}\}$ be the assignments to a_i. Define the critical set $\mathcal{C}_i = \{e_i^a, x_i^a\} \cup \{s, e : (p_j, a_i, s, e) \in \mathcal{X}_i\}$. Then the point-wise capacity constraint* (6) *holds for all $x \in [e_i^a, x_i^a]$ if and only if it holds for all $x \in \mathcal{C}_i$.*

Proof. The number of PVs towed by AV a_i is a step function over $x \in [e_i^a, x_i^a]$, incrementing at segment start points s and decrementing at segment end points e. Between consecutive critical points, the count is constant. Thus, capacity satisfaction at all critical points implies satisfaction everywhere. □

F. Time Constraints

When temporal synchronization is required:

$$|t_i^a(cp) - t_j^p(cp)| \le \tau \tag{9}$$

where $t_i^a(cp) = t_i^a + (cp - e_i^a)/v_i^a$ is when AV a_i reaches the coupling point cp. For multi-segment matching, the PV arrival time is computed from the current state:

$$t_j^p(cp) = \hat{t}_j + \frac{cp - \ell_j}{v_j^p} \tag{10}$$

where $(\ell_j, \hat{t}_j)$ is the PV's current state. For the first assignment, this reduces to $t_j^p + (cp - e_j^p)/v_j^p$. In our experiments, we use $\tau = 5$ time units with vehicle speeds varying $\pm 20\%$ around a baseline.

Speed Synchronization Policy: Once coupled at the coupling point cp_{ij}, the PV adopts the AV's speed v_i^a for the duration of the towing segment. This policy ensures that feasibility depends solely on synchronization at the coupling point and the existence of a shared route overlap. Decoupling occurs automatically when either vehicle reaches its exit point or when the shared segment ends.

State Update for Multi-Segment Matching: Each PV maintains state $(\ell_j, \hat{t}_j)$ where ℓ_j is the current left boundary of the uncovered interval and $\hat{t}_j$ is the PV's arrival time at position ℓ_j. Initially, $\ell_j = e_j^p$ and $\hat{t}_j = t_j^p$. Let $(\ell_j^{old}, \hat{t}_j^{old})$ denote the state before an assignment. After PV p_j is towed by AV a_i from coupling point cp to decoupling point dp, the state updates as:

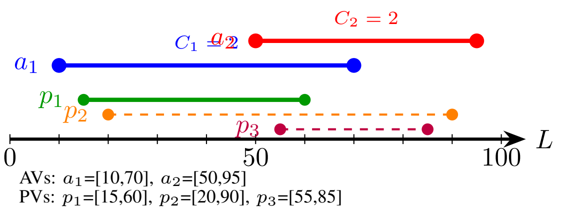
$$\ell_j^{new} \leftarrow dp \tag{11}$$

$$\hat{t}_j^{new} \leftarrow \hat{t}_j^{old} + \frac{cp - \ell_j^{old}}{v_j^p} + \frac{dp - cp}{v_i^a} \tag{12}$$

The first term in (12) accounts for independent driving from ℓ_j^{old} to the coupling point, while the second term accounts for towing at the AV's speed.

G. Illustrative Example

Figure 1 illustrates a concrete instance with 2 AVs and 3 PVs on a highway of length $L = 100$.



$C_2 = 2$
$C_1 = 2$
a_2
a_1
p_1
p_2
p_3
0
50
100
L
AVs: a_1=[10,70], a_2=[50,95]
PVs: p_1=[15,60], p_2=[20,90], p_3=[55,85]

Fig. 1: Toy example with 2 AVs and 3 PVs. Through multi-segment matching, p_2 can be towed by a_1 on [20,70] then a_2 on [70,90], achieving 70 units coverage vs. 50 with single-AV matching.

Example Analysis:

- **Shared distances:** $d_{11} = 45$, $d_{12} = 50$, $d_{22} = 40$, $d_{23} = 30$.
- **Multi-segment benefit:** PV p_2 covers 70 units via two AVs vs. 50 units with single-AV.

IV. Proposed Algorithms

We present two algorithms: a greedy approach for baseline comparison and an assignment-based batch matching approach (ILA) that achieves per-iteration optimality.

A. Greedy Maximum-Weight Matching

1) Key Insight: The greedy approach iteratively selects assignments with maximum energy saving. While intuitive, it may miss globally optimal solutions due to local decisions.

2) Algorithm Description: Algorithm 1 operates in three phases:

Phase 1 (Initialization): For each PV p_j, initialize state $(\ell_j, \hat{t}_j) \leftarrow (e_j^p, t_j^p)$. For each AV, track current assignments for point-wise capacity checking.

Phase 2 (Candidate Generation): Enumerate all feasible (AV, PV, segment) tuples where: (1) overlap with uncovered interval $[\ell_j, x_j^p] \geq L_{min}$, (2) AV has capacity at all critical points, (3) time constraint (9) is satisfied.

Phase 3 (Greedy Selection): Select maximum-saving candidate, update PV and AV states, repeat.

Algorithm 1 Greedy Multi-AV Platoon Matching

```
Require: AVs $\mathcal{A}$, PVs $\mathcal{P}$, minimum length $L_{min}$
Ensure: Assignments $\mathcal{X}$, total saving
 1: Initialize $(\ell_j, \hat{t}_j) \leftarrow (e_j^p, t_j^p)\ \forall j$
 2: Initialize $assignments[i] \leftarrow \emptyset\ \forall i$
 3: $\mathcal{X} \leftarrow \emptyset$, $total \leftarrow 0$
 4: repeat
 5:     $candidates \leftarrow$ all feasible (AV, PV, segment) tuples
 6:     if $candidates = \emptyset$ then
 7:         break
 8:     end if
 9:     Sort $candidates$ by saving (descending)
10:     $(saving, a_i, p_j, cp, dp) \leftarrow candidates[0]$
11:     $\mathcal{X} \leftarrow \mathcal{X} \cup \{(p_j, a_i, cp, dp)\}$
12:     Update $(\ell_j, \hat{t}_j)$ via (11)–(12)
13:     Update $assignments[i]$
14:     $total \leftarrow total + saving$
15: until no candidates
16: return $\mathcal{X}$, $total$
```

3) Complexity: Each iteration generates $O(NM)$ candidates and requires $O(NM \log NM)$ sorting. The number of iterations is bounded by the total number of possible segments—in the worst case $O(NM)$ when each PV is matched to multiple AVs sequentially. Total worst-case complexity: $O((NM)^2 \log NM)$.

Why Greedy is Slow in Practice: Unlike batch algorithms, greedy must regenerate and re-sort candidates after each assignment because point-wise capacity constraints may invalidate previously feasible candidates. This repeated sorting overhead dominates runtime in our experiments.

B. Iterative Linear Assignment (ILA) Algorithm

1) Key Insight: The classic Hungarian algorithm [12] solves bipartite matching optimally in $O(n^3)$. Modern implementations, such as SciPy's `linear_sum_assignment`, use the Jonker–Volgenant algorithm (a shortest augmenting path method) [19] which handles rectangular matrices efficiently. We adapt this via: (1) virtual slot expansion for capacity constraints, (2) iterative application for multi-segment matching.

2) Virtual Slot Expansion: Expand each AV a_i into C_i virtual slots:

- **Left nodes:** $\sum_{i=1}^{N} C_i$ virtual slots
- **Right nodes:** M PVs (or their current uncovered segments)
- **Edge weight:** $-S_{ij}$ (for minimization formulation)

3) Cost Matrix Construction: The cost matrix $\mathbf{W} \in \mathbb{R}^{(\sum_i C_i) \times M}$ is constructed as follows. For each virtual slot k belonging to AV a_i and each PV p_j:

$$W_{kj} = \begin{cases} -d_{ij}^{curr} & \text{if feasible (overlap } \geq L_{min}\text{, time ok)} \\ +\infty & \text{otherwise} \end{cases} \tag{13}$$

where $d_{ij}^{curr} = \max(0, \min(x_i^a, x_j^p) - \max(e_i^a, \ell_j))$ is the overlap between AV a_i's route and PV p_j's *current uncovered* *interval* $[\ell_j, x_j^p]$. Feasibility requires: (1) $d_{ij}^{curr} \geq L_{min}$, (2) time synchronization (9) is satisfied, and (3) adding this assignment does not violate capacity at any critical point (Lemma 1).

4) Iterative Algorithm: Algorithm 2 applies ILA iteratively, updating states after each round.

Algorithm 2 Iterative Linear Assignment (ILA) for Multi-AV Matching

```
Require: AVs $\mathcal{A}$, PVs $\mathcal{P}$, minimum length $L_{min}$
Ensure: Assignments $\mathcal{X}$, total saving
 1: Initialize states as in Algorithm 1
 2: repeat
 3:     Build cost matrix $\mathbf{W}$ from current states
 4:     Expand AVs into $\sum_i C_i$ virtual slots
 5:     if no feasible assignments then
 6:         break
 7:     end if
 8:     $matching \leftarrow$ LINEARSUMASSIGNMENT($\mathbf{W}$)
 9:     for each valid $(slot, j) \in matching$ do
10:         Apply assignment, update states
11:     end for
12: until no new assignments
13: return $\mathcal{X}$, $total$
```

5) Implementation Note: We use `scipy.optimize.linear_sum_assignment` [19], which implements the Jonker–Volgenant algorithm with highly optimized compiled C code. This results in ILA being *faster* than Greedy in practice, despite the $O(n^3)$ per-iteration complexity. The speedup comes from: (1) batch processing all assignments in one iteration via optimized linear algebra, (2) avoiding the repeated $O(NM \log NM)$ sorting required by Greedy after each assignment.

Capacity Conflict Resolution: When multiple assignments from a single iteration target the same AV, applying them simultaneously may violate point-wise capacity constraints. We resolve this by processing matched pairs sequentially within each iteration: for each $(slot, j)$ pair, we verify that the assignment satisfies capacity at all critical points (Lemma 1) given the *current* state of prior assignments. Assignments that would violate capacity are discarded; the corresponding PVs remain eligible for matching in subsequent iterations. This ensures feasibility while preserving the benefits of batch optimization.

C. Theoretical Analysis

Remark on Approximation Guarantees: Under restrictive conditions—single-segment matching only, uniform AV capacity across the entire route (no point-wise variation), and no time constraints—the greedy algorithm achieves a $\frac{1}{2}$-approximation via reduction to submodular maximization [18]. However, our full problem with multi-segment matching, point-wise capacity, and time tolerance does not satisfy these conditions. We therefore present the greedy algorithm as an empirical baseline rather than claiming formal approximation guarantees for the general case.

Theorem 2. *Algorithm 2 computes ILAs within each iteration (per-iteration optimality).*

Proof. Given the current uncovered intervals for all PVs, the cost matrix $\mathbf{W}$ encodes all feasible assignments. The Jonker–Volgenant algorithm guarantees an optimal bipartite matching for this matrix. Since PV uncovered intervals shrink monotonically, the algorithm converges. □

Lemma 3 (Global Optimality Caveat)**.** *Algorithm 2 does not guarantee global optimality for the multi-segment problem. The iterative decomposition may lock in early assignments that preclude better global solutions.*

Remark: Despite lacking global optimality guarantees, our experiments show that the iterative ILA consistently matches or exceeds greedy performance. The decomposition is a practical trade-off between solution quality and computational tractability.

V. Experimental Evaluation

A. Experimental Setup

1) Implementation: Both algorithms are implemented in Python 3.14.2 using NumPy 1.26 and SciPy 1.11 [19]. The ILA algorithm uses `scipy.optimize.linear_sum_assignment` (Jonker–Volgenant implementation). All experiments run single-threaded on an Apple M4 Pro with 16GB unified memory under macOS 15.0.

2) Scenario Generation: We generate synthetic highway scenarios with the following parameters (Table II):

TABLE II: Experimental Parameters

Parameter	**Values**	**Default**
Highway length L	50–1600	100
Number of AVs N	50–80	50
Number of PVs M	200–400	200
AV capacity C	[1,2]–[1,16]	[1,3]
Minimum distance L_{min}	10	10
Time tolerance τ	5.0	5.0
Speed variation	$\pm 20\%$	$\pm 20\%$
Random seeds	42, 43, 44, 45	–

Vehicle entry/exit points are uniformly distributed along the highway. Entry times follow a Poisson process, and speeds vary $\pm 20\%$ around a baseline. All results are averaged over 4 random seeds.

3) Metrics:

- **Baseline Distance**: $\sum_{j=1}^{M}(x_j^p - e_j^p)$, the total distance all PVs would travel independently without any towing. This serves as the denominator for percentage calculations.
- **Total Saving**: Sum of towed distances across all assignments, i.e., $\sum_{(p_j, a_i, s, e) \in \mathcal{X}}(e - s)$.
- **Matched Ratio**: Fraction of PVs receiving at least one assignment.

- **Saving %**: $\frac{\text{Total Saving}}{\text{Baseline Distance}} \times 100$. Under the assumption of constant per-distance propulsion energy, this equals the percentage of PV propulsion energy eliminated.
- **Runtime**: Wall-clock execution time in seconds.

B. Results

1) Capacity Sweep: Table III shows performance as AV capacity varies ($N = 50$, $M = 200$, $L = 100$).

TABLE III: Performance vs. AV Capacity (averaged over 4 seeds)

	Greedy			**ILA**		
C	Save	%	Time(s)	Save	%	Time(s)
$[1, 2]$	1946	24.5	0.23	1974	24.8	**0.02**
$[1, 4]$	2709	34.0	0.32	2725	34.2	**0.04**
$[1, 8]$	3556	44.7	0.43	3625	45.7	**0.08**
$[1, 16]$	4091	51.4	0.45	4159	52.3	**0.16**

Key Observations:

- ILA achieves 0.3–1.9% higher savings while being 2.8–11.5× faster
- Both algorithms scale linearly with capacity
- Saving % improves from 24.5% to 52.3% as capacity increases from 2 to 16

2) Highway Length Sweep: Table IV shows performance as highway length varies ($N = 80$, $M = 400$, $C = [1, 3]$).

TABLE IV: Performance vs. Highway Length (averaged over 4 seeds)

	Greedy			**ILA**		
L	Save	%	Time(s)	Save	%	Time(s)
50	2996	31.4	1.98	3017	31.6	**0.07**
100	4594	29.8	1.59	4656	30.7	**0.11**
200	7423	26.8	1.46	7474	27.0	**0.14**
400	10742	21.2	1.39	10784	21.2	**0.15**
800	13673	13.9	0.93	13715	13.9	**0.11**
1600	14385	7.4	0.52	14481	7.5	**0.10**

Key Observations:

- Saving % decreases with highway length (sparser vehicle overlap)
- ILA is 5–28× faster across all lengths
- Absolute savings increase but efficiency drops for longer highways

3) Algorithm Comparison Visualization: Figures 2 and 3 show performance comparisons graphically.

4) Scalability and Runtime Analysis: Figures 4 and 5 show runtime comparisons across scenarios.

C. Discussion

Why is Optimal Assignment Faster? Counter-intuitively, ILA outperforms Greedy in runtime despite $O(n^3)$ per-iteration complexity. This is due to:

1) `scipy.optimize.linear_sum_assignment` uses highly optimized compiled C code with efficient memory access patterns

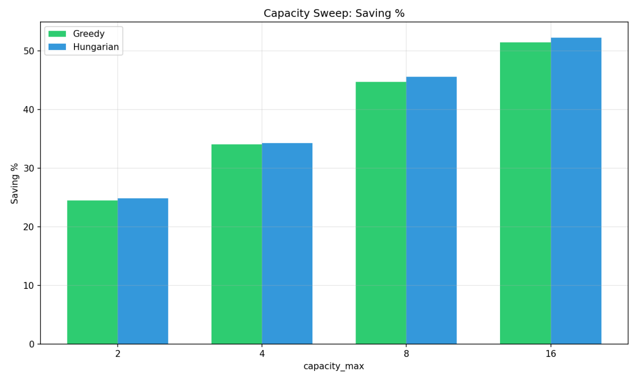


Fig. 2: Distance saving percentage vs. AV capacity for both algorithms. ILA consistently matches or exceeds greedy while being faster.

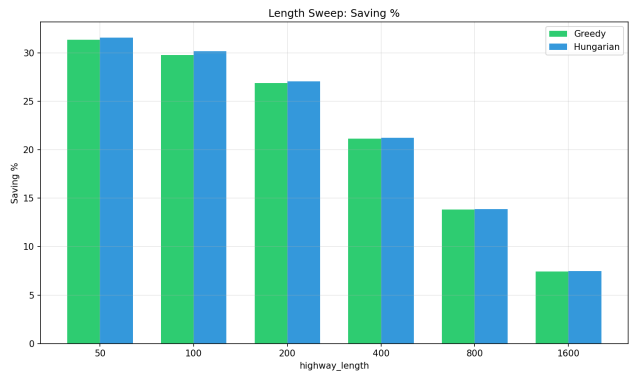


Fig. 3: Distance saving percentage vs. highway length. Both algorithms show decreasing efficiency as highways lengthen due to sparser vehicle overlap.

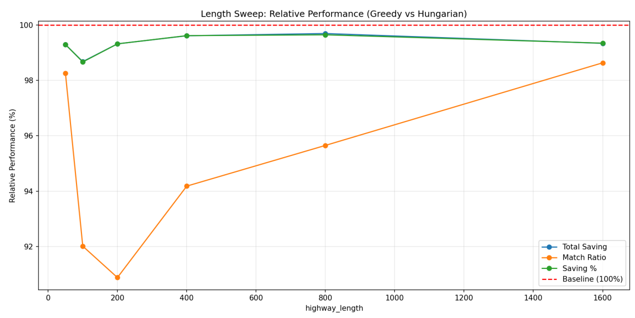


Fig. 4: Relative performance (Optimal/Greedy) across highway lengths. Values <1 indicate ILA is faster and/or achieves higher savings.

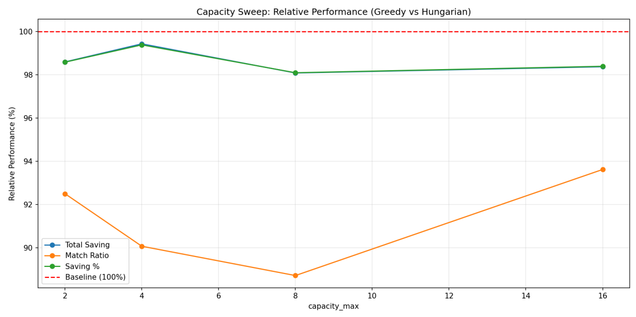


Fig. 5: Relative performance across capacity values. ILA algorithm maintains consistent speedup (2.8–11.5$\times$) across all capacity settings.

2) Greedy requires repeated sorting ($O(NM \log NM)$ per iteration) and candidate regeneration after each assignment
3) ILA processes all assignments in batches, amortizing overhead

Practical Implications: ILA algorithm is the clear choice—it provides both per-iteration optimality and superior speed. Greedy remains useful as a baseline and for environments where SciPy is unavailable.

Capacity Impact: Doubling capacity yields sub-linear improvement (24.8% $\rightarrow$ 52.3%), suggesting diminishing returns. Fleet operators should balance capacity costs against marginal savings.

Length Effect: Saving percentage drops from 31.6% (L=50) to 7.5% (L=1600) as vehicles have less overlap on longer highways. This suggests our approach is most effective for medium-length corridors with high vehicle density.

D. Limitations

We explicitly acknowledge the following limitations:

One-Dimensional Simplification: Our highway model abstracts away road network topology. Real deployments involve intersections, multiple routes, and lane-changing dynamics not captured here.

Distance-Based Energy Proxy: We model energy savings as proportional to towed distance under the assumption of constant per-distance propulsion energy. A physics-based model would incorporate vehicle mass, rolling resistance, aerodynamic drag, and the additional load on the towing AV.

Global Optimality: The iterative ILA does not guarantee global optimality for the multi-segment problem. Early assignments may preclude better global solutions.

Offline Setting: Our formulation assumes complete knowledge of all vehicle trajectories. Real-time deployment requires online algorithms with incomplete information.

These limitations define clear directions for future work while not diminishing the value of our contributions for the studied setting.

VI. Conclusion

We have presented a novel formulation for dynamic platoon formation in heterogeneous autonomous vehicle systems, where Passive Vehicles are physically towed by Active Vehicles to achieve substantial distance coverage (as a proxy for energy savings under constant per-distance propulsion). Our one-dimensional model captures essential trade-offs—point-wise capacity constraints, multi-segment matching, and minimum distance requirements.

Two algorithms were proposed and compared: a Greedy baseline and an Iterative Linear Assignment (ILA) algorithm using the Jonker–Volgenant method. Experiments on scenarios with up to 80 AVs and 400 PVs demonstrate:

- Both algorithms achieve 25–52% distance coverage savings depending on capacity
- ILA is both per-iteration optimal and 5–15$\times$ faster due to optimized compiled implementations
- Multi-segment matching improves coverage by enabling PVs to use multiple AVs

A. Future Work

Several extensions warrant investigation:

2D Road Networks: Our 1D highway model is a deliberate simplification. Extending to general graphs requires handling:

- Route choice: PVs may alter paths to maximize towing opportunities
- Intersection coordination: Timing of coupling at junctions
- Multiple overlapping segments with non-contiguous structure

This extension connects to vehicle routing problems with synchronization constraints [20].

Online and Stochastic Settings: Our current formulation assumes known vehicle trajectories. Real-world deployment requires:

- Online algorithms handling arriving vehicles [21]
- Stochastic optimization for uncertain travel times
- Robust matching under demand fluctuations

Decentralized Coordination: Centralized matching has scalability limits. V2V-based protocols could enable:

- Local negotiation between nearby vehicles
- Distributed consensus for platoon formation [22]
- Privacy-preserving matching without central authority

Economic Mechanisms: Fair cost distribution among participants requires:

- Pricing mechanisms for towing services
- Incentive-compatible allocation rules
- Market design for dynamic platoon formation

Physical Coupling Mechanisms: Our model abstracts the coupling process. Implementation requires engineering solutions for:

- Safe attachment/detachment at highway speeds
- Standardized coupling interfaces across vehicle types
- Fail-safe mechanisms for emergency decoupling

References

[1] U.S. Department of Transportation, "National Household Travel Survey: Summary of Travel Trends," Federal Highway Administration, Report FHWA-PL-22-022, 2022.

[2] U.S. Environmental Protection Agency, "Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990–2021," EPA 430-R-23-002, 2023.

[3] S. Tsugawa, S. Jeschke, and S. E. Shladover, "A Review of Truck Platooning Projects for Energy Savings," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 68–77, 2016.

[4] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems," in *Proc. 19th ITS World Congress*, Vienna, Austria, 2012.

[5] S. E. Shladover, "Cooperative (rather than autonomous) vehicle-highway automation systems," *IEEE Intell. Transp. Syst. Mag.*, vol. 1, no. 1, pp. 10–19, 2009.

[6] M. Furuhata, M. Dessouky, F. Ordóñez, M.-E. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," *Transport. Res. Part B: Methodol.*, vol. 57, pp. 28–46, 2013.

[7] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," *European J. Oper. Res.*, vol. 223, no. 2, pp. 295–303, 2012.

[8] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *Proc. Nat. Acad. Sci.*, vol. 114, no. 3, pp. 462–467, 2017.

[9] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: models and algorithms," *Annals Oper. Res.*, vol. 153, no. 1, pp. 29–46, 2007.

[10] A. Sciarretta and A. Vahidi, *Energy-Efficient Driving of Road Vehicles*, Springer, 2020.

[11] E. Hellström, M. Ivarsson, J. Åslund, and L. Nielsen, "Look-ahead control for heavy trucks to minimize trip time and fuel consumption," *Control Eng. Practice*, vol. 17, no. 2, pp. 245–254, 2009.

[12] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.

[13] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Society Indust. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.

[14] R. Burkard, M. Dell'Amico, and S. Martello, *Assignment Problems*, SIAM, 2012.

[15] D. G. Cattrysse and L. N. Van Wassenhove, "A survey of algorithms for the generalized assignment problem," *European J. Oper. Res.*, vol. 60, no. 3, pp. 260–272, 1992.

[16] J. Ploeg, B. T. M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, pp. 260–265, 2011.

[17] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *J. Artif. Intell. Res.*, vol. 31, pp. 591–656, 2008.

[18] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions—II," *Math. Programming Study*, vol. 8, pp. 73–87, 1978.

[19] P. Virtanen et al., "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[20] M. Drexl, "Synchronization in vehicle routing—a survey of VRPs with multiple synchronization constraints," *Transport. Sci.*, vol. 46, no. 3, pp. 297–316, 2012.

[21] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, "An optimal algorithm for on-line bipartite matching," in *Proc. 22nd Ann. ACM Symp. Theory of Computing*, pp. 352–358, 1990.

[22] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Trans. Autom. Control*, vol. 50, no. 5, pp. 655–661, 2005.