Coordinated Vehicle Platooning with Fixed Routes: Adaptive Time Discretization, Strengthened Formulations and Approximation Algorithms

Fengqiao Luo
Uber Technologies, Inc.
Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL, 60208, fengqiaoluo2014@u.northwestern.edu

We consider the coordinated vehicle platooning problem over a road network with time constraints and the routes of vehicles are given. The problem is to coordinate the departure time of each vehicle to enable platoon formation hence maximizing the total fuel saving. We first focus on the case that the routes form a tree. In this case, the flexibility time window of each vehicle admits a unified representation, under which the synchronization of departure time is equivalent to the management of time-window overlapping. Based on this observation, a time-window-overlapping induced mixed-integer linear program formulation has been established which is equivalent to the continuous-time formulation established in the previous work. By imposing an adaptive time-discretization procedure, we can further reformulate the time coordination problem using a vehicle-to-time-bucket assignment induced mixed-integer linear program. Compared to the continuous-time formulation, the assignment formulation is free of big-M coefficients, and it is demonstrated by numerical experiments that the assignment formulation is indeed more effective in computational performance. As an independent interest, three approximation algorithms have been derived with provable competitive ratios under a certain regularity assumptions, which is for the first time on this discrete optimization problem. As an independent interest, three approximation algorithms have been derived with provable competitive ratios under a certain regularity assumptions, which is for the first time on this discrete optimization problem. As an independent interest, three approximation algorithms have been derived with provable competitive ratios under a certain regularity assumptions, which is for the first time on this discrete optimization problem. As an independent interest, three approximation algorithms have been derived with provable competitive ratios under a certain regularity assumptions, which is for the first time on this discrete optimization problem.

Key words: coordinated vehicle platooning; mixed-integer linear program; adaptive time discretization; approximation algorithm; polynomial time approximation scheme; loop breaking scheme; lattice group isomorphism

1. Introduction

Vehicle platooning is a navigation and control technology under active development that is proposed to be implemented in an intelligent transportation system (ITS), especially with applications to trucks to improve safety, increase energy efficiency, reduce cost and CO$_2$ emission, and increase road-usage efficiency. In a platoon of trucks on the road, multiple trucks drive in tandem without being physically connected, and one truck follows another with a small inter-vehicle distance, e.g. 6m~10m.
Vehicle platooning is often associated with autonomous driving, where only the lead vehicle needs to be operated by a driver and the trailing vehicles are automatically (or semi-automatically) controlled. Vehicle platooning technology includes automatic braking and speed control, and the system is able to react much faster than a human, improving safety and reducing the likelihood of collisions. As a consequence, a much shorter inter-vehicle distance is needed to guarantee safety compared to the human-driving case, hence increasing the road-usage efficiency and throughput. Trucks (especially trailing trucks) in a platoon can drive at lower fuel (energy) cost compared to solo driving due to reduced aerodynamic drag incurred. Extra energy saving comes from adaptive cruise control.

According to the research (Muratori et al. 2017) conducted by the National Renewable Energy Laboratory based on data collected from in-car devices, 65% of the miles travelled by trucks could be platooned, resulting in a 4% reduction in total truck fuel consumption. The fuel saving factor is impacted by the inter-vehicle distance, the position of the vehicle within the platoon and the travel speed. A recent research (B. McAuliffe et al. 2018) shows that the middle truck in a platoon saved the most at shorter gaps, while the trailing truck saves the most at longer gaps. Results demonstrated a wide range of fuel savings – with the lead vehicle saving up to 10% at the closest separation distances, the middle vehicle saving up to 17%, and the trailing vehicle saving up to 13%. On average the lead truck demonstrated fuel saving up to 5.3% and the trailing truck saved up to 9.7%. While platooning improved fuel economy at all speeds, travel at 55 mph resulted in the best overall miles per gallon. See references (Lammert et al. 2014, B. McAuliffe et al. 2018) for more details. Innovative hardwares and technologies are needed to enable platoon formation, splitting and maintenance with safety. According to Peloton Technology (Peloton Technology 2020), a pioneer company in the industry of connected vehicle technology, in-car devices for platooning consists of the following five major components: platooning control unit, wireless communication system, radar-based collision mitigation, driver controls and system display. The platooning control unit contains microprocessors, memory, radios, and environmental control logic that interfaces with truck control logic to support platooning. The wireless communication system contains GPS, DSRC, Cellular and Wi-Fi antennae to enable vehicle-to-vehicle (V2V) communications, cloud communication, truck positioning and gap control between platooning trucks. A radar mounted to the front bumper of every truck constantly monitors the road ahead and engages the brakes faster than a human driver when necessary. Driver controls are mainly toggle and rotary switches used to start or end a platoon and to adjust settings such as trailer configuration. The system display is used to view platooning opportunities, system status, and video transmission. The following truck can see the lead truck view captured by the forward-looking camera.
In general, the planning for truck platooning at scale over a road network can be divided into three levels: the strategic level, the tactical level and the operational level (Hoef 2018). At the strategic layer, the strategies have been generated by a central system named platoon coordinator offered by a platoon service provider, an organization that provides crucial platooning services shared between road transport providers for services such as certification, insurance and coordination (Janssen et al. 2015). The platoon coordinator can be a cloud that receives data from traveling or static vehicles spread over the network via base stations, and then compute routes, departure time and speed profiles for vehicles to form platoon when they merge on a shared road segment. The computation should utilize the forecast of traffic and its impact on the travel time. The information out from the computation will be sent to vehicles via base stations. At the tactical layer, the platoon manager (usually played by the lead vehicle) controls the formation, splitting and reorganization of the platoon based on the plans provided by the strategic layer. This can be achieved using V2V communication. At the operational layer, the in-car devices from each member vehicle in a platoon start controlling the vehicle with the reference speed and inter-vehicle distance set by the platoon manager.

It is worth to note that truck platooning is more than an experimental technology. Industrialization of the technology has made steady progress, and the first generation of it is close for commercial launch. The truck platooning market was valued at USD 1405 million in 2020, and it is expected to reach USD 6077 million by 2026, registering a CAGR of about 32% during the forecast period (2021-2026) (Expert Market Research 2020). North America is the current largest market and Europe is the fastest growing market in which the technology development is led by Sweden and Germany. Major players are Peloton Technology, Daimler AG, Volvo Group, DAF Trucks (Paccar Company) and Scania AB, etc.

1.1. Our contributions

In this paper, we focus on the time coordination problem (scheduling problem) of coordinated vehicle platooning over a road network when the vehicle routes are given. We have made a series of contribution to this problem:

1. We have discovered that there exists a unified coordinate for characterizing the flexible time window of each vehicle for platooning if the routes form a tree or a forest. The unified coordinate is induced by picking a root node of the tree, and the time flexibility of a vehicle can be characterized by a relative time window (RTW) with respect to the root node. The time coordination problem is shown to be equivalent to managing overlap of the relative time windows from different vehicles.
2. Based on the above observation, a time-window-overlapping formulation has been established which is free of big-M coefficients. This formulation has a drawback of using a number of constraints in the order of $|\mathcal{V}|^3|\mathcal{E}|^2$, where $|\mathcal{V}|$ and $|\mathcal{E}|$ are the number of vehicles and road links. This drawback can be tackled by imposing an adaptive time discretization procedure, after which the time axis (shared by all vehicles due to the unified coordinate) is chopped into time buckets that are not uniform in size, and each relative time window consists of multiple consecutive time buckets. It can be shown that the time coordination problem is equivalent a vehicle-to-time-bucket assignment problem which leads to a further improved formulation using a number of variables in the order of $|\mathcal{V}||\mathcal{T}|$ and $|\mathcal{T}||\mathcal{E}|$, where $|\mathcal{T}|$ is the number of (adaptive) time buckets.

3. We provide estimations of the total number of time buckets generated by the adaptive time discretization procedure. In particular, we provide universal lower and upper bounds for any instance, and bounds for the expectation total number of time buckets when the position and the size of each relative time window follows appropriate uniform distributions.

4. Two major approximation algorithms with provable competitive ratio have been designed for the FRCVP problem on a tree. The first one is a delicate greedy algorithm that can beat the trivial ratio $1/T$ by a small fraction, where $T$ is the total number of time buckets, treated as a constant. The second one is a polynomial time approximation scheme (PTAS) for problem instances with a certain regularization conditions. The PTAS is achieved by three essential steps: a heavy-traffic edge contraction preprocess, a recursive tree decomposition algorithm and an enumeration algorithm applied to each component tree after decomposition. To the best of our knowledge, this is the first systematic investigation of approximation algorithms on the FRCVP problem with fixed routes.

5. In a general case, the vehicle routes can form a graph with loops, to which the above formulations and algorithms are not directly applicable. To handle the general case, we have developed a loop-breaking scheme which systematically splits a leading node in a loop, creates virtual copies of vehicles involved in loops, and introduces two modes of time-bucket projection to imposing time-consistency among virtual copies of vehicles. These procedures tend to convert a general instance to an equivalent tree-based instance. An insightful discovery is that whether such conversion is achievable has a close connection to the pattern of lattice groups. In any case, this conversion can lead to high-quality heuristic methods for a general instance.

6. Sufficient computational studies have been conducted to test the adaptive time-discretization method and compare the computational performance of the vehicle-to-time-bucket assignment mixed-integer linear program versus the continuous-time formulation developed in our previous work (Luo and Larson 2020) on 30 tree-based numerical instances created on an artificial
network and the greater-Chicago highway network. It shows that the assignment formulation leads to a much better computational performance in the sense that it can solve more instances to optimality or lead to a smaller optimality gap within a fixed amount of CPU time. Furthermore, some additional strong evidences have been provided to support that the assignment formulation is indeed much more ‘tight and compact’ compared to the continuous-time formulation. Heuristic methods recasted based on the approximation algorithms have also been implemented and compared.

In Section 2, we introduce the notion of relative time window and use it to build an interval graph that can represent the overlapping relations among the relative time windows of all vehicles, for tree-based FRCVP instances. The time-window-overlapping formulation is then established with the interval graph as an input. In Section 3, we present an adaptive time discretization algorithm that chops the time axis into time buckets with different length for tree-based instances, based on which a vehicle-to-time-bucket assignment formulation can be established and its equivalence to the continuous-time formulation is proved. The design of approximation algorithms and the proof of competitive ratios are presented in Section 4 for tree-based instances. The extension of our solution approach to general cases is provided in Section 5. The computational investigation is given in Section 6.

1.2. Literature review

The paper contributes to the development of strategic solutions for vehicle platooning over the road network, with focus on high-level time coordination to enable platooning. We review related literature on the development of platooning technology (especially on the operational and tactical levels) and the research on the strategic level. We also review the literature on the usage of time-expanded network in problem formulation in the area of transportation and logistics.

**Platooning technology development** Implementation of truck platooning technology depends on the development of hardware and software used for adaptive cruise control and wireless communication. The components of platooning hardware are illustrated in (Peloton Technology 2020). They can be grouped as devices for driver-vehicle and inter-vehicle control, devices for positioning (GPS), V2V communication (DSRC antennae) and cloud communication (LTE antennae), and devices for monitoring and sensing. We refer to (Robinson et al. 2010, Tsugawa 2013, Tsugawa et al. 2016) for an overview of platooning technology and observed energy saving. The impact of wireless communication protocol on platoon formation is studied in (Hobert 2012, Wang et al. 2012). Speed control algorithms applied during the process of platoon formation play an essential role in achieving fuel saving. The control theory and algorithms for maximizing the fuel saving at
the vehicle level are investigated in (Shida and Nemoto 2009, Liang et al. 2013, van de Hoef et al. 2015, Zhang et al. 2019, Gong et al. 2016, Ye et al. 2019). The control algorithms for maintaining a platoon at energy efficiency is studied in (Lu and Shladover 2011). The acceptable driver reaction time in the case when multiple trucks are under cooperative adaptive cruise control has been investigated in (Nowakowski et al. 2010, 2011). A variety of experiments and road tests have been conducted to gain a comprehensive understanding of the amount of fuel saving in different situations and the impact factors (e.g., speed, gap, wind, and slope, etc.) (Dávila 2013, Alam et al. 2010, Browand et al. 2004, Schittler 2003, Bonnet and Fritz 2000, Browned et al. 2004, Schrotten et al. 2012). The data from experiments is used to design simulators and control algorithms. At the network level, Liang et al. (2014) analyzed sparse data of truck position to identify potential platooning opportunities. Besides of fuel saving, there are studies concerning the impact of truck platooning on the highway throughput (Vander Werf et al. 2002, Shladover et al. 2012, Fernandes and Nunes 2012) and on the roadside hardware and safety (Dobrovolny et al. 2019).

**Strategic solution for mining platooning opportunities** On the strategic level, the central controller (either a control tower or a cloud) needs to create a plan for route design and travel schedule coordination of multiple vehicles to optimize the overall fuel consumption by enabling achievable platooning. Note that the scale of region managed by the central controller can vary. In particular, the region can cover a few highway junctions (Xiong et al. 2021), or it can be at the level of a road network that distributes over multiple states (Luo et al. 2018, Luo and Larson 2020). In general, there could be central controllers managing regions at different scale that form a top-down hierarchy. There is a series of investigation on optimal plan for platooning of a handful of vehicles using mathematical programming based models (Baskar et al. 2009a, Larson et al. 2016, Van De Hoef et al. 2015). Some models integrate route design, time coordination and speed control to find an optimal plan that covers the strategic level and tactical level operations to minimize the overall fuel consumption during the entire process (Kammer 2013, Liang 2014, Van De Hoef et al. 2015, Baskar et al. 2013). These models often contain nonlinear terms in the objective to count the impact of speed change on the fuel saving factor. Some other models focus on high-level planning using mixed-integer linear programs with an overall fuel saving factor that only differentiates lead and trailing vehicles but not intending to model the platoon formation process, and hence these models are better at scale (Larson et al. 2015, Larsson et al. 2015, Luo et al. 2018, Luo and Larson 2020). There are some other studies using local search to propose routes suitable for platooning (van Doremalen 2015, Larson et al. 2013). As the platooning process involves speed change and catch-up, dynamic programming is also a useful tool to model it at the tactical level (Valdés et al. 2011, Xiong et al. 2021). Recently, Xiong et al. (2021) developed a novel stochastic dynamic programming
model to optimize the platooning process focusing on the case when multiple vehicles merge at the highway junction. Luo and Larson (2020) established a novel repeatedly route-then-schedule framework for solving coordinated vehicle platooning problem with time constraints for hundreds of vehicles distributed over the road network. The quantitative impact of traffic flow and road capacity on the platoon formation is also counted in some literature (Baskar et al. 2009b, 2013).

**Formulation based on time discretization** Time discretization and time-expanded network are often used tools in models of transportation problems in a network with time constraints or time windows (Ford and Fulkerson 1958, 1962). For example, in a service network design problem (Crainic 2000, Crainic and Hewitt 2021), one is given a set of commodities, each having an origin and destination node. It seeks for the optimal construction of routes, truck resources, delivery plan and schedules such that the shipment of each commodity satisfies (origin and destination) time constraints while minimizing the overall transportation cost. Time discretization converts the continuous-time parameter into discrete time buckets. Once the spatial network couples with the set of time buckets to form a time-expanded network, it can support formulations that require to indicate events such as reaching a node at a certain time bucket. Due to the integrated treatment of transition in space and time, the time-expanded network has been widely applied in service network design (Jarrah et al. 2009, Andersen et al. 2011, Erera et al. 2013, Crainic et al. 2016), travel salesman problem with time windows (Boland et al. 2017b), road traffic control (Köhler et al. 2002), space logistics (Ho et al. 2014) and disaster evacuation logistics (Li et al. 2018), etc. The challenge of using time-expanded network is that the model complexity grows as the granularity increases. Boland et al. (2017a) established a continuous-time solution approach to the service network design problem without explicitly modeling each point in time. This approach has a careful control of the model complexity: Instead of using the fully time-expanded network, it starts with a partially time-expanded network and refines it progressively as necessary. The solution eventually converges to the original continuous-time problem. The spirit of not using fully time-expanded network to reduce model complexity has also been explored in (Fischer and Helmberg 2012, Contardo et al. 2015, Pecin et al. 2017), etc. The time-expanded network has in particularly been used by Abdolmaleki et al. (2019) to optimize the routes and schedule for truck platooning over a road network with time constraints.

1.3. **Problem description**

In the fixed-route coordinated vehicle platooning (FRCVP) problem, we are given a set $\mathcal{V}$ of vehicles. For each vehicle $v \in \mathcal{V}$, the following information is given: the origin node $O_v$, destination node $D_v$, earliest possible departure time (from the origin) $T^O_v$, and the deadline of arriving time (at the
destination) $T^D_v$. In addition, the route $R_v$ of each vehicle $v$ is given, which consists of edges that form a path from the origin to the destination of $v$. Note that vehicles can have shared edges in their routes. The objective is to find an optimal departure time of each vehicle to maximize fuel saving of the vehicle system by forming platoons on edges along the vehicle routes. Multiple vehicles can form a platoon on an edge if they enter the edge at the same time. We consider the case that once a vehicle has departed from the origin, it is not allowed to pause at any intermediate point in the route. We focus on the deterministic problem setting in the paper, as we have discovered that even the deterministic problem has a rich mathematical structure. Incorporation of the uncertainty in the travel time and the impact of traffic requires further development, which is briefly discussed in Section 7. The following assumption summarizes the deterministic problem setting.

**Assumption 1.** (1) For any given edge in $G$, the speed on the edge is the same for all vehicles, but it can be edge dependent. (2) Once a vehicle has departed from the origin, it is not allowed to pause at any intermediate point in the route. (3) The fuel saving factor is a constant, and it is $\sigma^l$ for a lead vehicle and $\sigma^f$ for a trailing vehicle.

An implication of the above assumption is that the arrival time is equal to the departure time of a vehicle plus the travel time of all edges along the route. The FRCVP problem is a special case of the general coordinated vehicle platooning (GCVP) problem. In the GCVP problem, it requires to determine an optimal route and departure time for every vehicle to minimize the total fuel cost, whereas in the FRCVP problem vehicle routes are given.

### 1.4. Connection to the previous work

The GCVP has a high computational complexity, which is due to that forming a platoon on a road link requires well planned coincidence in space and time over the road network, i.e., all vehicles in that platoon need to enter the road link at the same time. In (Luo et al. 2018), we have established a mixed-integer linear program (MILP) that gives an optimal solution of GCVP, i.e., it can find optimal routes and departure times of all vehicles when solved to optimality. This formulation involves a large amount of big-M coefficients to activate or de-activate constraints that ensure the spatial coincidence and simultaneousness. It has been demonstrated in the computational study that these big-M coefficients can make the formulation very weak as the scale of the problem instance increases. To overcome this challenge, we have developed a repeated route-then-schedule algorithmic framework to decompose the integrated problem into routing sub-problems and scheduling sub-problems in a heuristic approach (Luo and Larson 2020). This approach is motivated by the fact that vehicle routes that can enable platooning and fuel saving are very limited in practice: qualified routes can only have a small amount of deviation from the shortest path. This fact can reduce
the size of MILP models and it also indicates possible benefits from decomposing the routing and scheduling procedures. In the novel route-then-schedule approach, routes are first assigned to each vehicle by solving a routing sub-problem and then a scheduling sub-problem is solved based on the given routes. The two sub-problems are independent MILP’s. In this framework, the routing and scheduling sub-problems are solved repeatedly, which is guided by a novel try-and-learn feedback mechanism to regenerate routes that are more promising for fuel saving. It has been shown via a polyhedral study that the routing sub-problem formulation is very tight and it does not involve any big-M coefficients, while the scheduling sub-problem still involves several big-M coefficients to enforce simultaneousness, but the amount is much less than the integrated MILP formulation from (Luo et al. 2018). The computational study shows that the routing sub-problem can be solved to optimality very efficiently for systems of up to 800 vehicles, while the scheduling sub-problem cannot be solved to optimality within 10 minutes for a system of more than 200 vehicles in general.

The FRCVP problem is exactly the scheduling sub-problem which is the second phase in each iteration of the repeated route-then-schedule algorithm developed in (Luo and Larson 2020). What we have discovered in this paper is that there exists a very strong reformulation and approximations to solve the FRCVP problem (or the scheduling sub-problem) more efficiently if the vehicle routes satisfy some regularity conditions, and the strong reformulation can be extended for general cases.

2. A Time-Window-Overlapping Formulation

Since the formulation of FRCVP established in (Luo and Larson 2020) involves big-M coefficients, it is natural to wonder if there exists an alternative formulation of FRCVP that is free of big-M coefficients. In this section, we give an affirmative answer to this question if the routes of all vehicles satisfy a mild regularity condition that should hold in most of the practical situations. The formulation of FRCVP with the desired property is named as a time-window-overlapping (TWO) formulation, which is based on an insightful necessary and sufficient condition on whether a subset of vehicles are platoonable given the feasible departure time window of each vehicle in the subset. The interpretation of a time window will become clear in a moment with assistance of some definitions and properties. The TWO formulation can avoid big-M coefficients at the cost of introducing additional binary variables. In this sense, it is a more compact formulation defined in a higher dimensional space. To formally describe the formulation and algorithms, we first introduce notations of sets, the graph representation of the road network and parameters. They are summarized in Table 1. Some other parameters will be defined later when basic properties are revealed. For the rest of the paper, we assume that the graph $G$ is connected, otherwise, one can decompose the FRCVP problem into multiple independent FRCVP sub-problems each induced by a component of $G$. 
**Definition 1 (Ideal Path).** A path \( P \) in \( G \) is an ideal path if \( |\mathcal{V}_e| \geq 2 \). A path \( P \) in \( G \) is a maximal ideal path if \( P \) is ideal and there does not exist another ideal path \( P' \) in \( G \) which is different from \( P \) such that \( P \subseteq P' \) and \( \cap_{e \in P'} \mathcal{V}_e = \cap_{e \in P} \mathcal{V}_e \).

**Definition 2 (Path Shared by Vehicles).** Let \( P \) be a path on the graph \( G \), and \( \mathcal{V}' \) be a subset of vehicles. Vehicles in \( \mathcal{V}' \) share the path \( P \) if \( P \subseteq \mathcal{R}_v \) \( \forall v \in \mathcal{V}' \). We use the notation \( \mathcal{V}(P) \) to denote the set of vehicles that share the path \( P \).

**Definition 3 (Time Window).** Denote \( t_{u,i} \) and \( t_{u,i}' \) \( \forall u \in \mathcal{V}, \forall i \in \mathcal{N}_u \) as the lower and upper bound of the arrival time at node \( i \) for vehicle \( u \), where \( t_{u,i} = T_u^o + \sum_{(r,s) \in \mathcal{R}_u(O_u,i)} T_{rs} \) and \( t_{u,i}' = T_u^d - \sum_{(r,s) \in \mathcal{R}_u(i,D_u)} T_{rs} \). For any vehicle \( v \in \mathcal{V} \) and any node \( i \in \mathcal{N}_v \), the time window of the vehicle \( v \) at node \( i \) is defined as \([t_{v,i}, t_{v,i}']\).

An example of the ideal path and maximal ideal path is given in Figure 1. Proposition 1 gives a necessary and sufficient condition on whether a subset of vehicles that share a maximal ideal path can form a platoon on the path. This condition is characterized by the relation of time windows (Definition 3) between any two vehicles in the subset.
Proposition 1. Let \( P \) be a maximal ideal path in \( G \) and \( V' \) is a set of vehicles that share the path \( P \). Let \( s \) be the first node of \( P \). By coordinating the starting time of each vehicle at their origin, the vehicles in \( V' \) can form a single platoon when driving on \( P \) if and only if for any two vehicles \( u, v \in V' \) the intersection of their time windows at \( s \) is non-empty.

2.1. Maximal shared tree and unified measure of time windows

We have discovered a structure which is named as maximal shared tree (MST) (see Definition 5). A MST has a nice property which admits a unified measure of time windows of all vehicles involved in the MST to characterize the platooning structure among these vehicles when driving on the links of the MST. The notion and properties of MST are explained in this section.

Definition 4 (generalized path). A set \( P \) of edges on a directed graph form a generalized-path between nodes \( s \) and \( t \) if the undirected counterpart of \( P \) is an undirected path.

Definition 5 (maximal shared tree). (1) A directed subgraph \( T \) of the directed graph \( G \) is a tree if the undirected counterpart of \( T \) is a tree. (2) A tree \( T \) of the graph \( G \) is a shared tree if the following conditions hold: There exist a subset \( V' \) of vehicles such that \( E(T) = \bigcup_{v \in V'} R_v \). (3) For a shared tree \( T \), let \( V(T) \) be the maximal set of vehicles satisfying \( E(T) = \bigcup_{v \in V(T)} R_v \). (4) A shared tree \( T \) is decomposable if \( V(T) \) can be partitioned into two subsets \( V_1 \) and \( V_2 \) such that

![Figure 1](image-url)
An undecomposable shared tree $\mathcal{T}$ is extendable if there exists a vehicle $v \notin \mathcal{V}(\mathcal{T})$ satisfying that the edges in $\mathcal{E}(\mathcal{T}) \cup R_v$ form another undecomposable shared tree.

A maximal shared tree (MST) is a shared tree that is non-decomposable and non-extendable.

Example 1. We give an example of the shared tree and maximal shared tree based on Figure 1. The tree defined by the nodes $\{B, C, D, G, J\}$ and edges $\{(B, C), (C, D), (G, C), (J, C)\}$ is a shared tree but not a maximal shared tree since it can be extended. The tree defined by all nodes and all edges in Figure 1 is a maximal shared tree.

Given a maximal shared tree $\mathcal{T}$, we can select any node $r$ which satisfies that $d^-(r) = 0$ (indegree is 0), $d^+(r) = 1$ (outdegree is 1) as the root of $\mathcal{T}$, where the degrees are computed restricted to $\mathcal{T}$. For example, the root can be selected as the origin of any vehicle in $\mathcal{V}(\mathcal{T})$ such that there are no incoming edge to that node. We can define a relative time window (RTW) of a vehicle involved in a MST with respect to the root node. The definition of RTW provides a unified adjustment of time windows of all vehicles involved in a MST. The notion of RTW is formally defined in Definition 6, and it is based on properties of a shared tree revealed in Proposition 2.

Proposition 2. Let $\mathcal{T}$ be a non-decomposable shared tree, and $|\mathcal{V}(\mathcal{T})| \geq 2$. The following properties hold: (a) For any $u \in \mathcal{V}(\mathcal{T})$ there exists a vehicle $v \in \mathcal{V}(\mathcal{T}) \setminus \{u\}$ (depending on $u$) such that $R_u \cap R_v$ is non-empty. (b) For any distinct $u, v \in \mathcal{V}(\mathcal{T})$, the intersection $R_u \cap R_v$ is either an empty set or a directed path.

Definition 6 (relative time, relative time window). (1) Let $P$ be a generalized path between nodes $s$ and $t$. The relative time of $s$ with respective to $t$ is

$$\tilde{T}_{s,t} := \sum_{(i,j) \in P_{st}} T_{i,j} - \sum_{(i,j) \in P_{ts}} T_{i,j},$$

where $P_{st}$ is the set of edges on $P$ that are along the direction from $s$ to $t$, and $P_{ts}$ is the set of edges on $P$ that are along the direction from $t$ to $s$. (2) Let $\mathcal{T}$ be a MST and $r$ be the root of $\mathcal{T}$. Let $v$ be a vehicle in $\mathcal{V}(\mathcal{T})$. The relative time window (RTW) of $v$ (with respect to $r$) is defined as the interval: $[\tilde{t}_{v, O_v} + \tilde{T}_{O_v, r}, \tilde{t}_{v, O_v} + \tilde{T}_{O_v, r}]$, where $\tilde{T}_{O_v, r}$ is the relative time of $O_v$ with respective to $r$.

Example 2. We given an example for the definition of relative time and relative time window based on Figure 1. The relative time of node $I$ with respect to node $A$ is given as

$$\tilde{T}_{I,A} = T_{I,J} + T_{J,C} - T_{B,C} - T_{A,B},$$

where $P_{I,A} = \{(I, J), (J, C)\}$ and $P_{A,I} = \{(B, C), (A, B)\}$ in this case. If we select the node $A$ as the root of the shared tree given in Figure 1, the RTW of $v_4$ is given by the interval $[\tilde{t}_{v_4, I} + \tilde{T}_{I,A}, \tilde{t}_{v_4, I} + \tilde{T}_{I,A}].$
For a given maximal shared tree \( T \), let \( r \) be the root of \( T \), we can calculate the RTW of each vehicle in \( V(T) \) following the Definition 6. Based on these RTWs, we can construct an undirected graph named as the pseudo-platooning graph \( \tilde{G}(T) \) specified in Definition 7.

**Definition 7 (Pseudo-platooning Graph).** The pseudo-platooning graph \( \tilde{G}(T) \) consists of \(|V(T)|\) vertices representing each vehicle in \( V(T) \). An edge is added between two vertices (vehicles) if and only if their RTW’s intersect.

**Remark 1.** The pseudo-platooning graph is actually an interval graph according to the glossary of graph theory.

**Definition 8.** Let \( T \) be a MST and \( V(T) \) be the set of vehicles involved in \( T \). A subset \( V' \) of \( V(T) \) is pseudo-platoonable on \( T \) if there exists a departure-time coordination such that for any two vehicles \( u, v \in V' \) with \( R_u \cap R_v \neq \emptyset \), \( u \) and \( v \) are platooned on every \((i, j)\in R_u \cap R_v\), assuming that there is no restriction on the platoon size.

The definition of pseudo-platoonability is motivated from the situation in which we need to impose constraints on some vehicles’ departure time even though they do not form a platoon. For example, consider three vehicles \( v_1, v_2, v_3 \) involved in a MST. Suppose \( v_1 \) and \( v_2 \) form a platoon on the path \( s_1 \rightarrow s_2 \), \( v_2 \) and \( v_3 \) form a platoon on the path \( s_2 \rightarrow s_3 \) but there is no edges shared by \( v_1 \) and \( v_3 \). Obviously, just imposing departure-time constraints on \( v_1, v_2 \) and on \( v_2, v_3 \) separately is not sufficient to ensure the desired platooning configuration. Instead, we should impose departure time constraints on \( v_1, v_2 \) and \( v_3 \) simultaneously. In this situation, we require \( v_1, v_2 \) and \( v_3 \) to form a pseudo-platoon, which is different from a real platoon because \( v_1 \) and \( v_3 \) are not platooned on any edge though there are departure-time constraints imposed on them.

**Lemma 1.** Let \( T \) be a shared tree (not necessarily a MST), and \( V(T) \) be the set of vehicles involved in \( T \). (a) For any two vehicles \( u, v \in V(T) \) with \( R_u \cap R_v \neq \emptyset \), a departure-time coordination can make \( u \) and \( v \) platooned on \( R_u \cap R_v \), if and only if the departure times \( t_{u,O_u}, t_{v,O_v} \) determined by the coordination satisfy the following equation:

\[
t_{u,O_u} + T_{O_u,r} = t_{v,O_v} + T_{O_v,r},
\]

where \( r \) is the root of \( T \). (b) The vehicles in \( V(T) \) are pseudo-platoonable on \( T \) if and only if the intersection of their RTWs is non-empty.

**Corollary 1.** Let \( T \) be a MST, and \( V^0 \) be a subset of \( V(T) \). The vehicles in \( V^0 \) are pseudo-platoonable if and only if the subgraph of \( \tilde{G}(T) \) induced by \( V^0 \) is a clique.
Figure 2 The pseudo-platooning graph corresponding to the MST and vehicle set given in Figure 1.

Table 2 Variables for the time-window-overlapping formulation.

| Variables | Definition |
|-----------|------------|
| $\ell_{v,e} \in \{0,1\}$ | indicator variable of whether a vehicle $v$ is a lead vehicle in a platoon on edge $e$, $v \in V$; |
| $f_{u,v,e} \in \{0,1\}$ | indicator variable of whether a vehicle $u$ follows a vehicle $v$ in a platoon on edge $e$, $u,v \in V$, $u > v$; |

Example 3. For the vehicle routes given in Figure 1, suppose the travel time of each edge is 1, and we consider the following departure time and arrival time restrictions: the $(T^O_v, T^D_v)$ for $v_1 \sim v_7$ is (4, 12), (3, 11), (4, 13), (4, 11), (9, 19), (11, 15) and (13, 19), respectively. Let $A$ be the root of the MST. Using Definition 3, we can obtain the RTWs of vehicles $v_1 \sim v_7$ as [4, 8], [3, 7], [3, 9], [4, 7], [9, 15], [10, 11] and [12, 15], respectively. Using Definition 7 to construct the pseudo-platooning graph corresponding to the MST and the vehicle set. The pseudo-platooning graph is shown in Figure 2.

2.2. The time-window-overlapping formulation

The graph representation of pseudo-platoonability shown in Corollary 1 indicates that we can formulate the FRCVP problem on a MST as a binary linear program without using any big-M coefficients. In this section, we develop a time-window-overlapping formulation (TWOF) for this problem. It is established based on the RTWs, the pseudo-platooning graph and the result of Corollary 1. The formulation is given in (TWOF) and the variables are defined in Table 3. Note that we formulate the problem as a maximization problem of total fuel saving, since the fuel saving is more sensitive to the platooning configuration compared to the fuel cost when the routes are fixed. We use the same convention as in our previous work (Luo and Larson 2020) in describing a platoon with variables: to describe a platoon consisting of vehicles $\{v_1, \ldots, v_k\}$ on a road link $e$, we set the vehicle that has the smallest index say $v_{\min}$ ($v_{\min} = \arg \min_{i \in [k]} v_i$) as the lead vehicle i.e., $\ell_{v_{\min},e} = 1$ and it is followed by the other vehicles in this platoon, i.e., $f_{u,v_{\min},e} = 1$ for all $u \in \{v_1, \ldots, v_k\} \setminus \{v_{\min}\}$. For two vehicle indices $u$ and $v$, the notation $u,v := u',v'$ denotes the ordered tuple such that $u' = \max\{u,v\}$ and $v' = \min\{u,v\}$.

$$\max \sum_{e \in \mathcal{R}} \left( \sum_{v \in V} \sigma^f C_e \ell_{v,e} + \sum_{u > v \in V} \sigma^f C_e f_{u,v,e} \right)$$  \tag{TWOF.1}
s.t. \[\sum_{u \in \mathcal{V}_e : u > v} f_{u,v,e} \leq (\lambda - 1) \ell_{v,e} \quad \forall e \in \mathcal{R}, u \in \mathcal{V}_e, \] (TWOF.2)
\[\sum_{w \in \mathcal{V}_e : w < v} f_{w,v,e} \leq 1 - \ell_{v,e} \quad \forall e \in \mathcal{R}_v, v \in \mathcal{V}, \] (TWOF.3)
\[\sum_{u \in \mathcal{V}_e : u > v} f_{u,v,e} \geq \ell_{v,e} \quad \forall e \in \mathcal{R}_v, v \in \mathcal{V}, \] (TWOF.4)
\[f_{u,v,e} \leq a_{u,v} \quad \forall e \in \mathcal{R}, u, v \in \mathcal{V}_e, u > v, \] (TWOF.5)
\[f_{u,v,e} + f_{w,v,e'} \leq 1 + a_{u,v} \quad \forall e, e' \in \mathcal{R}, \text{ distinct } u, v, w \in \mathcal{V}, \] (TWOF.6)
\[\ell_{u_{e}\max,e} = 0 \quad \forall e \in \mathcal{R}, \] (TWOF.7)
\[\ell_{v,e} \in \{0, 1\}, f_{u,v,e} \in \{0, 1\} \quad \forall e \in \mathcal{R}, v \in \mathcal{V}_e, u, v \in \mathcal{V}_e, u > v, \] (TWOF.8)

where \(u_{e}\max\) is the vehicle with the maximum index in \(\mathcal{V}_e\). The constraint (TWOF.2) enforces that the size of a platoon should be no greater than \(\lambda\). The constraint (TWOF.3) indicates that a vehicle can only follow at most one other vehicle with the index smaller than it when traveling on an edge. The constraint (TWOF.4) simply means if a vehicle \(v\) is a lead vehicle, there must be at least one vehicle with a larger index following it. Note that the constraints (TWOF.2)-(TWOF.4) are the same as for the scheduling subproblem in (Luo and Larson 2020). The constraint (TWOF.5) enforces that if vehicle \(u\) follows vehicle \(v\) on any road link, their RTW's must intersect, i.e., there is an undirected edge \((u, v)\) on the pseudo-platooning graph. The constraint (TWOF.6) enforces the time coordination among three vehicles \(u, v\) and \(w\) such that if \(u, w\) are in a platoon at an edge \(e\) and \(v, w\) are in a platoon at an edge \(e'\), the departure time of \(u\) and \(v\) should be coordinated. Note that the constraint includes the case that \(e = e'\) and \(u, v, w\) are in a same platoon.

3. Adaptive time discretization and a vehicle-to-time-bucket assignment formulation

Time-expanded networks, first proposed by Ford and Fulkerson (1958), is an approximation tool for modeling problems in which the spatial and temporal information information is required to formulate the objective or constraints. The general idea is to discretize the measure of time by dividing the entire time horizon (uniformly in most cases) into many small time intervals (buckets) such that the difference between two instants within a same time interval is ignored. The original spatial network is then transformed into a time-expanded network by attaching time-interval labels to a node of the spatial network to form multiple nodes in the time-expanded network i.e., \(i \rightarrow (i, t)\) for all time-interval labels. A directed edge is created from node \((i, t)\) to node \((j, t')\) in the time-expanded network if and only if the travel time from \(i\) to \(j\) equals \(t' - t\) (subject to the approximation tolerance of time discretization). This technique has been applied by Abdolmaleki et al. (2019)
to establish a joint routing-and-scheduling model for the coordinated vehicle platooning. In their work, vehicles that visit a same node in the time-expanded network are regarded as being coincident in space and time, and they are eligible to form a platoon if traveling along a same road link. Scalability is a major drawback of using a time-expanded network. As the time horizon becomes large or discretization accuracy increases, the scale of the time-expanded network increases very quickly which can make computation intractable for models based on the network.

We develop an adaptive time-discretization (ATD) method that leads to an improved formulation for the FRCVP problem while maintaining the size of the problem to a highly manageable scale. The ATD method is established for vehicles that are involved in a MST. As demonstrated in Section 2.1, the feasible time interval is characterized by the RTW for vehicles involved in a MST. As a result, the essence of ATD is to generate time buckets based on the intersected and non-intersected segments of all RTWs involved. The details of the ATD method is given in Algorithm 1. An example of the ATD method is given in Example 4 and Figure 3.

**Definition 9 (sorted time buckets).** A set \( S = \{[p_k, q_k]\}_{k=1}^{\vert S\vert} \) is a *collection of sorted time buckets* if \( S \) is finite and \( q_k \leq p_{k+1} \) for all \( k \in \{1, \ldots, \vert S\vert - 1\} \).

**Example 4.** We consider four relative time windows (RTWs) given as \([0, 8]\), \([3, 11]\), \([5, 10]\) and \([9, 14]\). If the adaptive time discretization algorithm (Algorithm 1) is applied to the above RTWs, it will generate a collection of 7 sorted time buckets given as \([0, 3]\), \([3, 5]\), \([5, 8]\), \([8, 9]\), \([9, 10]\), \([10, 11]\) and \([11, 14]\).

1 If the intersection is a singleton, then the newly generated time bucket is a singleton.
2 The symbol \( \text{cl}(\cdot) \) means taking the closure of an input set.
3 The algorithm will ensure that for any two intervals \([\tau^k_1, \tau^k_2]\) and \([\tau^l_1, \tau^l_2]\) in \( S \), either \( \tau^k_2 \leq \tau^l_1 \) or \( \tau^l_2 \leq \tau^k_1 \). In this case, all intervals in \( S \) can be sorted ascendingly.
Algorithm 1 Adaptive time discretization.

1: **Input**: A set $\mathcal{V} = \{1, \ldots, N\}$ of vehicles that are involved in a MST. Denote $[\tau^1_k, \tau^2_k]$ as the RTW for each $k \in \mathcal{V}$.

2: **Output**: A set $S = \{(p_k, q_k)\}_{k=1}^{|S|}$ of sorted time buckets as the result of adaptive time discretization applied to $\{(\tau^1_k, \tau^2_k)\}_{k=1}^N$.

3: Set $S = \emptyset$, $t \leftarrow 0$.

4: for $k \in \{1, \ldots, N\}$ do

5: if $t = 0$ then

6: $S \leftarrow S \cup \{(p_k, q_k)\}$, $t \leftarrow t + 1$.

7: else

8: Refine($S, [\tau^1_k, \tau^2_k]$), $t \leftarrow t + 1$.

9: return $S$.

1: **procedure** Refine($S, I$)

2: The input $S = \{(p_k, q_k)\}_{k=1}^m$ is a collection of sorted time buckets and $I$ is an interval.

3: for $k \in \{1, \ldots, m\}$ do

4: if both $[p_k, q_k] \cap I$ and $[p_k, q_k] \setminus I$ are non-empty. then

5: $S \leftarrow S \setminus \{(p_k, q_k)\}$, $S \leftarrow S \cup \{(p_k, q_k) \cap I\}$, $S \leftarrow S \cup \{\text{cl}([p_k, q_k] \setminus I)\}$.

6: for $k \in \{1, \ldots, m-1\}$ do

7: if $[q_k, p_{k+1}] \cap I$ is proper then

8: $S \leftarrow S \cup \{(q_k, p_{k+1}) \cap I\}$.

9: if $[-\infty, p_1] \cap I$ is proper then

10: $S \leftarrow S \cup \{[-\infty, p_1] \cap I\}$.

11: if $[q_m, \infty] \cap I$ is proper then

12: $S \leftarrow S \cup \{[q_m, \infty] \cap I\}$.

13: Sort all time buckets in $S$ in the ascending order.

Proposition 3. Suppose the adaptive time discretization algorithm is applied to a set $C = \{[\tau^1_v, \tau^2_v] : v \in \mathcal{V}\}$ of RTWs corresponding to the set $\mathcal{V}$ of vehicles that share a MST. Let $S$ be the collection of sorted time buckets returned by the algorithm. Then the following bounds hold:

$$|C| \leq |S| \leq 2|C| - 1. \quad (4)$$

The bounds given in Proposition 3 are universal. A follow-up question is that if the position and length of each RTW is randomly distributed, what can be said about the expectation of $|S|$? To partially address this question, we investigate the case that the parameters for specifying RTW’s are independent and identically distributed, and derive lower and upper bounds on $E[|S|]$. The results are given in Theorem 1.

Theorem 1. Let $\mathcal{V} = \{1, \ldots, N\}$ be the set of vehicles involved in a MST, $0 < h < H$ be two parameters, and $r = h/H$ be the ratio of the two parameters. Suppose $C = \{[T^1_k, T^2_k]\}_{k=1}^N$, where $\{T^1_k : k \in [N]\}$ are i.i.d. random parameters following the uniform distribution on $[0, H]$. Suppose
Table 3  Variables for the vehicle-to-time-bucket assignment formulation (VA).

| Set        | Definition                                                                 |
|------------|---------------------------------------------------------------------------|
| $\mathcal{S}$ | the set of time buckets, i.e., the output of Algorithm 1;                 |
| $\mathcal{S}_u$ | the subset of time buckets that are feasible to vehicle $u$, i.e., $\cup_{t \in \mathcal{S}_u} t = \text{RTW}_u, u \in \mathcal{V}$; |
| $\mathcal{V}_{e,t}$ | the subset of vehicles that share edge $e \in \mathcal{R}$ and are feasible at time bucket $t \in \mathcal{S}$; |

| Parameter Definition | |
|---------------------|------------------------------------------------|
| $\lambda$          | the capacity of a platoon, i.e., the maximal number of vehicles allowed in a platoon |

| Variable Definition | |
|---------------------|------------------------------------------------|
| $x_{u,t}$           | 1 if vehicle $u$ is assigned to time bucket $t \in \mathcal{S}$, 0 otherwise; |
| $z_{e,t}^c$         | the number of full-size platoons (i.e., a platoon of size $\lambda$) taking $e \in \mathcal{R}$ at time bucket $t \in \mathcal{S}$; |
| $y_{e,t}^c$         | 1 if there exists a unsaturated platoon (i.e., a platoon of size at least one but less than $\lambda$) taking $e \in \mathcal{R}$ at time bucket $t \in \mathcal{S}$, 0 otherwise; |
| $y_{e,t}'$          | 1 if the unsaturated platoon taking $e \in \mathcal{R}$ at time bucket $t \in \mathcal{S}$ is of the size at least 2, 0 otherwise; |
| $q_{e,t}$           | the size of the unsaturated platoon taking $e \in \mathcal{R}$ at time bucket $t \in \mathcal{S}$; |
| $w_{e,t}$           | number of trailing vehicles taking $e \in \mathcal{R}$ at $t \in \mathcal{S}$, 0 otherwise. |

$\{T^k_k - T^k_k : k \in [N]\}$ are i.i.d. random parameters independent of anything else that follow the uniform distribution on $[0, h]$. Let $\mathcal{S}$ be the output of Algorithm 1 applied to $\mathcal{S}$. The expectation $\mathbb{E}[|\mathcal{S}|]$ of the total number of time buckets has the following lower and upper bounds:

$$N + \left(\frac{1}{2} \beta - \frac{1}{6} \beta^2\right) N \leq \mathbb{E}[|\mathcal{S}|] \leq N + \frac{q}{2(1-q)} N,$$

(5)

where $q = e(N + 1)r/2$ and $\beta = (N - 1)r$.

The key observation based on the adaptive time discretization is that the FRCVP on a tree can be formulated an advanced assignment problem. In the assignment problem, every vehicle can be assigned to a set of consecutive time buckets, and all vehicles that are assigned to a same time bucket are pseudo-platoonable. The validness of this formulation is shown in Theorem 2. The sets and variables used in the assignment formulation are given in Table 3, and the novel formulation is given as (VA).

$$\max \sum_{e \in \mathcal{R}} \sum_{t \in \mathcal{S}} (\sigma^t C_{e,t} z_{e,t} + \sigma^t C_{e,t} y_{e,t}^c + \sigma^t C_{e,t} w_{e,t})$$

(VA.1)

s.t.

$$\sum_{t \in \mathcal{S}_u} x_{u,t} = 1 \quad \forall u \in \mathcal{V},$$

(VA.2)

$$\lambda z_{e,t} - \sum_{u \in \mathcal{V}_{e,t}} x_{u,t} \leq 0 \quad \forall e \in \mathcal{R}, t \in \mathcal{S},$$

(VA.3)

$$\lambda (z_{e,t} + 1) - \sum_{u \in \mathcal{V}_{e,t}} x_{u,t} \geq 1 \quad \forall e \in \mathcal{R}, t \in \mathcal{S},$$

(VA.4)

$$w_{e,t} - \sum_{u \in \mathcal{V}_{e,t}} x_{u,t} + z_{e,t} + y_{e,t} = 0 \quad \forall e \in \mathcal{R}, t \in \mathcal{S},$$

(VA.5)
The objective is to maximize the total fuel saving of all vehicles. The total fuel saving is a sum of the fuel saving achieved at every \((e, t)\) combination. For each \((e, t)\), the fuel saving is determined by the number of lead vehicles \(z_{e,t} + y'_{e,t}\) and the number of following vehicles \(w_{e,t}\). As our convention, a single-vehicle platoon (i.e., a trivial platoon consisting of just one vehicle) at \((e, t)\), the only vehicle in that platoon does not count as a lead vehicle and it cannot save fuel consumption at \((e, t)\).

The constraint (VA.2) ensures that each vehicle must be assigned to exact one time bucket that is feasible for it. The constraints (VA.3) and (VA.4) ensure that \(z_{e,t}\) is the quotient of the total number of vehicles at \((e, t)\) divided by the platoon size limit \(\lambda\). The constraint (VA.5) establishes the quantitative relation among the numbers of all vehicles, the number of trailing vehicles and the single or lead vehicle(s). The constraint (VA.6) ensures that \(q_{e,t}\) is the number of vehicles involved in the unsaturated platoon, and its value is zero if all platoons are saturated. The constraint (VA.7) ensures that \(y_{e,t}\) is equal to one if and only if \(q_{e,t} \geq 1\). The constraints (VA.8) and (VA.9) are necessary conditions for \(y'_{e,t}\) according to its definition. But when combining with the sense of maximization, \(y'_{e,t}\) will take the value 1 instead of 0 as long as the two constraints are respected.

It is clear that an optimal time coordination (schedule) and platoon configuration of the FRCVP problem can be constructed based on the optimal solution of (VA). The construction is given as follows: For every \(e \in \mathcal{R}\) and \(t \in \mathcal{S}\), let \(V^*_{e,t}\) be the subset of vehicles taking road link \(e\) that are assigned to \(t\). Select any time instant from the time bucket \(t\) as the common time instant for vehicles in \(V^*_{e,t}\), and compute the corresponding departure times for these vehicles respectively. Let vehicles in \(V^*_{e,t}\) form platoons in a greedy manner when traversing \(e\) such that there is at most one unsaturated platoon for the \((e, t)\) combination, i.e., forming saturated platoons as many as possible.

The connection between optimal solutions of (VA) and FRCVP are revealed in Theorem 2.

**Definition 10.** Consider the set \(\mathcal{V}\) of vehicles involved in a MST. A time bucket is **feasible** to vehicle \(v\) if it is a subset of the RTW of \(v\). A sequence \(\{s_v: v \in \mathcal{V}\} \in \mathbb{R}^{|\mathcal{V}|}\) is a **feasible schedule** to the FRCVP problem if \(s_v \in [\tau_1^v, \tau_2^v]\) for all \(v \in \mathcal{V}\), where \([\tau_1^v, \tau_2^v]\) is the RTW of \(v\). A sequence \(\{t_v: v \in \mathcal{V}\} \in \mathbb{Z}_+^{|\mathcal{V}|}\) is a **feasible time-bucket assignment** to the formulation (VA) if \(t_v\) indexes a feasible time bucket for all \(v \in \mathcal{V}\).
Table 4  Comparison of three formulations for the FRCVP problem.

| Formulation                          | Variables  | Constraints | Using big-M coefficients | Network topology |
|--------------------------------------|------------|-------------|---------------------------|------------------|
| continuous-time                      | $O(|V|^2|E|)$ | $O(|V|^2|E|)$ | Yes                       | any              |
| time-window-overlapping              | $O(|V|^2|E|)$ | $O(|V|^3|E|^2)$ | No                        | tree             |
| vehicle-to-time-bucket assignment    | $O(|V||T|+|E||T|)$ | $O(|E||T|)$ | No                        | tree, extendable |

Let $\mathcal{F}^{FRCVP}$ be the set of all feasible schedules to a tree-based FRCVP problem and $\mathcal{F}^{VA}$ be the set of all feasible time-bucket assignments of (VA). We define a mapping $\varphi: \mathcal{F}^{FRCVP} \rightarrow \mathcal{F}^{VA}$ as

$$\varphi(\{s_v: v \in V\}) = \{t_v: v \in V\},$$

such that $t_v$ is feasible to $v$ for any $v \in V$ and $t_u = t_v$ if $s_u = s_v$ for any $u, v \in V$.

**Theorem 2.** The mapping $\varphi$ is surjective. Let $\pi$ be any feasible schedule to the FRCVP problem. Then $\text{Obj}^{FRCVP}(\pi) = \text{Obj}^{VA} \circ \varphi(\pi)$, where $\text{Obj}^{FRCVP}$ and $\text{Obj}^{VA}$ are objective function of the FRCVP problem and (VA), respectively. Let $\chi^*$ be an optimal solution of (VA), then any schedule $\pi^* \in \varphi^{-1}(\chi^*)$ is an optimal solution of the FRCVP problem if it satisfies that $\pi_u^* = \pi_v^*$ when $u$ and $v$ are assigned to the same time bucket by $\chi^*$.

Properties of the three formulations: continuous-time formulation, time-window-overlapping formulation and vehicle-to-time-bucket assignment formulation are compared in Table 4.

### 4. Approximation Algorithms

As an independent interest for understanding the problem structure, hardness and properties, we first show that the tree-based FRCVP problem without any parameter regularization is NP-hard by reducing the problem of scheduling unit execution time tasks with unbounded number of machines to the tree-based FRCVP problem. We then develop three approximation algorithms and analyze their performance with different types of assumption on problem parameters. These approximation algorithms can also show some insights on designing heuristic algorithms for solving the FRCVP in practice. For the development and analysis of approximation algorithms, we consider for the case of a uniform time discretization, which is formally stated in the following assumption.

**Assumption 2.** Assume there exists a bounded universal time range of all possible RTW’s and a universal time discretization such that the time range is uniformly divided into $T$ time buckets labeled as $1, 2, \ldots, T$ such that all vehicles with the relative departure time dropping in a same time bucket are pseudo-platoonable. The total number $T$ of uniform time buckets is considered as a constant throughout the section.

The above assumption aligns with the setting for a plain time-expanded network in literature, and it is useful in the complexity and optimality analysis for the approximation algorithms given later.
assumption is rational in practice. For example, if the scope of the schedule planning is for vehicles of which the departure time is within 12 hours or 24 hours, the universal time range can be set up accordingly. Furthermore, imposing exact simultaneous constraints on the relative departure time for vehicles to be able to form platoons is impractical, and in practice vehicles that enter a shared road segment within a tolerable time range can be coordinated to form a platoon with a short-term speed adjustment. This motivates the specification of a basic time bucket, within which the relative departure time points can be regarded as simultaneous. From Section 4.1 to Section 4.3, the total number of time buckets in the time range is regarded as a constant for the design of approximation algorithms. To better present the results, we focus on the case that the size constraint of a platoon is disregarded, i.e., $\lambda = \infty$. We note that the analysis of the approximation algorithms presented in this section can be straightforwardly extended to the case that $\lambda$ is finite with some minor technical polish.

**Assumption 3.** The size constraint of a platoon is not considered in this section.

The hardness result is given in the following theorem.

**Theorem 3.** The FRCVP problem with the route graph being a tree and with no limit on the platoon size is NP-hard.

### 4.1. A constant-ratio approximation algorithm

Consider the following straightforward greedy algorithm: For each time bucket $t$, assign all vehicles that are feasible at $t$ to $t$. Compute the amount of fuel saving achieved by platooning these vehicles as possible, and take it as the objective of this assignment. Compute the objective for every $t \in [T]$ independently, and take the maximum objective value as the output of the algorithm. The algorithm can guarantee that the output objective is at least $\frac{1}{T} \text{OPT}$, where OPT is the optimal objective of the instance. It turns out that improving the competitive ratio $\frac{1}{T}$ is non-trivial. Achieving a better competitive ratio requires a more sophisticated design of the greedy algorithm and a more delicate analysis, which is presented in this section.

Let $\mathcal{U}$ be a subset of vehicles that are assigned to a same time bucket. Define the function $H(\mathcal{U})$ to be the total fuel saving achieved by these vehicles assuming there is no other vehicles, i.e.,

$$H(\mathcal{U}) = \sum_{e \in E} C_e \phi \circ n_e(\mathcal{U}),$$

(8)

where $n_e$ is the counting operator that maps a set of vehicles to the number of vehicles that share the edge $e$ from the set, i.e., $n_e(\mathcal{U}) = \text{card}\{v \in \mathcal{V} : e \in R_v\}$. The function $\phi : \mathbb{Z}_+ \to \mathbb{R}_+$ is defined as

$$\phi(n) = \begin{cases} 0 & \text{if } n \leq 1, \\ \sigma^l + (n - 1)\sigma^f & \text{if } n \geq 2. \end{cases}$$

(9)
For a given subset \( \mathcal{E}' \) of edges, we define \( H(\mathcal{U}, \mathcal{E}') \) as the total fuel saving restricted to \( \mathcal{E}' \):

\[
H(\mathcal{U}, \mathcal{E}') = \sum_{e \in \mathcal{E}'} C_e \phi(n_e(\mathcal{U})).
\]  

(10)

An advanced greedy algorithm is given as Algorithm 2. The core of Algorithm 2 is the function \( \text{Assignment}(t_1, t_2) \) that focuses on two time buckets \( t_1, t_2 \) and intends to maximize the total fuel saving from the two time buckets in a greedy manner. The problem is how to assign vehicles that are feasible at both \( t_1 \) and \( t_2 \) to either \( t_1 \) or \( t_2 \). The assignment decision of a specific vehicle \( v \) is based on comparing the marginal fuel saving increment \( \Delta_1 = H(V_1 \cup \{v\}, \mathcal{E}_v) - H(V_1, \mathcal{E}_v) \) versus \( \Delta_2 = H(V_2 \cup \{v\}, \mathcal{E}_v) - H(V_2, \mathcal{E}_v) \), where the vehicle sets \( V_1, V_2 \) and the edge set \( \mathcal{E}_v \) are defined in the function. The crucial reason for why the competitive ratio \( 1/T \) is improvable is based on a key observation given in Proposition 4.

**Proposition 4.** Let \( V_1 \) be the subset of vehicles for which the time bucket \( t \) is feasible. Suppose the inequality \( H(V_1) \leq \frac{4}{\sigma^f} \text{OPT} \) hold for all \( t \in [T] \). Then there exists a polynomial algorithm with complexity \( O(|V||T||E|) \) that can identify an optimal solution of the tree-based FRCV problem.

Proposition 4 indicates that we can play a game with the instance generator. If an instance is generated such that the total amount of fuel saving is distributed uniformly across all time buckets, one can easily obtain an optimal solution which yields an competitive ratio equal to 1 in this case. Otherwise, once the instance generator allocates more fuel saving on a certain time bucket, we can obtain a competitive ratio greater than the trivial \( 1/T \).

**Lemma 2.** Let \( t_1 \) and \( t_2 \) be two different time buckets. Let \( V_1 \) (resp. \( V_2 \)) be a subset of vehicles that are only feasible at \( t_1 \) (resp. \( t_2 \)). Let \( V_3 \) be a subset of vehicles that are feasible at both \( t_1 \) and \( t_2 \). Let \( V_3 = U_1 \cup U_2 \) be any 2-partition of \( V_3 \). Let \( \mathcal{E}_0 = \{e \in \cup_{v \in V_3} \mathcal{R}_v : n_e(V_3) \geq 2\} \) be the subset of edges that are shared by at least two vehicles in \( V_3 \). Then the following property holds:

\[
H(V_1 \cup U_1, \mathcal{E}_0) + H(V_2 \cup U_2, \mathcal{E}_0) - H(V_1, \mathcal{E}_0) - H(V_2, \mathcal{E}_0) \leq 2H(V_3, \mathcal{E}_0).
\]  

(11)

**Lemma 3.** Let \( t_1 \) and \( t_2 \) be two time buckets. Let the sets \( V_1, V_2, V_3 \) and \( \mathcal{E}_0 \) be defined the same as in Lemma 2. Let \( H^* = \max_{U \subseteq V_3} H(V_1 \cup U) + H(V_2 \cup V_3 \setminus U) \), and \( H_0 \) be the value returned by the function \( \text{Assignment}(t_1, t_2) \) in Algorithm 2. Then the following bounds hold:

\[
H^* - H_0 \leq 2 \left( 1 + \frac{\sigma^f}{\sigma^l} \right) [H(V_1 \cup V_3, \mathcal{E}_0) + H(V_2 \cup V_3, \mathcal{E}_0) - H(V_1 \cup U_1^*, \mathcal{E}_0) - H(V_2 \cup U_2^*, \mathcal{E}_0)],
\]  

(12)

where \( U_1^* \) be an optimal subset of \( V_3 \) that leads to the optimal value \( H^* \), and \( U_2^* = V_3 \setminus U_1^* \).

**Theorem 4.** Let \( \text{OPT} \) be the optimal value of a tree-based FRCVP instance with \( T \) uniform time buckets. The objective \( H_0 \) given by Algorithm 2 satisfies the following lower bound:

\[
H_0 \geq \left( \frac{4\sigma^l + 6\sigma^f}{4\sigma^f + 5\sigma^l} \right) \frac{1}{T} \text{OPT}.
\]  

(13)
Algorithm 2 A greedy algorithm.

1: **Input**: A set of vehicle $V$, the MST shared by vehicles in $V$, and the feasible set $S_v$ of time buckets for each $v \in V$.
2: Let $W_t$ be the subset of vehicles in $V$ for which $t$ is a feasible time bucket.
3: Let $H_0 \leftarrow \max_{t \in [T]} H(W_t)$.
4: for any two distinct tuple $(t_1, t_2) \in [T] \times [T]$ with $t_1 < t_2$ do
5: Set $H_0 \leftarrow \max\{H_0, \text{ASSIGNMENT}(t_1, t_2)\}$.
6: return $H_0$.

1: **function** ASSIGNMENT$(t_1, t_2)$
2: Let $S_v$ be the set of feasible time buckets for the vehicle $v$.
3: Let $U_1 \leftarrow \emptyset$ and $U_2 \leftarrow \emptyset$.
4: Let $V_1 = \{v \in V : t_1 \in S_v, t_2 \notin S_v\}$, $V_2 = \{v \in V : t_2 \in S_v, t_1 \notin S_v\}$, and $V_3 = \{v \in V : t_1 \in S_v$ and $t_2 \in S_v\}$.
5: Let $E_0 = \{e \in \cup_{s \in V_2} R_s : n_e(V_t) \geq 2\}$.
6: for $v \in U$ do
7: Let $E_v = R_v \setminus E_0$.
8: Compute $\Delta_1 \leftarrow H(V_1 \cup \{v\}, E_v) - H(V_1, E_v)$ and $\Delta_2 \leftarrow H(V_2 \cup \{v\}, E_v) - H(V_2, E_v)$.
9: if $\Delta_1 \geq \Delta_2$ then
10: Assign $v$ to $t_1$, and set $U_1 \leftarrow U_1 \cup \{v\}$.
11: else
12: Assign $v$ to $t_2$, and set $U_2 \leftarrow U_2 \cup \{v\}$.
13: return $H(V_1 \cup U_1) + H(V_2 \cup U_2)$.

Algorithm 3 A greedy algorithm for heuristically solving the FRCVP on a MST.

1: **Input**: A set of vehicle $V$, the MST shared by vehicles in $V$, and the feasible set $S_v$ of time buckets for each $v \in V$.
2: Let $S \leftarrow \cup_{v \in V} S_v$.
3: while $V$ is non-empty do
4: for $t \in S$ do
5: Let $V_t$ be the subset of vehicles for which $t$ is a feasible time bucket.
6: Compute the fuel saving $Q_t$ achieved by vehicles in $V_t$ when they are assigned to $t$.
7: Take an arbitrary $t^*$ from the set $\arg \max_{t \in S} Q_t$, and assign vehicles from $V_{t^*}$ to $t^*$ for the solution.
8: return the solution and objective generated above.

4.2. A polynomial time approximation scheme for FRCVP instances with metric regularization

We consider a family of tree-based FRCVP instances satisfying the following three regularity conditions: (1) The ratio between the longest edge and the shortest edge of $R$ is bounded by $\rho$, i.e., $(\max_{e \in R} C_e) / (\min_{e \in R} C_e) \leq \rho$; (2) The number of edges on the route of each vehicle is bounded by $L$; and (3) The in-degree and out-degree of each node of the MST is bounded by $d$. The regularization parameters $\rho$, $L$ and $d$ are treated as constants in this section. The above three conditions are referred as the regularity conditions for the tree-based FRCVP problem. In this case, there exists a polynomial time approximation scheme (PTAS) for solving the family of tree-based FRCVP
instances to an arbitrary accuracy. The development of the PTAS is based on the following observations:

- Since the total number of time buckets is constant $T$, it implies that if the number $n$ of vehicles sharing an edge is greater than a threshold, the ratio between the amount of fuel saving on that edge by these vehicles in any scheduling and that in an optimal scheduling approaches to 1 as $n$ goes to infinity (Lemma 4).

- The previous observation implies that the heavy-traffic edges can be disregarded (by edge contraction). Then in the updated MST, the number of vehicles on each edge is bounded by a pre-determined constant.

- The MST can be decomposed into sub-trees and the size of each is bounded by a constant (Lemma 6). Since the number of edges in the route of any vehicle is bounded by $L$, the decomposition can guarantee that the amount of potential fuel saving achieved at the connection part of two adjacent sub-trees is bounded by a constant.

The PTAS is given in Algorithm 5, and the functions used in PTAS are given in Algorithm 4. The PTAS consists of three major steps: First, it contracts heavy-traffic edges (edges that shared by a sufficient amount of vehicles) by calling the function \texttt{HeavyTrafficEdgeContraction}($\mathcal{T}, N_0$), which removes all edges from $\mathcal{T}$ that are shared by at least $N_0$ vehicles (Line 3). The output MST is denoted as $\mathcal{T}_{ctr}$. Note that only the heavy-traffic edges are disregarded, but the vehicles that share these heavy-traffic edges will be considered on edges of their routes in $\mathcal{T}_{ctr}$.

Second, it runs a sub-tree decomposition algorithm (\texttt{Decomposition}) along a maximal directed path of $\mathcal{T}_{ctr}$ (Line 5). The decomposition algorithm will be recursively applied to every branch along the maximal directed path. As the output of \texttt{Decomposition}, $\mathcal{T}_{ctr}$ is decomposed into multiple sub-trees stored in \texttt{treeList}. The size of each sub-tree is lower and upper bounded by two pre-determined constants (Lemma 6). Figure 4 illustrates the mechanism of the \texttt{Decomposition} algorithm when it is applied to $\mathcal{T}_{ctr}$ and the branches along the selected maximal directed path.

For every sub-tree, the algorithm then uses a brute-force enumeration to obtain an optimal schedule for vehicles on the sub-tree. The upper bound on the sub-tree size ensures that the complexity of the enumeration is bounded by a constant, while the lower bound on the sub-tree size ensures that the potential underestimation of fuel saving on edges around the conjunction neighborhood between two adjacent sub-trees is just a small fraction.
Algorithm 4 Functions used in the PTAS.

1: function HeavyTrafficEdgeContraction($T, N_0$)
2: $T$ is the original MST and $N$ is an input threshold number such that if an edge $e$ is defined as a heavy-traffic edge if $|V_e| \geq N_0$. Let $EdgeList \leftarrow \emptyset$
3: while There exists a heavy-traffic edge $e = (i, j)$ in $T$ do
4: Contract the edge $e$ by removing $e$ from $T$ and combine the nodes $i$ and $j$.
5: $EdgeList \leftarrow EdgeList \cup \{e\}$.
6: return $[T, EdgeList]$.

---

Algorithm 4 Functions used in the PTAS.

1: function Enumeration($T$, $U$, $S$)
2: The notations $T$, $U$ and $S$ are similar as in Decomposition.
3: Enumerate all possible choices of the relative departure time for all vehicles in $U$.
4: Compute the objective value at $T$ achieved by every schedule constructed from the enumeration.
5: Let $obj^*$ be the maximum objective value found in the enumeration.
6: Assign the optimal schedule to vehicles in $U$ that yields $obj^*$.
7: Return $obj^*$.
Algorithm 5 A polynomial time approximation scheme (PTAS) for solving FRCVP with metric regularization.
1: Input: A FRCVP instance $(T, V, S)$, two threshold integers $N_0$ and $N_1$.
2: Output: A feasible solution of the FRCVP instance.
3: Let $[\mathcal{T}_{ctr}, \text{EdgeList}] \leftarrow \text{HeavyTrafficEdgeContraction}(T, N_0)$.
4: Find a maximal directed path $P$ in $\mathcal{T}_{ctr}$, i.e., $P$ cannot be extended by adding more directed edges from $\mathcal{T}_{ctr}$.
5: Let $\mathcal{treeList} \leftarrow \text{Decomposition}(P, \mathcal{T}_{ctr}, N_1)$. Let $U_0 \leftarrow \emptyset$.
6: for every tree $T' \in \mathcal{treeList}$ do
7: $\text{Enumeration}(T', \mathcal{V}(T') \setminus U_0, S)$.
8: $U_0 \leftarrow U_0 \cup \mathcal{V}(T')$.
9: for every vehicle of which the departure time has not been determined, do
10: Assign an arbitrary feasible departure time to the vehicle.
11: return the objective corresponding to the above schedule.

We first provide some auxiliary results for analyzing the performance of Algorithm 4. Lemma 4 guarantees that the optimality gap is small for an edge if the number of vehicles sharing the edge is large enough. This result drives the motivation of the heavy-traffic edge contraction.

**Lemma 4.** Suppose Assumption 2 hold. Let $\pi$ and $\pi^*$ be an arbitrary schedule and an optimal schedule, respectively. For an edge $e \in \mathcal{R}$, let $\text{obj}(\pi, e)$ and $\text{obj}(\pi^*, e)$ be the objective value (the
amount of fuel saving) contributed by $e$ under the schedule $\pi$ and $\pi^*$, respectively. If the edge $e \in \mathcal{R}$ is shared by $N$ vehicles, where $N \geq T + 1$, then $\frac{\text{obj}(\pi,e)}{\text{obj}(\pi^*,e)} \geq 1 - \frac{T}{N}$.

A FRCVP problem instance on a MST $T$ can be naturally decomposed into two independent problem instances if there exist a node $i$ and a partition of the (directed) edges adjacent to $i$ into two subsets $E_1$ and $E_2$ such that the route of any vehicle can only intersect with at most one of $E_1$ and $E_2$. Since this trivial decomposition is not of our interest, we focus on the analysis for a MST that is inseparable which is given by the following definition.

**Definition 11.** Let $T$ be a MST, $U$ be the set of vehicles sharing $T$, and $S_v$ be the set of feasible time buckets for a vehicle $v \in U$. The FRCVP instance specified by $(T,U,S)$ is **separable** if $U$ can be partitioned into two non-empty subsets $U_1$ and $U_2$ such that for any $u \in U_1$ and $v \in U_2$ either $R_u \cap R_v = \emptyset$ or $S_u \cap S_v = \emptyset$. Otherwise, the instance is **inseparable**.

**Corollary 2.** If an instance $(T,U,S)$ is inseparable, then for any vehicle $u \in U$ there exists a vehicle $v \in U \setminus \{u\}$ such that both $R_u \cap R_v$ and $S_u \cap S_v$ are non-empty.

The following lemma provides a lower bound for the objective value achieved on an inseparable MST. The lower bound is proportional to the number of vehicles sharing the MST. This lemma will be applied to every sub-tree decomposed from $T_{ctr}$, and eventually it leads to the conclusion that the potential underestimation of fuel saving at the conjunction neighborhood is just a small fraction of the total saving.

**Lemma 5.** Suppose the regularity condition (2) hold. Let $T$ be a MST, $U$ be the set of vehicles sharing $T$, and $S_v$ be the set of feasible time buckets for a vehicle $v \in U$. Suppose $|V_e| \leq N_0$ hold for every $e \in \mathcal{E}(T)$, and the instance $(T,U,S)$ is inseparable. Then a vehicle-pairing algorithm applied on $U$ can lead to an objective value $\text{obj}(T)$ restricted on $T$ satisfying

$$\text{obj}(T) \geq \frac{(\sigma^i + \sigma^f)C_{min}}{2LN_0(LN_0 + 1)}|U|.$$  \hspace{1cm} (14)

The following lemma provides a lower and a upper bound on the size of each sub-tree decomposed from $T_{ctr}$. In Theorem 5, the lower bound is used to guarantee that the optimality gap is small and the upper bound is used to bound the computational complexity of the PTAS.

**Lemma 6.** Suppose the regularity condition (3) given at the beginning of Section 4.2 hold. When the condition at Line 15 of the Decomposition function in Algorithm 5 is satisfied, the cardinality of $\mathcal{E}(T')$ must satisfy $N \leq |\mathcal{E}(T')| < 3dN$.

By utilizing all the above technical results, the following theorem summarizes the performance of the PTAS on the competitive ratio and complexity.
Theorem 5. Suppose Assumption 2 and regularity conditions (1)-(3) hold for a FRCVP instance $(\mathcal{T}, \mathcal{V}, \mathcal{S})$, where $\mathcal{T}$ is a MST, $\mathcal{V}$ is the set of all vehicles sharing $\mathcal{T}$, and $\mathcal{S} = \{S_v : v \in \mathcal{V}\}$ is the collection of the feasible set of time buckets for each vehicle. Apply Algorithm 5 to the instance. If $N_0$ and $N_1$ in the algorithm are selected such that $N_0 \geq (1 + 1/\epsilon)T$ and $N_1 \geq 5L^4N_0^3\rho/\epsilon$, the objective returned by the algorithm satisfies $\text{obj} \geq (1 - \epsilon)\text{Opt}$. The complexity of the algorithm is $|E| \cdot O\left(\frac{\rho d L_4}{\epsilon^5}\right)$, where $E$ is the set of edges in $\mathcal{T}$.

4.3. Asymptotic optimality for vehicle-saturated FRCVP instances

We consider a family of FRCVP instances in which the number of vehicles in the system is much more than the number of edges of the road network shared by the vehicles. We refer them as vehicle-saturated instances. It can be shown that in this extremal case, a linear relaxation of the formulation can lead to a randomized algorithm that makes the relative optimality gap goes to zero as the ratio between the number of vehicles and the number of edges goes to infinity. To gain some intuition, notice that the average number of vehicles on an edge is sufficiently large in a vehicle-saturated FRCVP instance, which leads to a very small relative optimality gap as shown in Lemma 4. Theoretically, this result reveals that as long as the number of vehicles grows faster than the number of edges in the network, a linear-program based algorithm can yield a near-optimal solution to the problem instance. The approximation algorithm for vehicle-saturated FRCVP instances is based on the following linear program relaxation of FRCVP:

$$\begin{align*}
\text{max} & \quad \sum_{e \in \mathcal{R}} \sum_{t \in \mathcal{S}} (\sigma C_e y'_{e,t} + \sigma C_e w_{e,t}) \\
\text{s.t.} \quad & \sum_{t \in \mathcal{S}} x_{u,t} = 1 \quad \forall u \in \mathcal{V}, \quad (LP.2) \\
& y'_{e,t} \geq x_{u,t} + x_{v,t} - 1 \quad \forall (u,v) \in \mathcal{V}_{e,t} \forall t \in \mathcal{S}, \quad (LP.3) \\
& 2y'_{e,t} \leq \sum_{u \in \mathcal{V}_{e,t}} x_{u,t} \quad \forall t \in \mathcal{S}, \quad (LP.4) \\
& w_{e,t} = \sum_{u \in \mathcal{V}_{e,t}} x_{u,t} - y_{e,t} \quad \forall e \in \mathcal{R}, \forall t \in \mathcal{S}, \quad (LP.5) \\
& y_{e,t} \geq x_{u,t} \quad \forall u \in \mathcal{V}_{e,t} \forall t \in \mathcal{S}, \quad (LP.6) \\
& 0 \leq x_{u,t}, y_{e,t}, y'_{e,t} \leq 1, \quad \forall e, u, t. \quad (LP.7)
\end{align*}$$

Notice that the decision variables $x_{u,t}$, $y_{e,t}$ and $y'_{e,t}$ are relaxed to take continuous values in the interval $[0,1]$. The constraint $(LP.3)$ ensures that for an edge $e$ if there are at least two vehicles sharing $e$ assigned to the same time bucket $t$, $y'_{e,t}$ should be equal to 1. This variable indicates wether
there exists a non-trivial platoon (size at least 2) on \( e \) at \( t \). The constraint (LP.4) is used to evaluate the number of following vehicles for the edge-time-bucket pair \((e, t)\) if the platoon is non-trivial. Our LP-based randomized algorithm is straightforward: Step 1. Solve the linear program (LP) and let \( x^* = \{x^*_u, t : u \in V, t \in S_u\} \) be the \( x \)-vector of optimal solution; Step 2. For each vehicle \( u \in V \), assign it to the time bucket \( t \) with probability \( x^*_u, t \). The performance analysis of the randomized algorithm when applied to the vehicle-saturated instances is given by the following theorem.

**Theorem 6.** Suppose the Assumption 2 and the regularity condition (1) hold. Let \( V \) be the set of vehicles and \( R \) be the set of edges in the MST shared by vehicles in \( V \). Let \( \text{obj}(x^*) \) be the objective value of the schedule obtained from the randomized algorithm, and \( \text{OPT} \) be the optimal value of the instance. If the ratio \( \frac{|V|}{|R|} \to \infty \), then \( \frac{\sum \text{obj}(x^*)}{\text{OPT}} \to 1 \).

### 5. Extension to General Cases

The analysis given in previous sections is valid for the case that the graph induced by routes of all vehicles is a directed forest, i.e., it consists of multiple unconnected maximal shared trees. It turns out that the formulation idea can be modified and extended to a general case where the graph of routes may have loops (after disregarding the edge direction). To see why loops can exist in optimal routes, we analyze an example given in Figure 5(a). In this example, vehicles \( v_1, v_2 \) and \( v_3 \) form a platoon on the edge \( AB \), and then \( v_1 \) leaves the platoon at the node \( B \) and travels through the path \( B \to F \to G \) in order to platoon with the vehicle \( v_4 \) along the path \( F \to G \to C \to D \) while \( v_2 \) and \( v_3 \) remain platooned long the path \( B \to C \to D \). The platoon of \( v_1 \) and \( v_4 \) is unlikely to be able to merge with that of \( v_2 \) and \( v_3 \) on the edge \( CD \) as their paths are different. This solution can be better than the one in which \( v_1, v_2 \) and \( v_3 \) are platooned through the path \( A \to B \to C \to D \) but \( v_4 \) is left to be un-platooned.

When loops are involved in the graph of routes, the way of defining a root for the vehicle system and the relative time window for each vehicle becomes invalid. For example, consider the case in Figure 5(a). Suppose \( A \) is the origin of \( v_1 \sim v_3 \), and \( F \) is the origin of \( v_4 \). In this case, we can let \( v_1 \) platoon with \( v_2 \) either on \( AB \) or \( CD \). The condition for them to platoon on \( AB \) is \( t_{v_1, A} = t_{v_2, A} \), while the condition for platooning on \( CD \) is \( t_{v_1, A} + T_{BF} + T_{FG} + T_{GC} = t_{v_2, A} + T_{BC} \). The two conditions can not be met at the same time in general. This is in contrast different from the case of MST, in which the shared edges by any two vehicle form a single path and there exists a unique platooning condition for the two vehicles on every edge they share. It turns out that this difficulty can be handled for general cases with loops if a certain conditions on the network and vehicle time windows are satisfied.
5.1. Illustration via a concrete example

We first provide an example-based description of the modification to deliver some insight, and then present the rigorous loop-break scheme in Section 5.2 that reduces a general case to a MST based instance. Focusing on the case in Figure 5(a), one can first generate an arbitrary directed spanning tree of the graph and pick an arbitrary node from the tree that has no parent. As an example, the spanning tree can be set as the one defined by all the edges excluding $BF$, and $A$ can be set as the root. For vehicles $v_2$, $v_3$ and $v_4$, their relative time windows can be defined the same as in Section 2 by taking the spanning tree as the MST. Since adding $BF$ introduces a cycle, which leads to two different conditions for platooning that involves $v_1$, one way for resolving this technical difficulty is to create two copies of the vehicle $v_1$ denoted as $v_1'$ and $v_1''$. One can let $v_1'$ (resp. $v_1''$) be defined only on $A \rightarrow B$ (resp. $B \rightarrow F \rightarrow G \rightarrow C \rightarrow D$). In other words, the path $A \rightarrow B$ (resp. $B \rightarrow F \rightarrow G \rightarrow C \rightarrow D$) can be viewed as the scope of $v_1'$ (resp. $v_1''$). This implies that the departure node of $v_1''$ is set at $B$. Suppose $T_{v_1}^O$ and $T_{v_1}^D$ be the earliest possible departure time and deadline for the destination time of $v_1$, the lower and upper bounds on the departure time of $v_1'$ and $v_1''$ can be set respectively as

$$t_{v_1,A}' = T_{v_1}^O, \quad t_{v_1,A}'' = T_{v_1}^D - T_{AB} - T_{BF} - T_{FG} - T_{GC} - T_{CD},$$
$$t_{v_1,B}' = T_{v_1}^O + T_{AB}, \quad t_{v_1,B}'' = T_{v_1}^D - T_{BF} - T_{FG} - T_{GC} - T_{CD}. \quad (16)$$

To construct the RTW’s of $v_1'$ and $v_1''$, we can image that the node $B$ is split into two different nodes $B'$ and $B''$ as shown in Figure 5(b) with the following settings: no edge between $B'$ and $B''$, $B'$ is on the path $A \rightarrow B' \rightarrow C$, $B''$ is on the path $B'' \rightarrow F \rightarrow G$, and $T_{B''F} = T_{BF}$. Then the RTW’s of $v_1'$ and $v_1''$ are constructed as before by applying Definition 6 of relative time on the tree in Figure 5(b). Specifically, the RTW of $v_1'$ is just

$$\text{RTW}_{v_1}' = [t_{v_1,A}', t_{v_1,A}'] = [T_{v_1}, T_{v_1}^D - T_{AB} - T_{BF} - T_{FG} - T_{GC} - T_{CD}]$$

and the RTW of $v_1''$ is given by

$$\text{RTW}_{v_1}'' = [t_{v_1,B}', T_{B''F} + T_{FG} + T_{GC} - T_{B'C} - T_{AB'}, t_{v_1,B}'' = T_{v_1}^O + T_{BF} + T_{FG} + T_{GC} - T_{B'C} - T_{AB}]$$
$$= [T_{v_1}^O + T_{BF} + T_{FG} + T_{GC} - T_{BC}, T_{v_1}^D - T_{AB} - T_{BC} - T_{CD}].$$

One can verify that $\text{RTW}_{v_1}'$ and $\text{RTW}_{v_1}''$ have an equal length. Since $v_1'$ and $v_1''$ are virtual copies of $v_1$, one should impose the following additional constraint for time consistency:

$$t_{v_1}' - \text{RTW}_{v_1}' = t_{v_1}'' - \text{RTW}_{v_1}'', \quad (17)$$

where $t_{v_1}'$ and $t_{v_1}''$ are feasible time instant of $v_1'$ and $v_1''$ within the range of their RTW’s respectively, $\text{RTW}_{v_1}' = T_{v_1}^O$ is the lower bound of $\text{RTW}_{v_1}'$ and similarly for $\text{RTW}_{v_1}''$. Notice that adding the
constraint (17) will introduce time-instant variables which will eventually lead to constraints with big-M coefficients as (CT.5) and (CT.6) used by the formulation developed in (Luo and Larson 2020). To obtain a big-M free formulation, we should figure out an equivalent form of (17) using the notion of time buckets and vehicle-to-time-bucket assignment. This can be achieved in a few steps.

In Step 1, we construct the time buckets for the vehicle set \{v_1', v_1'', v_2, v_3, v_4\} using Algorithm 1. Then map the break points from RTW\_v_1' to RTW\_v_1'' according to (17) and vice versa, which refines the time buckets on the two RTW’s. To be more specific, suppose the RTW’s of \(v_1', v_1'', v_2, v_3\) and \(v_4\) are given in Figure 6 as the black line segments from the bottom to the top, respectively. We first apply Algorithm 1 to these RTW’s, which generates the set of time buckets \\{(a_1a_2a_3), (b_1b_2), (c_1c_2), (d_1d_2), e, f, g, h\} where \(a_1, a_2\) and \(a_3\). All the projected points generated by the algorithm are in blue color. Notice that there are 2 intermediate break points projected on RTW\_v_1', which yield 3 time buckets: \((a_1a_2a_3), (b_1b_2)\) and \((c_1c_2)\). Similarly, 3 intermediate break points on RTW\_v_1'' lead to 4 time buckets: \((b_1b_2), (c_1c_2), (d_1d_2)\) and \(e\).

In Step 2, we impose time consistency between \(v_1'\) and \(v_1''\). To this end, the break point separating \(a_3\) and \(b_1\) from RTW\_v_1' is mapped to its counterpart at RTW\_v_1'' following the law of (17) which separates \(c_1\) and \(c_2\), and the one separating \(b_2\) and \(c_1\) is mapped to the one separating \(d_1\) and \(d_2\). Similarly, the point separating \(b_2\) and \(c_1\) on RTW\_v_1' is mapped to the one separating \(a_2\) and \(a_3\) on RTW\_v_1'', the point separating \(c_2\) and \(d_1\) is mapped to the one separating \(b_1\) and \(b_2\), and the point separating \(d_2\) and \(e\) is mapped to the one separating \(c_1\) and \(c_2\). All the (shifting) projected points generated in this step are in red color.

In Step 3, all the newly generated points in Step 2 are vertically projected to the RTW of every (real and virtual) vehicle. These points are in green color. To impose time consistency again, the projected green point that separates \(b_1\) and \(b_2\) on RTW\_v_1'' is further projected to the black point that separates \(a_1\) and \(a_2\) on RTW\_v_1'. Eventually, the vehicle-to-time-bucket assignment constraints and time-consistency constraints can be formulated as follows:

\[
\begin{align*}
x_{v_2,b_1} + x_{v_3,b_2} + x_{v_2,c_1} + x_{v_2,c_2} + x_{v_2,d_1} + x_{v_2,d_2} + x_{v_2,e} + x_{v_2,f} + x_{v_2,g} &= 1, \\
x_{v_3,c_1} + x_{v_3,c_2} + x_{v_3,d_1} + x_{v_3,d_2} + x_{v_3,e} + x_{v_3,f} &= 1, \\
x_{v_4,e} + x_{v_4,f} + x_{v_4,g} + x_{v_4,h} &= 1, \\
x_{v_1',a_1} + x_{v_1',a_2} + x_{v_1',a_3} + x_{v_1',b_1} + x_{v_1',b_2} + x_{v_1',c_1} + x_{v_1',c_2} &= 1, \\
x_{v_1'',b_1} + x_{v_1'',b_2} + x_{v_1'',c_1} + x_{v_1'',c_2} + x_{v_1'',d_1} + x_{v_1'',d_2} + x_{v_1'',e} &= 1, \\
x_{v_1',a_1} &= x_{v_1'',b_1}, x_{v_1',a_2} = x_{v_1'',b_2}, \\
x_{v_1',a_3} &= x_{v_1'',c_1}, x_{v_1',b_1} = x_{v_1'',c_2}, \\
x_{v_1',b_2} &= x_{v_1'',d_1}, x_{v_1',c_1} = x_{v_1'',d_2}, \\
x_{v_1',c_2} &= x_{v_1'',e} \\end{align*}
\]

(imposing time consistency).
Similarly as in (VA), the count of vehicles on each edge and time bucket can be written in terms of the above variables. For example, the number of vehicles on the edge \( CD \) at the time bucket \( e \) has the expression 
\[
x_{v_2,e} + x_{v_3,e} + x_{v_4,e} + x_{v'_1,e}.
\]
Then a vehicle-to-time-bucket assignment formulation for this illustrative example can be established similarly as (VA) but with extra constraints of time consistency.

5.2. A loop-break scheme and formulation for general cases

As illustrated in Section 5.1, the key step that can lead to a vehicle-to-time-bucket assignment formulation for a general case is to break a loop by splitting the start node of a maximal deviated path (See Definition 12), and create virtual copies of vehicles that have traversed through edges on the maximal deviated path. In general, the graph of vehicle routes can have multiple maximal deviated paths. A procedure of systematic loop breaking, virtual-vehicle generation, break point projection and time bucket generation is given in Algorithm 6. Notations related to the output of the algorithm are defined in Table 5. If Algorithm 6 is applied to the instance given in Figure 5(a), some of the quantities in Table 5 are given as:

\[
\begin{align*}
V_{\text{real}} &= \{v_2, v_3, v_4\},
V_{\text{vtl}} &= \{v'_1, v''_1\},
V_{\text{ext}} &= V_{\text{real}} \cup V_{\text{vtl}},
\mathcal{M}(v_1) &= \{v'_1, v''_1\} \text{ and } \mathcal{K}(v_1, 1, 2) &= \{(a_1, b_1), (a_2, b_2), (a_3, c_1), (b_1, c_2), (b_2, d_1), (c_1, d_2), (c_2, e)\}.
\end{align*}
\]

It is important to point out that Algorithm 6 may not terminate in general since the projection in the vertical and shifting modes may repeat for an infinite number of iterations. A sufficient condition for it to terminate is presented in Proposition 5.

**Definition 12.** For a given graph of routes and a predefined maximal spanning tree \( \mathcal{T} \), a directed path \( P \) is **maximal deviated** if \( P \) does not have a shared edge with \( \mathcal{T} \), and both the start and end nodes of \( P \) are on \( \mathcal{T} \).

**Proposition 5.** Let \( G \) be the group generated by the elements \( \{0\} \cup \{a_u, b_u : u \in V_{\text{ext}}\} \) under a finite number of plus and minus operations, where \( a_u = \text{RTW}_u \) and \( b_u = \text{RTW}_u \) are the coordinates of the lower and upper bounds of RTW. If \( G \) is isomorphic to a one-dimensional lattice group, running Algorithm 6 on this instance will terminate in a finite number of iterations, and hence the corresponding general FRCVP instance admits a valid vehicle-to-time-bucket assignment formulation given in (GVA).

**Remark 2.** A sufficient condition for \( G \) being a lattice group is that all the end points \( a_u, b_u \) are multiples of a time unit, e.g., second or minute.

After running Algorithm 6 that converts a graph with loops into a tree, and with the help of the quantities returned by the algorithm, it is ready to formulate a MILP for the general instance of FRCVP given that the condition in Proposition 5 holds:

\[
\max \sum_{e \in \mathcal{E}} \sum_{t \in \mathcal{S}} (\sigma^i C_e z_{e,t} + \sigma^j C_e y'_{e,t} + \sigma^f C_e w_{e,t})
\]
Table 5  Notations of quantities returned by Algorithm 6.

| Notation | Definition |
|----------|------------|
| $T_{ext}$ | the maximal shared tree generated after loop breaking; |
| $V_{real}$ | the subset of vehicles in $V$ that does not have virtual copy; |
| $V_{vtl}$ | the subset $V \setminus V_{real}$; |
| $n_v$ for $v \in V_{vtl}$ | the number of virtual copies of $v \in V_{vtl}$; |
| $M$ | a mapping defined as $M(v) = \{v^{(1)}, \ldots, v^{(n_v)}\}$ which is the set of virtual copies of $v$ for a $v \in V_{vtl}$ and $M(v) = \{v\}$ for $v \in V_{real}$; |
| $V_{ext}$ | the extend set of vehicles, e.g., $V_{ext} = \cup_{v \in V_{vtl}} M(v)$; |
| $S_u$ | the subset of feasible time buckets for $u \in V_{ext}$; |
| $S$ | the set of all time buckets generated, i.e., $S = \cup_{v \in V_{ext}} S_v$; |
| $K(v, i, j)$ for $v \in V_{vtl}, i < j \in [n_v]$ | the set of time-consistent time-bucket pairs for $v^{(i)}$ and $v^{(j)}$. |

\[ \text{s.t. } \sum_{t \in S_u} x_{u,t} = 1 \quad \forall u \in V_{ext}, \]

\[ x_{v^{(i)}, s} = x_{v^{(j)}, t} \quad \forall v \in V_{vtl}, \forall i < j \in [n_v], \forall (s, t) \in K(v, i, j), \quad (GVA) \]

where the notations of decision variables are the same as (VA) with $V_{ext}$ in this case being the set of vehicles in $V_{ext}$ that are feasible at the time bucket $t$. The second line of constraints impose the time consistency among all virtual copies of a vehicle in $V_{vtl}$.

5.3. A heuristic method for general cases based on loop breaking

In general, the lattice-group isomorphic condition in Proposition 5 may not hold. Even if it holds, the number of time buckets generated by Algorithm 6 could be large. However, inspired by the loop-break idea, some high quality and practical heuristic methods can be developed. To give an example, we notice that the complexity is brought by the enforcement of time consistency among virtual copies of a same physical vehicle when traveling along different deviated paths. The complexity can be naturally resolved if each vehicle in $V_{vtl}$ is bound with a fixed virtual copy, and the potential fuel saving that may be achieved by other virtual copies of the vehicle is abandoned. To be more specific, consider again the instance given by Figure 5 and 6. For the vehicle $v_1 \in V_{vtl}$, one could only consider the platooning opportunities of $v_1$ along the path $B \rightarrow F \rightarrow G \rightarrow C \rightarrow D$ but disregarding those when it travels along $A \rightarrow B$, or verse vice. This is equivalent to selecting only one virtual copy from $\{v'_1, v''_1\}$ as the realization of $v_1$. In summary, the guideline is that for every $v \in V_{vtl}$, select only one of its virtual copy and then the adaptive time discretization is straightforwardly applicable. The selection rule could be based on the potential fuel saving of a virtual vehicle when traveling along its path of scope.
Algorithm 6 A preprocessing algorithm for a general FRCVP instance with loops.

1: **Input:** The graph $G$ of vehicle routes, a predefined maximal spanning tree $T$ and time bounds $T^O_v, T^D_v$ for all $v \in V$.

2: **Output:** A maximal shared tree, an extended set of vehicles and feasible time buckets for each vehicle.

3: **Step 1.** Loop breaking and virtual vehicle generation:
   4: Partition the set $E(G) \setminus E(T)$ of edges into a set $D$ of maximal deviated paths.
   5: for every maximal deviated path $P$ from $D$ do
   6:   Break the corresponding loop by splitting the start node of $P$ as illustrated in Figure 7.
   7: for every $v$ such that $R_v \cap P \neq \emptyset$ do
   8:   Create an extra virtual copy of $v$ and add it to the current extended set of vehicles.

9: Generate the sets $T_{\text{ext}}, V_{\text{real}}, V_{\text{ext}}$, and $M$. (See Table 5 for the definitions.)

10: **Step 2.** Time bucket generation:
11: for $u \in \bigcup_{v \in V} M(v)$ do
12:   Compute the relative time window $RTW_u$, reference to the root of $T_{\text{ext}}$ following Definition 6.
13: Apply Algorithm 1 to the collection $\{RTW_u : u \in \bigcup_{v \in V} M(v)\}$ of RTW’s for the extended set of vehicles.
14: Let $\Delta_u$ be the set of break points (including two end points) generated by Algorithm 1 at $RTW_u$ for $u \in \bigcup_{v \in V} M(v)$.
15: Set $\text{flag} \leftarrow 1$ and projMode $\leftarrow$ shift.
16: for $u \in V_{\text{ext}}$ do
17:   if $u \in V_{\text{real}}$ then set $\Delta_u' \leftarrow \emptyset$; else set $\Delta_u' \leftarrow \Delta_u$.
18: while $\text{flag} = 1$ do
19:   if projMode $= \text{shift}$ then
20:     for $v \in V_{\text{ext}}, i \in [n_v]$ and $j \in [n_v] \setminus \{i\}$ do
21:       $\text{PROJECTION}(\Delta_{v(i)}, v, v(i), \text{projMode}).$
22:   else
23:     for $u \in V_{\text{ext}}$ and $w \in V_{\text{ext}} \setminus \{u\}$ do
24:       $\text{PROJECTION}(\Delta_u', w, u, \text{projMode}).$
25:   Set $\Delta_u'$ to be the set of current distinct break points on RTW$_{u(i)}$.
26: if no additional break points are generated on any RTW then $\text{flag} \leftarrow 0$ and break;
27: else If projMode $= \text{shift}$, set projMode $\leftarrow$ vertical, otherwise set projMode $\leftarrow$ shift.
28: Return $T_{\text{ext}}, V_{\text{ext}}$, and $\{S_u : u \in V_{\text{ext}}\}$.

1: **procedure** $\text{PROJECTION}(A, u, v, \text{projMode})$
2: **Input:** $u$ and $v$ are two vehicles (can be virtual) $A$ is a subset of break points on RTW$_u$. If projMode $= \text{shift}$ the projection is in the shifting mode defined as in (17), otherwise it is in the vertical mode.
3: **purpose:** Project a subset $A$ of break points from RTW$_u$ to RTW$_v$ in the mode specified by projMode.
4: for $a \in A$ ($a$ is the coordinate of the break point) do
5: if projMode $= \text{shift}$ then
6:   Add a break point to RTW$_v$ at the coordinate $a - \text{RTW}_u + \text{RTW}_v$ if it does not exist.
7: else if $\text{RTW}_u \leq a \leq \text{RTW}_v$ then
8:   Add a break point to RTW$_v$ at the coordinate $a$ if it does not exist.
Figure 5  Example of a graph of routes that has a loop. (a) The nodes (B, F, G, C) form a loop in the undirected counterpart of the route-network graph. (b) Split node B into $B'$ and $B''$ to break the loop and create two virtual vehicles $v'_1$ and $v''_1$.

Figure 6  Construction of time buckets for vehicles $v'_1$, $v''_1$, $v_2$, $v_3$ and $v_4$ based on the tree given in Figure 5(b). The non-vertical parallel dashed lines indicate that the time consistency is imposed between schedules of $v'_1$ and $v''_1$.

Figure 7  Illustration of the loop-break preprocess. Part (a) gives a directed graph formed by the routes of all vehicles. The predefined maximal spanning tree is built by the solid lines, and the dashed lines are edges and paths that do not align with the spanning tree. Note that each (dashed and solid) line can be a path that consists of multiple edges. Part (b) illustrates that after running the loop-break algorithm, the starting node of every maximal derived path is split out and the resulting graph is a directed tree.
6. Numerical Investigation

The proposed mixed-integer linear programs (TWOF) and (VA) are implemented and solved using the Gurobi-Python API. Their computational performance of solving FRCVP using the two reformulations is tested against the MILP established in (Luo and Larson 2020) (also given in the Appendix to be self-contained) that is based on a continuous-time formulation with big-M coefficients involved. Besides the new reformulations, the following algorithms developed in this paper are implemented in Python 3.8: the adaptive time discretization algorithm (Algorithm 1) to generate the feasible time buckets for each vehicle, the greedy algorithm (Algorithm 2) that assigns vehicles to time buckets, and a simple randomized algorithm based on the solution of linear relaxation (LP). The later two are the approximation algorithms for solving tree-based FRCVP instances heuristically. The set of time buckets generated by Algorithm 1 has been used in the vehicle-to-time-bucket assignment formulation (VA), Algorithm 2, and the formulation (LP). The Algorithm 5 that provides a polynomial time approximation scheme for solving FRCVP instances with certain regularization on the network structure is of prohibitive computational complexity despite of being polynomial in problem parameters and the optimality tolerance, and hence it is more of a theoretical result than an algorithm with practical value. Therefore, it is not investigated numerically. Different numerical experiments have been conducted to test the computational performance and tightness of proposed reformulations, and the impact of the RTW size on the amount of fuel saving and the number of time buckets.

The investigation is based on problem instances generated from three networks. The generation of networks and numerical instances are described in Section 6.1. In Section 6.2, we present the comparison of computational performance of solving FRCVP instances via the formulations developed in this paper versus the continuous-time formulation given in the Appendix. Numerical comparison of the two approximation algorithms is presented in Section 6.3. In Section 6.4, we provide the numerical results of the objective (the amount of fuel saving) and the total number of time buckets at different size of vehicle RTW.

6.1. Problem instance generation

A tree-based FRCVP problem instance consists of four parts: the road network topology, the origin and destination nodes of each vehicle, and the origin and destination time of each vehicle (i.e., \( T_v^O \) and \( T_v^D \)). We create three road networks as the base for all problem instances. The first one is an artificial network that is generated using an attachment random graph model. The network generation process starts from a single node locating at \((0,0)\) and an edge is added to a randomly selected node (named as an active node) that is still available for edge attachment with a random
orientation while maintaining the network to be a tree. A node is considered to be available for edge attachment if its current degree is no more than three. The length of an edge follows the uniform distribution on $[1, 1.5]$. The orientation of a newly added edge is chosen from the list of angles: $0, \pm \pi/6, \pm \pi/3, \pm 2\pi/3, \pm 5\pi/6, \pi$, with respect to the node, by following a carefully defined probability mass function. The probability mass function depends on the position of the active node. In particular, it is generated such that if an active node locates on the positive side of the vertical axis, the probability mass on angles $(\pi/6, \pi/3, 2\pi/3, 5\pi/6)$ will be greater than other angles. Similarly, if the active node locates at the positive side of the horizontal axis, the probability mass on angles $(0, \pm \pi/6, \pm \pi/3)$ will be greater. This network is denoted as ArtifNet in the section, and the network structure is shown in Figure 12(a), which consists of 100 directed edges.

The second and third road networks are both based on the Chicago high-way network in reality. The third road network referred as Chicago-full (CF) is the Chicago high-way network itself, shown in Figure 12(e). The network has about 36K nodes and 48K edges. The second road network is a simplified version of the Chicago-full network by imposing clustering on it. Specifically, we apply a $K$-mean clustering algorithm with $K = 100$ to the nodes of Chicago-full network with the distance of two nodes measured by the $\ell_2$ norm of the two positions. So all nodes from the Chicago-full network are partitioned into 100 clusters, and then for nodes in a same cluster, a representative node is created at the geometric center of nodes in the cluster. The 100 representative nodes are connected with newly generated edges to form a planar graph shown in Figure 12(c), and it is referred as the Chicago-cluster (CC) network in the section.

The second part of creating a problem instance is to specify the number of vehicles in the system and generate the origin and destination nodes of each vehicle. We consider five options for the number $N$ of vehicles: 100, 200, 300, 400 and 500. For a given network and vehicle number $N$, we select $N$ pairs of origin and destination nodes, each for a vehicle. Note that vehicles could share a same origin (resp., destination) node. For the ArtifNet network, the origin (resp., destination) nodes are randomly selected from the negative (resp., positive) side of horizontal axis. For the Chicago-cluster and Chicago-full networks, the origin (resp., destination) nodes are randomly selected from the Northern (resp., Southeastern) part of the network. We set the route of a vehicle to be the shortest path from the origin to the destination in all the numerical experiments investigated in this paper. Obviously there are many ways for the route selection. In particular, the routes can also be taken as an optimal solution of the coordinated vehicle platooning problem without time constraints, i.e., the solution of (RDP) in (Luo and Larson 2020). An example of origin and destination nodes for a 100-vehicle system is given in Figure 12(b) for the ArtifNet network, Figure 12(d) for the Chicago-cluster network, and Figure 12(f) for the Chicago-full network, respectively. Restricting
origin (resp., destination) nodes to a certain portion of the network can potentially induce large overlap of vehicle routes, which is in favor of coordinated platooning.

The last step is to generate origin and destination time (i.e., $T^O_v$ and $T^D_v$) for each vehicle. To simplify the implementation, we assume that the travel speed of every vehicle on every road link is a constant, and hence the traversing time of a road link can be simply set to be the length of the road link. In order to generate reasonable origin and destination time for each vehicle, we first compute the mean value $\bar{L}$ of route length over all vehicles, i.e., $\bar{L} = (\sum_{v=1}^{N} L_v)/N$, where $L_v$ is the route length of vehicle $v$. Based on two input parameters $\gamma_{\text{full}}$, and $\gamma_{\text{ext}}$ ($\gamma_{\text{full}} > \gamma_{\text{ext}}$), we generate $T^O_v$ and $T^D_v$ independently for each vehicle by the following two steps: First, draw a sample from the uniform distribution $U[0, \gamma_{\text{full}}\bar{L}]$ and set it to be $T^O_v$, and then we set $T^O_v \leftarrow T^O_v + (1 + \gamma_{\text{ext}})L_v$. The parameter $\gamma_{\text{full}}$ controls the full scope of the time horizon while $\gamma_{\text{ext}}$ controls the portion of idle time relative to the travel time. For the instances generated for computational performance comparison discussed in Section 6.2, we choose $\gamma_{\text{full}} = 50$ and $\gamma_{\text{ext}} = 2$.

The above three parts (road network topology, vehicle set, and time windows) cover most details of problem-instance generation. We also need to choose a number for the maximum platoon size $\lambda$ to finalize the instance generation. If $\lambda = \infty$ is chosen, it means no capacity constraint is imposed on the platoon size.

6.2. Computational performance comparison

By applying the problem-instance generation techniques described in Section 6.1, we generate 30 different instances to test the computation performance of the proposed formulations in this paper against the continuous-time based formulation (Appendix). The 30 instances can be divided into two major groups, each containing 15 instances. For all instances in the first group, there is no constraint on the platoon size ($\lambda = \infty$), while for those in the second group, we choose $\lambda = 10$ as the capacity. The 15 instances within each group are generated corresponding to 15 combinations of the network topology and the vehicle number $N$, where there are 3 different networks: ArtifNet (Art), Chicago-cluster (CC) and Chicago-full (CF), and 5 options for $N$: 100, 200, 300, 400 and 500. For each combination, only one instance is generated by selecting origin and destination nodes, and generating $T^O_v$ and $T^D_v$ following the rules described in Section 6.1. Therefore, the 15 instances can be uniquely labeled by the network type and $N$. For example, Art200 denotes the instance of a 200-vehicle system on the ArtifNet, and the labels of all instances are provided in the first two columns of Table 6. The experiments of computational performance are conducted on the case that the route graph (the graph induced by the routes of all vehicles) is a maximal shared tree (Definition 5). However, the route graph formed by shortest paths of all vehicles is not necessarily a MST. Some minor adjustments of routes have been made to cast the route graph to be a MST.
Examples of recasted route graphs (maximal ideal trees) are given in Figure 12 (b), (d) and (f) for a 100-vehicle system on the three networks, respectively.

We compare the computational performance of the time-overlap (TWOF), assignment (VA) and continuous-time (CT) formulations when they are applied to model and solve the 30 tree-based FRCVP numerical instances. The time-overlap formulation is the first one to be ruled out based on preprocess of Gurobi. It involves too many variables and constraints, and it takes a very long time for the solver to load the model using this formulation. Therefore, it remains to compare the assignment formulation against the continuous-time formulation. To build the assignment formulation, an origin node of an arbitrary vehicle is chosen to be the root of the MST, the RTW of each vehicle is constructed following Definition 6, and then all the RTW’s are discretized using Algorithm 1. The computation is performed on a single core 2.3 GHz CPU with 32 GB memory. For each numerical instance, the solution time limit is one hour. The comparison of the two formulations are based on the solution time (if the instance is solved within the time limit), the best objective value and the relative optimality gap at 10 mins and 1 hour, respectively. The results of comparison are given in Table 6. The objective values reported in the table are rescaled by the 100-vehicle system for each road network. (See the caption of Table 6 for more details of rescaling.) The general trend is that solving the instances with no constraint on the platoon size is easier than those with capacity constraints for both formulations, which is as expected. In particular, using the assignment formulation, 6 (out of 15) instances (Art100, Art200, CC100, CC200, CF100 and CF200) are solved to optimality for the non-capacitated case compared to the 4 instances (Art100, CC100, CC200 and CF100) for the capacitated case. While using the continuous-time formulation, 2 instances (CC100, CF100) are solved to optimality for both the non-capacitated and capacitated cases, but less computational time is reported for the first case.

The results show that the assignment formulation outperforms the continuous-time formulation in almost every numerical instance. More instances can be solved to optimality by using the assignment formulation compared to using the continuous-time formulation, i.e., 6 versus 2 for the non-capacitated case and 4 versus 2 for the capacitated case. Furthermore, for the instances that can be solved to optimality using both formulations, a significant reduction in computational time is reported for the assignment formulation (at least 6 times faster than the continuous-time formulation), except for the instance (CC100, \( \lambda = 10 \)). For the instances that cannot be solved to optimality, the assignment formulation outperforms the continuous-time formulation in terms of the objective value and the optimality gap with a few exceptions on instances with capacity constraints. At the 10 mins checkpoint, the two formulations are tie in 6 (out of 30) instances. The assignment formulation outperforms in 17 instances, while it underperforms in 7 instances which are all in the capacitated
case (Art200, Art300, CF500, CC500, CF300, CF400 and CF500 with $\lambda = 10$). At the one-hour time limit, the assignment formulation underperforms in only 1 instance (CF500, $\lambda = 10$).

In the Gurobi solver, one needs to specify the method for solving the root relaxation linear program, and the default is to use the dual simplex method. In most cases, one can rely on the default method. But it turns out that in our case, the method selection plays a crucial role to make the two formulations perform at their best, especially for instances involving a large number of vehicles (i.e., $N \geq 300$), and the reason is interestingly related to the effectiveness of the formulation. Our investigation shows that the dual simplex method is sufficiently effective for solving the root LP from the continuous-time formulation, while the barrier method (interior point method) is most effective for solving the that from the assignment formulation. It is therefore worth to note that the results reported in Table 6 are based on using the most effective LP-solving method for the two formulations, i.e., the barrier method for the assignment formulation and the dual simplex method for the continuous-time formulation, respectively. To get some insight, we have conducted an independent investigation of the three options of LP-solving methods: the primal simplex, dual simplex and barrier methods on solving the root-relaxation linear programs of the two formulations. The results are presented in Table 7. For the continuous-time formulation, the primal and dual simplex methods are roughly on-par for solving the root LP’s, whereas the barrier method takes longer time compared to the former two methods, especially on CF300, CF400 and CF500. However, the situation is very different for the assignment formulation, in which the barrier method remarkably outperforms the other two methods in all instances investigated, and there are 4~5 instances that can not be solved to optimality within the one-hour time limit by the primal and dual simplex methods. The linear program given by the assignment formulation seems to be much harder to solve compared to the one given by the continuous-time formulation. But once the assignment LP is solved, it yields a much more effective LP solution compared to the continuous-time LP. This is observed from that the objective ratio (See the caption of Table 7 for the definition) given by the assignment LP is 10%~20% smaller than the one given by the continuous-time LP. This ‘no free lunch’ observation is in fact a clear evidence that the assignment formulation is ‘tight and compact’, and is much stronger than the continuous-time formulation. Indeed, in the extreme case, the most effective linear program corresponds to the convex hull of all feasible solutions (mixed-integer points). But the polytope defined by that convex hull can have exponentially many extremal points (vertices), in which case iterating from one vertex to another in its neighborhood (i.e., the approach used by the primal and dual simplex methods) can be extremely inefficient for large instances. However, the high complexity of vertex enumeration on the surface can be successfully avoided by the barrier method since it always iterates within the interior of the polytope. Following this logic, if a formulation induces a polytope that is very close to the convex hull of feasible points,
it is very likely to have a similar computational behavior described above, which is exactly what has been observed for the assignment formulation, showing that the assignment formulation is indeed very strong.

6.3. Performance of approximation algorithms

Additional numerical experiments have been conducted to investigate the performance of the LP-relaxation based randomized algorithm (presented in Section 4.3) and the greedy algorithm (Algorithm 3) for solving tree-based FRCVP instances heuristically. Note that Algorithm 3 is a practical version of Algorithm 2, as the later one is an approximation algorithm designed for achieving a better competitive ratio in theory. The two algorithms are compared based on 15 numerical instances with \( \lambda = \infty \). For each numerical instance, the LP-relaxation based randomized algorithm has been repeated 50 times to choose the best objective value corresponding to this algorithm. The objective values are rescaled with respect to the best objective obtained by the assignment formulation. That means, for each numerical instance, the objective value given by the assignment formulation is set to be one. The results are summarized in Table 8. The table shows that the greedy algorithm can find a significantly better objective value in every instance considered, compared to the randomized algorithm. Specifically, the objective value given by the randomized algorithm is in the range \([0.622 \sim 0.878]\), while the objective range is \([0.899 \sim 1.000]\) by the greedy algorithm, showing that the greedy algorithm is clearly better in practice.

6.4. Fuel saving and time-window size correlation

Recall that the origin and destination time \( T^O_v \) and \( T^D_v \) for a vehicle \( v \) is generated such that \( T^D_v - T^O_v = (1 + \gamma_{\text{ext}})L_v \). The length of the ‘spare’ time for \( v \) is \( T^D_v - T^O_v - L_v = \gamma_{\text{ext}}L_v \), and it reflects the time flexibility of \( v \) for platooning. Given that \( T^O_v \) is randomly drawn from the interval \([0, \gamma_{\text{full}}L]\), the ratio between the average length of spare time and the length of time horizon is given as

\[
\frac{\text{avg}_{v \in V} \gamma_{\text{ext}}L_v}{\gamma_{\text{full}}L} = \frac{\gamma_{\text{ext}}}{\gamma_{\text{full}}}
\]

We refer \( \gamma_{\text{ext}}/\gamma_{\text{full}} \) as the extension ratio in this section, and it measures the overall flexibility for platooning. We investigate the total fuel saving of the vehicle system as a function of the extension ratio. In the experiments, we let the extension ratio grow from 0.01 to 0.15 with the step size 0.01. For a fixed extension ratio, we generate 20 numerical instances by randomly sampling \( T^O_v \) for every vehicle \( v \) and solve them to obtain the total fuel saving. The experiments are conducted on four network-N combinations: Art100, Art200, CC100 and CC200, respectively. The results are given in Figure 8. The figure shows that the total fuel saving monotonically increases as the extension ratio increases, but the magnitude of increment decreases. It is also observed that the uncertainty
led by the randomness of selecting $T^O_v$ is relatively small compared to the objective value (i.e., less than 5%). For Art100 and CC100, the level of uncertainty decreases as the extension ratio increases, which shows that the increment of spare time can mitigate the uncertainty of origin time generation. Furthermore, the 200-vehicle system seems to have lower level of uncertainty compared to the 100-vehicle counterpart. Indeed, given that the time horizon $[0, \gamma_{\text{full}}]$ is fixed, when more vehicles are involved, the RTW’s of these vehicles are more likely to overlap for a fixed extension ratio.

We investigate the total number of distinct time buckets (the quantity $|S| = |\bigcup_{v \in V} S_v|$) generated by the adaptive time discretization (Algorithm 1) versus the extension ratio. The numerical results are given in Figure 9 for 20 random instances based on Art100 and CC100, for which the extension ratio increases from 0.01 to 0.1 with the step size 0.01. Notice that the number of time buckets satisfies the lower bound $N = 100$ and the upper bound $2N = 200$ as shown in Proposition 3. The standard deviation is larger at the smaller values of the extension ratio, which is intuitive as there is more uncertainty on whether two randomly positioned RTW’s can overlap when the spare time extension is low. We have also investigated the $|S|$ versus the parameter $\beta$ defined in Theorem 1, and the results are given in Figure 10 for $N = 100$ and 200. Notice that the key difference between the investigation shown in Figure 9 and that in Figure 10 is on the mechanism of RTW generation. For Figure 9, the size of a RTW is generated such that it is deterministically proportional to the travel time of a vehicle (measured by $\gamma_{\text{ext}}$), while for Figure 10 the size of a RTW follows a uniform distribution on $[0, h]$ (with $h$ defined in Theorem 1). For both investigations, the position of a RTW follows a uniform distribution on a time horizon. As shown in Figure 10, the simulation result of $E[|S|]$ is compared with its theoretical lower bound derived in Theorem 1(b) at each $\beta$ value. It seems that the lower bound is a good approximation of the simulation result at $\beta \leq 0.8$. As $\beta$ increases, the negative quadratic term increasingly dominates the lower bound, making the lower bound increasingly deviate from the simulation result. To get a better understanding of the vehicle-level count $|S_v|$ (the number of feasible time buckets of a vehicle $v$), we further investigate the distribution of vehicles over the number of feasible time buckets, and the results are given in Figure 11 for Art100 and CC100. In this case, the distribution is generated by counting the number of vehicles (within the 100-vehicle system) that have exactly $k$ feasible time buckets with $k$ from 1 to 30, and for the extension ratio being 0.01, 0.02, 0.03, 0.04 and 0.05, respectively. Intuitively, Figure 11 shows that the distribution increasingly spreads out as the extension ratio increases, meaning that more vehicles will have a larger amount of feasible time buckets as the RTW size increases.

7. Discussion and Further Research

The adaptive time discretization algorithm and the novel vehicle-to-time-bucket assignment formulation are proven to be more effective than the continuous-time formulation from the theoretical and
computational perspectives, especially for FRCVP instances with tree-based routes. This approach can be elegantly extended to general instances to produce high-quality heuristic methods. The assignment formulation is shown to be a promising base to derive approximation algorithms, which is of independent theoretical interests. The combination of this work and our previous one has concluded the framework of coordinated vehicle platooning over a road network with time constraints in the deterministic setting. Further research could consider incorporating travel time uncertainty and the impact of traffic under the framework of stochastic and robust optimization. Note that the adaptive time discretization and assignment formulation provide an ideal foundation of this mission, as because the impact of travel time uncertainty can be naturally modeled as shrinkage of the relative time windows, which diminishes the chance of platooning.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357. We are grateful for the data of greater Chicago highway network provided by Dr. Jeffrey Larson from the Argonne National Laboratory for generating numerical instances.

References

Abdolmaleki M, Shahabi M, Yin Y, Masoud N (2019) Itinerary planning for cooperative truck platooning. *SSRN Electronic Journal* URL http://dx.doi.org/10.2139/ssrn.3481598.

Alam AA, Gattami A, Johansson KH (2010) An experimental study on the fuel reduction potential of heavy duty vehicle platooning. *13th International IEEE Conference on Intelligent Transportation Systems*, 306–311, URL http://dx.doi.org/10.1109/itsc.2010.5625054.

Andersen J, Christiansen M, Crainic TG, Grønhaug R (2011) Branch and price for service network design with asset management constraints. *Transportation Science* 45(1):33–49, URL http://dx.doi.org/10.1287/trsc.1100.0333.

B McAuliffe XL M Lammert, Shladower S, Surcel M, Kailas A (2018) Influences on energy savings of heavy trucks using cooperative adaptive cruise control. *WCX World Congress Experience* (SAE International), ISSN 0148-7191, URL http://dx.doi.org/https://doi.org/10.4271/2018-01-1181.

Baskar LD, De Schutter B, Hellendoorn H (2009a) Optimal routing for intelligent vehicle highway systems using mixed integer linear programming. *Proceedings of the 12th IFAC Symposium on Transportation Systems*, 569–575.

Baskar LD, De Schutter B, Hellendoorn J (2009b) Optimal routing for intelligent vehicle highway systems using a macroscopic traffic flow model. Barth M, ed., *Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems*, 1–6.
Table 6  Comparison of computational performance between the assignment formulation and the continuous-time formulation based on 30 numerical instances of FRCVP with the route graph being a maximal shared tree. The time limit for solving each instance is one hour and the tolerance threshold for optimality gap is set to be 0.1%. The solution time is reported if the instance can be solved within the one-hour time limit, otherwise it is labeled as ‘-’. For the two formulations, the objective value and relative optimality gap (in percent) are reported after computing for 10 mins, and one hour, respectively. Note that the objective value has been rescaled to increase the interpretability. Specifically, for each road network, the optimal objective value of the 100-vehicle system is set as the reference value for representing the objective values of other instances defined on the same network and with the same setting of the platoon-size constraint (by taking the ratio with respect to the reference value). For example, the optimal objective of Art100-inf is set to be 1.0 as the reference for representing objective values of Art200-inf to Art500-inf.

| Instance | Capacity | Assignment Formulation | Continuous-time Formulation |
|----------|----------|------------------------|-----------------------------|
|          |          | 10 min | 1 hour | 10 min | 1 hour |
|          | sol. time (s) | obj. gap (%) | obj. gap (%) | sol. time (s) | obj. gap (%) | obj. gap (%) |
| Art100   | inf      | 14     | 1.000 | 0.00   | 1.000 | 1.000 | 0.23 |
| Art200   | inf      | 2454   | 2.169 | 0.25   | 2.170 | 0.00  | 3.21 |
| Art300   | inf      | -      | 3.496 | 1.77   | 3.523 | 0.70  | 5.09 |
| Art400   | inf      | -      | 4.631 | 4.29   | 4.778 | 0.97  | 7.26 |
| Art500   | inf      | -      | 5.891 | 5.10   | 5.972 | 3.66  | 9.79 |
| CC100    | inf      | 4      | 1.000 | 0.00   | 1.000 | 1.000 | 0.00 |
| CC200    | inf      | 155    | 2.371 | 0.00   | 2.371 | 0.00  | 0.70 |
| CC300    | inf      | -      | 3.818 | 0.23   | 3.819 | 0.05  | 1.10 |
| CC400    | inf      | -      | 5.253 | 0.29   | 5.253 | 0.18  | 2.05 |
| CC500    | inf      | -      | 6.683 | 1.02   | 6.719 | 0.31  | 2.17 |
| CF100    | inf      | 10     | 1.000 | 0.00   | 1.000 | 1.000 | 0.00 |
| CF200    | inf      | 1291   | 2.328 | 0.12   | 2.329 | 0.01  | 1.77 |
| CF300    | inf      | -      | 3.680 | 1.71   | 3.721 | 0.44  | 4.86 |
| CF400    | inf      | -      | 5.129 | 0.50   | 5.132 | 0.20  | 5.99 |
| CF500    | inf      | -      | 6.320 | 2.64   | 6.400 | 0.47  | 6.24 |
| Art100   | 10       | 407    | 1.000 | 0.10   | 1.000 | 1.000 | 0.23 |
| Art200   | 10       | -      | 1.979 | 11.75  | 2.143 | 1.20  | 4.01 |
| Art300   | 10       | -      | 3.144 | 13.60  | 3.438 | 3.32  | 8.32 |
| Art400   | 10       | -      | 4.063 | 18.02  | 4.534 | 7.16  | 9.62 |
| Art500   | 10       | -      | 4.948 | 22.00  | 5.733 | 8.41  | 13.89 |
| CC100    | 10       | 118    | 1.000 | 0.00   | 1.000 | 1.000 | 0.00 |
| CC200    | 10       | 1228   | 2.357 | 1.62   | 2.370 | 0.00  | 0.74 |
| CC300    | 10       | -      | 3.722 | 4.81   | 3.792 | 1.15  | 1.79 |
| CC400    | 10       | -      | 4.948 | 22.00  | 5.733 | 8.41  | 13.89 |
| CC500    | 10       | -      | 6.293 | 8.51   | 6.568 | 3.41  | 4.94 |
| CF100    | 10       | 142    | 1.000 | 0.00   | 1.000 | 1.000 | 0.01 |
| CF200    | 10       | -      | 2.246 | 5.03   | 2.315 | 0.41  | 1.60 |
| CF300    | 10       | -      | 3.276 | 14.35  | 3.619 | 3.58  | 6.96 |
| CF400    | 10       | -      | 4.394 | 16.31  | 4.856 | 6.11  | 7.02 |
| CF500    | 10       | -      | 5.695 | 13.58  | 5.948 | 8.78  | 8.15 |

Baskar LD, Schutter BD, Hellendoorn H (2013) Optimal routing for automated highway systems. *Transportation Research Part C: Emerging Technologies* 30:1–22, URL http://dx.doi.org/10.1016/j.trc.2013.01.006.

Boland N, Hewitt M, Marshall L, Savelsbergh M (2017a) The continuous-time service network design problem. *Operations Research* 65(5):1303–1321, URL http://dx.doi.org/10.1287/opre.2017.1624.
Table 7  The solution time comparison of solving the root relaxation linear programs (LP’s) using three different methods: the primal simplex, the dual simplex and the barrier method (interior point method) for the assignment and continuous-time formulations, respectively, tested on numerical instances involving large number of vehicles without limit on the platoon size. The time limit for solving the LP relaxation is one hour. If an instance cannot be solved to optimality by a method, the corresponding cell of the computational time is marked as ‘-‘. The column ‘obj. ratio’ is the ratio between the optimal objective of the root relaxation LP and the optimal objective of the instance. As a maximization problem, the ratio is always greater than or equal to 1.0.

| Instance | Assignment Formulation | Continuous Time Formulation |
|----------|------------------------|----------------------------|
|          | primal (sec.) | dual (sec.) | barrier (sec.) | obj. ratio | primal (sec.) | dual (sec.) | barrier (sec.) | obj. ratio |
| Art300   | 1617         | 707         | 40 | 1.040 | 10 | 8 | 22 | 1.166 |
| Art400   | -            | -           | 91 | 1.035 | 54 | 25 | 47 | 1.143 |
| Art500   | -            | -           | 200| 1.055| 47 | 64 | 86 | 1.150 |
| CC300    | 239          | 66          | 10 | 1.024 | 3  | 4  | 11 | 1.144 |
| CC400    | 963          | 302         | 22 | 1.016 | 7  | 15 | 28 | 1.111 |
| CC500    | -            | 1147        | 43 | 1.012 | 11 | 11 | 40 | 1.089 |
| CF300    | 2913         | 3392        | 96 | 1.060 | 20 | 30 | 67 | 1.200 |
| CF400    | -            | -           | 350| 1.044 | 29 | 41 | 115| 1.157 |
| CF500    | -            | -           | 499| 1.049 | 75 | 83 | 215| 1.155 |

Table 8  The performance of two approximation algorithms for solving 15 FRCVP numerical instances with \( \lambda = \infty \) heuristically. For each numerical instance, the objective value given by the assignment formulation is set to be one.

| Instance | LP relaxation | Greedy algorithm |
|----------|---------------|-----------------|
|          | obj. gap (%)   | obj. gap (%)    |
| Art100   | 0.764         | 23.61           | 0.952 | 4.83 |
| Art200   | 0.651         | 34.90           | 0.952 | 4.77 |
| Art300   | 0.622         | 37.79           | 0.969 | 3.07 |
| Art400   | 0.645         | 35.47           | 0.967 | 3.30 |
| Art500   | 0.632         | 36.82           | 1.000 | 0.01 |
| CC100    | 0.878         | 12.20           | 0.902 | 9.81 |
| CC200    | 0.801         | 19.91           | 0.941 | 5.87 |
| CC300    | 0.782         | 21.79           | 0.930 | 7.02 |
| CC400    | 0.772         | 22.77           | 0.942 | 5.78 |
| CC500    | 0.752         | 24.78           | 0.962 | 3.80 |
| CF100    | 0.851         | 14.89           | 0.899 | 10.12 |
| CF200    | 0.767         | 23.30           | 0.955 | 4.46 |
| CF300    | 0.724         | 27.59           | 0.972 | 2.77 |
| CF400    | 0.752         | 24.80           | 0.964 | 3.64 |
| CF500    | 0.734         | 26.57           | 0.972 | 2.75 |

Boland N, Hewitt M, Vu DM, Savelsbergh M (2017b) Solving the traveling salesman problem with time windows through dynamically generated time-expanded networks. Salvagnin D, Lombardi M, eds., Integration of AI and OR Techniques in Constraint Programming, 254–262 (Cham: Springer International Publishing), ISBN 978-3-319-59776-8.

Bonnet C, Fritz H (2000) Fuel consumption reduction in a platoon: Experimental results with two electronically coupled trucks at close spacing. SAE Technical Paper Series, 1–7 (SAE International), URL http://dx.doi.org/10.4271/2000-01-3056.
Figure 8 The correlation between total fuel saving and the extension ratio ($\gamma_{\text{ext}}/\gamma_{\text{full}}$). The extension ratio has been sequentially increased from 0.01 to 0.15 with the step size 0.01. For a fixed extension ratio, 20 independent numerical instances are generated by randomly choosing $T_{\text{v}}$ from the interval $[0, \gamma_{\text{full}}L]$ for every vehicle. The sample mean and standard deviation are computed for the objective obtained from the 20 samples. The blue curve represents the mean value with respect to the extension ratio, and the shaded region is corresponding to the $\pm 2\sigma$ confidence interval, where $\sigma$ is the standard deviation. The investigation is conducted on Art100, Art200, CC100 and CC200, respectively.

Browand F, McArthur J, Radovich C (2004) Fuel saving achieved in the field test of two tandem trucks. Technical report, California PATH Research Report UCB-ITS-PRR-2004-20.

Browned F, McArthur J, Radovich C (2004) Fuel saving achieved in the field test of two tandem trucks. California PATH Research Report UCB-ITS-PRR-2004-20, University of Southern California.

Contardo C, Desaulniers G, Lessard F (2015) Reaching the elementary lower bound in the vehicle routing problem with time windows. Networks 65(1):88–99, URL http://dx.doi.org/10.1002/net.21594.
The correlation between the total number of time buckets and the extension ratio. The investigation is conducted on Art100 and CC100, and the extension ratio increases from 0.003 to 0.06 with step size 0.003. For each extension ratio 100 samples are generated to compute the sample mean and standard deviation. The extension ratio is represented by an index ranging from 1 to 20 on the horizontal line in the above two figures.

The sample mean (solid curve) of total number of time buckets as a function of $\beta$ (from 0.1 to 1.5 with the step size 0.1) defined in Theorem 1. The theoretical lower bound (dashed line) as a function of $\beta$ is plotted for comparison. The sample size is 20 for each $\beta$ value.

Crainic TG (2000) Service network design in freight transportation. *European Journal of Operational Research* 122(2):272–288, URL http://dx.doi.org/10.1016/S0377-2217(99)00233-7.

Crainic TG, Hewitt M (2021) *Service network design* (Springer, Cham).

Crainic TG, Hewitt M, Toulouse M, Vu DM (2016) Service network design with resource constraints. *Transportation Science* 50(4):1380–1393, URL http://dx.doi.org/10.1287/trsc.2014.0525.
Figure 11  The distribution of vehicles according to the cardinality $|S_u|$ defined in Table 3 (the number of time buckets that are feasible to a vehicle $u$ after running Algorithm 1). Figure (a) and (b) are corresponding to Art100 and CC100, respectively, and the distribution of vehicle number is plotted for the extension ratio $\gamma_{ext}/\gamma_{full} = 0.01, 0.02, 0.03, 0.04$ and 0.05 within the same figure. For each value of $\beta$, 20 random instances are generated, and Algorithm 1 has been applied to all the instances. The solid line represents the distribution of mean value and the shaded region shows the $\pm 2\hat{\sigma}$ confidence interval, where $\hat{\sigma}$ is the sample standard deviation.

DasGupta A (2008) Lecture notes: Finite sample theory of order statistics and extremes. URL: https://www.stat.purdue.edu/~dasgupta/orderstats.pdf.

Dávila A (2013) SARTRE report on fuel consumption. Technical report, Technical Report for European Commission under the Framework 7 Programme Project 233683 Deliverable 4.3.

Dobrovolny CS, Untaroiu C, Sharma R, Jin H, Meng Y (2019) Implications of truck platoons for roadside hardware and vehicle safety. Technical report, Texas A&M Transportation Institute, Virginia Tech.

Durham University (2017) Lecture notes: order statistics. Lecture notes, URL: https://www.maths.dur.ac.uk/stats/courses/Prob34/1617Probability34H.pdf.

Erera A, Hewitt M, Savelbergh M, Zhang Y (2013) Improved load plan design through integer programming based local search. Transportation Science 47(3):412–427, URL http://dx.doi.org/10.1287/trsc.1120.0441.

Expert Market Research (2020) Truck platooning market - growth, trends, covid-19 impact, and forecasts (2021 - 2026). https://www.expertmarketresearch.com/reports/truck-platooning-market, accessed: 2022-03-10.

Fernandes P, Nunes U (2012) Platooning with IVC-enabled autonomous vehicles: Strategies to mitigate communication delays, improve safety and traffic flow. IEEE Transactions on Intelligent Transportation Systems 13(1):91–106, URL http://dx.doi.org/10.1109/tits.2011.2179936.
Figure 12  Part (a), (c) and (e) are networks used for generating numerical instances. Part (b), (d) and (e) are examples of a 100-vehicle system defined on the three networks, respectively, with circle and square dots representing the origin and destination nodes, respectively. The original route of each vehicle is the shortest path, but minor adjustment on routes has been made to ensure that the route-graph is a tree. Straight lines are used to link the origin and destination nodes in (d) and (f) to simplify the plot.
Fischer F, Helmberg C (2012) Dynamic graph generation for the shortest path problem in time expanded networks. *Mathematical Programming* 143(1-2):257–297, URL http://dx.doi.org/10.1007/s10107-012-0610-3.

Ford LR, Fulkerson DR (1958) Constructing maximal dynamic flows from static flows. *Operations Research* 6(3):419–433, URL http://dx.doi.org/10.1287/opre.6.3.419.

Ford LR, Fulkerson DR (1962) *Flows in networks* (Princeton University Press), URL http://dx.doi.org/10.1007/978-3-319-11008-0.

Gong S, Shen J, Du L (2016) Constrained optimization and distributed computation based car following control of a connected and autonomous vehicle platoon. *Transportation Research Part B: Methodological* 94:314–334, URL http://dx.doi.org/10.1016/j.trb.2016.09.016.

Ho K, de Weck OL, Hoffman JA, Shishko R (2014) Dynamic modeling and optimization for space logistics using time-expanded networks. *Acta Astronautica* 105(2):428–443, URL http://dx.doi.org/10.1016/j.actaastro.2014.10.026.

Hobert LHX (2012) *A study on platoon formations and reliable communication in vehicle platoons*. Master’s thesis, University of Twente.

Hoef SVD (2018) *Coordination of Heavy-Duty Vehicle Platooning*. Ph.D. thesis, KTH Royal Institute of Technology.

Janssen R, Zwijnenberg H, Blankers I, de Kruijff J (2015) Truck platooning driving the future of transportation. Technical report, TNO Mobility and Logistics.

Jarrah A, Johnson E, Neubert L (2009) Large-scale, less-than-truckload service network design. *Operations Research* 57(2):609–625, URL http://dx.doi.org/10.1287/opre.1080.0587.

Kammer C (2013) *Coordinated heavy truck platoon routing using global and locally distributed approaches*. Master’s thesis, KTH Royal Institute of Technology.

Köhler E, Langkau K, Skutella M (2002) Time-expanded graphs for flow-dependent transit times. Möhring R, Raman R, eds., *Algorithms — ESA 2002*, 599–611 (Berlin, Heidelberg: Springer Berlin Heidelberg), ISBN 978-3-540-45749-7.

Lammert M, Duran A, Diez J, Burton K, Nicholson A (2014) Effect of platooning on fuel consumption of class 8 vehicles over a range of speeds, following distances, and mass. *SAE Int. J. Commer. Veh* 7(2), URL http://dx.doi.org/10.4271/2014-01-2438.

Larson J, Kammer C, Liang KY, Johansson KH (2013) Coordinated route optimization for heavy-duty vehicle platoons. *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, 1196–1202, URL http://dx.doi.org/10.1109/itsc.2013.6728395.

Larson J, Liang KY, Johansson KH (2015) A distributed framework for coordinated heavy-duty vehicle platooning. *IEEE Transactions on Intelligent Transportation Systems* 16(1):419–429, URL http://dx.doi.org/10.1109/tits.2014.2320133.
Larson J, Munson T, Sokolov V (2016) Coordinated platoon routing in a metropolitan network. 2016 Proceedings of the Seventh SIAM Workshop on Combinatorial Scientific Computing, 73–82, URL http://dx.doi.org/10.1137/1.9781611974690.ch8.

Larsson E, Senfton G, Larson J (2015) The vehicle platooning problem: Computational complexity and heuristics. Transportation Research Part C: Emerging Technologies 60:258–277, URL http://dx.doi.org/10.1016/j.trc.2015.08.019.

Li X, Li Q, Claramunt C (2018) A time-extended network model for staged evacuation planning. Safety Science 108(0):225–236, URL http://dx.doi.org/10.1016/j.ssci.2017.08.004.

Liang KY (2014) Coordination and routing for fuel-efficient heavy-duty vehicle platoon formation. Licentiate Thesis, KTH.

Liang KY, Mårtensson J, Johansson KH (2013) When is it fuel efficient for a heavy duty vehicle to catch up with a platoon? IFAC Proceedings Volumes, volume 46, 738–743 (Elsevier BV), URL http://dx.doi.org/10.3182/20130904-4-jp-2042.00071.

Liang KY, Mårtensson J, Johansson KH (2014) Fuel-saving potentials of platooning evaluated through sparse heavy-duty vehicle position data. IEEE conference proceedings, 1061–1068, URL http://dx.doi.org/10.1109/IVS.2014.6856540.

Lu XY, Shladover SE (2011) Automated truck platoon control. Technical Report UCB-ITS-PRR-2011-13, University of California, Berkeley, california PATH Research Report.

Luo F, Larson J (2020) A repeated route-then-schedule approach to coordinated vehicle platooning: algorithms, valid inequalities and computation, URL https://arxiv.org/pdf/2004.13758.pdf, forthcoming in Operations Research.

Luo F, Larson J, Munson T (2018) Coordinated platooning with multiple speeds. Transportation Research Part C: Emerging Technologies 90:213–225, URL http://dx.doi.org/10.1016/j.trc.2018.02.011.

Muratori M, Holden J, Lammert M, Duran A, Young S, Gonder J (2017) Potentials for platooning in U.S. highway freight transport. SAE International Journal of Commercial Vehicles 10(1):1–5, URL http://dx.doi.org/10.1016/j.procs.2012.06.111.

Nowakowski C, O’Connell J, Shladover SE, Cody D (2010) Cooperative adaptive cruise control: Driver acceptance of following gap settings less than one second. Proceedings of the Human Factors and Ergonomics Society Annual Meeting 54(24):2033–2037, URL http://dx.doi.org/10.1177/154193121005402403.

Nowakowski C, Shladover SE, Cody D (2011) Cooperative adaptive cruise control: Testing drivers’ choice of following distances. California PATH Research Report UCB-ITS-PRR-2011-01, University of California, Berkeley.

Papadimitriou CH, Yannakakis M (1979) Scheduling interval-ordered tasks. SIAM Journal on Computing 8(3):405–409, URL http://dx.doi.org/10.1137/0208031.
Pecin D, Contardo C, Desaulniers G, Uchoa E (2017) New enhancements for the exact solution of the vehicle routing problem with time windows. *INFORMS Journal on Computing* 29(3):489–502, URL http://dx.doi.org/10.1287/ijoc.2016.0744.

Peloton Technology (2020) Platooning hardware. http://peloton-tech.com/platoon-pro/hardware/, accessed: 2022-03-08.

Robinson T, Chan E, Coelingh E (2010) Operating platoons on public motorways: An introduction to the SARTRE platooning programme. *Proceedings of the 17th ITS World Congress*.

Schittler M (2003) State-of-the-art and emerging truck engine technologies for optimized performance, emissions and life cycle costs. Tech. report, DaimlerChrysler AG.

Schroten A, Warringa G, Bles M (2012) Marginal abatement cost curves for heavy duty vehicles. Background report, CE Delft.

Shida M, Nemoto Y (2009) Development of a small-distance vehicle platooning system. *16th ITS World Congress and Exhibition on Intelligent Transport Systems and Services*, 1–8.

Shladover SE, Su D, Lu XY (2012) Impacts of cooperative adaptive cruise control on freeway traffic flow. *Transportation Research Record: Journal of the Transportation Research Board* 2324(1):63–70, URL http://dx.doi.org/10.3141/2324-08.

Tsugawa S (2013) An overview on an automated truck platoon within the Energy ITS project. *IFAC Proceedings Volumes* 46(21):41–46, URL http://dx.doi.org/10.3182/20130904-4-jp-2042.00110.

Tsugawa S, Jeschke S, Shladover SE (2016) A review of truck platooning projects for energy savings. *IEEE Transactions on Intelligent Vehicles* 1(1):68–77, URL http://dx.doi.org/10.1109/tiv.2016.2577499.

Ullman JD (1973) Polynomial complete scheduling problems. *Operating Systems Review* 7(4):96–101, URL http://dx.doi.org/10.1145/800009.808055.

Valdés F, Iglesias R, Espinosa F, Rodríguez MA (2011) An efficient algorithm for optimal routing applied to convoy merging manoeuvres in urban environments. *Applied Intelligence* 37(2):267–279, URL http://dx.doi.org/10.1007/s10489-011-0326-8.

van de Hoef S, Johansson KH, Dimarogonas DV (2015) Coordinating truck platooning by clustering pairwise fuel-optimal plans. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 408–415, URL http://dx.doi.org/10.1109/itsc.2015.75.

Van De Hoef S, Johansson KH, Dimarogonas DV (2015) Fuel-optimal centralized coordination of truck platooning based on shortest paths. *American Control Conference, 2015*, 3740–3745 (IEEE).

van Doremalen KP (2015) *Platoon coordination and routing*. Master’s Internship Report, Eindhoven University of Technology.
Appendix. A continuous-time formulation of the FRCVP problem

The FRCVP problem has a continuous-time formulation developed in (Luo and Larson 2020). The formulation is given in (CT) and the definition of notations used in the formulation is provided in Table 9. Note that we assume the order of vehicles in a platoon does not affect its fuel saving. In the continuous-time formulation, all vehicles are labeled with positive integers and we only allow a vehicle with a greater index to follow a vehicle with a smaller index as a convention. (See the definition of the decision variable $f_{u,v,i,j}$ in Table 9.)

\[
\begin{align*}
\text{maximize} & \quad \sum_{(i,j)\in \bigcup_{v} R_{v}} \left( \sum_{v \in V_{i,j}} \sigma^l C_{i,j} f_{v,i,j} + \sum_{u,v \in V_{i,j}, u > v} \sigma^l C_{i,j} f_{u,v,i,j} \right) \\
\text{subject to:} & \quad t_{v,O_{v}} \geq T^{O}_{v}, \quad \forall v \in V, \quad (CT.1) \\
& \quad t_{v,D_{v}} \leq T^{D}_{v}, \quad \forall v \in V, \quad (CT.2) \\
& \quad t_{v,j} = t_{v,i} + T_{i,j} \quad \forall v \in V, \forall (i,j) \in R_{v}, \quad (CT.3) \\
& \quad t_{u,i} - t_{v,i} \leq M(1 - f_{u,v,i,j}) \quad \forall (i,j) \in \bigcup_{v} R_{v}, \forall u > v \in V_{i,j}, \quad (CT.4) \\
& \quad t_{u,i} - t_{v,i} \geq -M(1 - f_{u,v,i,j}) \quad \forall (i,j) \in \bigcup_{v} R_{v}, \forall u > v \in V_{i,j}, \quad (CT.5) \\
& \quad \sum_{w \in V_{i,j}, w < v} f_{v,w,i,j} \leq 1 - \ell_{v,i,j} \quad \forall v \in V, \forall (i,j) \in R_{v}, \quad (CT.6) \\
& \quad \sum_{u \in V_{i,j}, u > v} f_{u,v,i,j} \leq (\lambda - 1)\ell_{v,i,j} \quad \forall v \in V, \forall (i,j) \in R_{v}, \quad (CT.7)
\end{align*}
\]
∑_{u \in V; u > v} f_{u,v,i,j} \geq \ell_{v,i,j} \quad \forall v \in V, (i, j) \in R_v, \quad (CT.9)

f_{u,v,i,j} \in \{0, 1\} \quad \forall u, v \in V, u > v, \forall (i, j) \in R_v, \quad (CT.10)

\ell_{v,i,j} \in \{0, 1\} \quad \forall v \in V, \forall (i, j) \in R_v, \quad (CT.11)

where the M in constraints (CT.5) and (CT.6) is a big-M coefficient satisfying \(-M \leq t_{u,i} - t_{v,i} \leq M\) for all \((i, j) \in \bigcup_v R_v\), \(u, v \in V_{i,j}\) with \(u > v\) and any feasible values of \(t_{u,i}\) and \(t_{v,i}\).

The objective (CT.1) consists of two terms corresponding to fuel saving of lead and trailing vehicles on each edge, respectively. Note that maximizing the total fuel saving is equivalent to minimizing the total fuel cost. Constraints (CT.2)–(CT.4) ensure that vehicle travel times are consistent. Constraints (CT.5)–(CT.6) ensure that if vehicle \(u\) follows vehicle \(v\) in a platoon on edge \((i, j)\), they must traverse the edge simultaneously. Constraint (CT.7) ensures that if vehicle \(v\) is a lead vehicle on an edge, it cannot follow any other vehicles. If \(v\) is not a lead vehicle, it can follow no more than one vehicle (the lead vehicle) on an edge. (Note that our formulation does not allow any trailing vehicle to lead another vehicle.) Constraint (CT.8) enforces that a vehicle can lead at most \(\lambda - 1\) other vehicles. Constraint (CT.9) states that if \(v\) is a lead vehicle on an edge, the number of vehicles following \(v\) on that edge must be at least one.

| Table 9 | Sets, parameters, and variables for the continuous-time formulation. |
|---|---|
| **Set** | **Definition** |
| \(R_v\) | route of vehicle \(v \in V\) (an ordered subset of edges in the route) |
| \(N_v\) | set of nodes on \(R_v\), \(v \in V\) |
| \(V_{i,j}\) | set of vehicles taking the edge \((i, j) \in \bigcup_v R_v\) |
| **Parameter** | **Definition** |
| \(T^O_v, T^D_v, T_{i,j}\) | defined in Table 1 |
| \(\lambda\) | maximum number of vehicles permitted in a platoon |
| \(M_{u,v,i,j}\) | a vehicle-pair and edge dependent big-M coefficient |
| **Variable** | **Definition** |
| \(t_{v,O_v}\) | departure time of vehicle \(v\) from node \(O_v\) |
| \(t_{v,i}\) | arrival time of vehicle \(v\) at an intermediate node \(i\) along its route \(R_v\) |
| \(f_{u,v,i,j}\) | 1 if vehicle \(u\) follows vehicle \(v\) at edge \((i, j) \in R_u \cap R_v\), 0 otherwise. (We require \(u > v\) by the convention of ordering.) |
| \(\ell_{v,i,j}\) | 1 if vehicle \(v\) leads a platoon at edge \((i, j) \in R_v\), 0 otherwise |
Supplemental Material

EC.1. Proofs

**Proposition 1.** Let $P$ be a maximal ideal path in $G$ and $V'$ is a set of vehicles that share the path $P$. Let $s$ be the first node of $P$. By coordinating the starting time of each vehicle at their origin, the vehicles in $V'$ can form a single platoon when driving on $P$ if and only if for any two vehicles $u, v \in V'$ the intersection of their time windows at $s$ is non-empty.

Proof. **Necessity:** Suppose that by coordinating the starting time, all vehicles in $V'$ can form a platoon when driving along the path $P$. Then the arrival time at the node $s$ for vehicles in $V'$ must be the same. Let $t_0$ be the arrival time at the node $s$ of these vehicles. Since for any $v \in V'$, the arrival time of $v$ at the node $s$ must be within the time window $[t_{v,s}, \tilde{t}_{v,s}]$, it implies that $t_0 \in [t_{v,s}, \tilde{t}_{v,s}] \forall v \in V'$. Therefore, the intersection of any two vehicles' time windows at $s$ must contain $t_0$, and hence non-empty.

**Sufficiency:** Suppose $|V'| = n$. We label vehicles using indices $1, 2, \ldots, n$ in the proof. The time window for the vehicle $k$ at $s$ is denoted as $[t_{k,s}, \tilde{t}_{k,s}]$. It amounts to show the following claim: If any two vehicles' have non-empty time window intersection at $s$, then the intersection of all vehicles' time windows at $s$ is non-empty. If the above claim holds, then any element in $\cap_{k=1}^{n}[t_{k,s}, \tilde{t}_{k,s}]$ can be chosen as the simultaneous arrival time at node $s$ for all vehicles in $V'$, and the departure time of each vehicle can be determined accordingly. We prove the claim by induction on $n$. The claim obviously holds for the case $n = 2$. Suppose the claim holds for $n$, we now show that it holds for $n + 1$. By the induction hypothesis, the set $\cap_{k=1}^{n}[t_{k,s}, \tilde{t}_{k,s}]$ is non-empty. Since the intersection of finitely many closed interval is also a closed interval, the set $\cap_{k=1}^{n}[t_{k,s}, \tilde{t}_{k,s}]$ must be a closed interval and we denote this interval as $[\hat{t}_1, \hat{t}_2]$. If $[t_{n+1,s}, \tilde{t}_{n+1,s}] \cap [\hat{t}_1, \hat{t}_2]$ is non-empty, then the claim holds. Suppose $[t_{n+1,s}, \tilde{t}_{n+1,s}] \cap [\hat{t}_1, \hat{t}_2]$ is empty. Then it follows that either $\tilde{t}_{n+1,s} < \hat{t}_1$ or $t_{n+1,s} > \hat{t}_2$. Without loss of generality, we assume that $\tilde{t}_{n+1,s} < \hat{t}_1$. Since $[\hat{t}_1, \hat{t}_2] = \cap_{k=1}^{n}[t_{k,s}, \tilde{t}_{k,s}]$, there exists a vehicle $m$ $(1 \leq m \leq n)$ such that $t_{m,s} = \tilde{t}_1$. It implies that $\tilde{t}_{n+1,s} < t_{m,s}$, and hence $[t_{n+1,s}, \tilde{t}_{n+1,s}] \cap [t_{m,s}, \tilde{t}_{m,s}] = \emptyset$, which leads to a contradiction. □

**Proposition 2.** Let $T$ be a non-decomposable shared tree, and $|V(T)| \geq 2$. The following properties hold: (a) For any $u \in V(T)$ there exists a vehicle $v \in V(T) \setminus \{u\}$ (depending on $u$) such that $R_u \cap R_v$ is non-empty. (b) For any distinct $u, v \in V(T)$, the intersection $R_u \cap R_v$ is either an empty set or a directed path.

Proof. The proof is straightforward. For (a) if there exists a vehicle $u$ which does not share any edges with any other vehicle, then the shared tree $T$ is decomposable. For (b), obviously $R_u \cap R_v$
cannot contain any loops since $T$ is a tree. Suppose $R_u \cap R_v$ is non-empty and it is not a path. Then it must consist of multiple disjointed paths. Let us denote these paths as $s_1 \rightarrow s_2, s_3 \rightarrow s_4, \ldots, s_{2k-1} \rightarrow s_{2k}$, where $s_1, s_2, \ldots, s_{2k}$ are in the order of being visited by the two vehicles. Consider the two disjointed paths $s_1 \rightarrow s_2$ and $s_3 \rightarrow s_4$. We conclude that the path $R_u(s_2, s_3)$ is disjointed with $R_v(s_2, s_3)$, and these two paths form a loop in the undirected graph, which contradicts to the tree structure. □

**Lemma 1.** Let $T$ be a shared tree (not necessarily a MST), and $V(T)$ be the set of vehicles involved in $T$. (a) For any two vehicles $u, v \in V(T)$ with $R_u \cap R_v \neq \emptyset$, a departure-time coordination can make $u$ and $v$ platooned on $R_u \cap R_v$, if and only if the departure times $t_{u,O_u}, t_{v,O_v}$ determined by the coordination satisfy the following equation:

$$t_{u,O_u} + \tilde{T}_{O_u,r} = t_{v,O_v} + \tilde{T}_{O_v,r},$$

where $r$ is the root of $T$. (b) The vehicles in $V(T)$ are pseudo-platoonable on $T$ if and only if the intersection of their RTWs is non-empty.

**Proof.** (a) Let $s$ be the first visited node on the directed path $R_u \cap R_v$. A necessary and sufficient condition for $u$ and $v$ to form a platoon is that the arrival times of $u$ and $v$ at the node $s$ must be equal. Let $R_u(O_u, s)$ be the directed path from $O_u$ to $s$, and let $R_v(O_v, s)$ be the directed path from $O_v$ to $s$. We must have $R_u(O_u, s) \cap R_v(O_v, s) = \emptyset$. Let $P_0$ be the generalized path from $s$ to the root $r$. The simultaneous arrival condition for $u$ and $v$ at node $s$ is written as:

$$t_{u,O_u} + \sum_{(i,j) \in R_u(O_u,s)} T_{i,j} = t_{v,O_v} + \sum_{(i,j) \in R_v(O_v,s)} T_{i,j}.$$  

(EC.1)

By definition, the relative times of $O_u$ and $O_v$ with respective to $r$ are

$$\tilde{T}_{O_u,r} = \sum_{(i,j) \in R_u(O_u,s)} T_{i,j} + \sum_{(i,j) \in P_{u,r}} T_{i,j} - \sum_{(i,j) \in P_{r,u}} T_{i,j},$$

$$\tilde{T}_{O_v,r} = \sum_{(i,j) \in R_v(O_v,s)} T_{i,j} + \sum_{(i,j) \in P_{v,r}} T_{i,j} - \sum_{(i,j) \in P_{r,v}} T_{i,j},$$

(EC.2)

where $P_{s,r}$ (resp. $P_{r,s}$) is the subset of edges in $P_0$ that are pointing from $s$ to $r$ (resp. from $r$ to $s$).

Using (EC.2), we find that the equation (EC.1) is equivalent to $t_{u,O_u} + \tilde{T}_{O_u,r} = t_{v,O_v} + \tilde{T}_{O_v,r}$.

(b) Proof of the necessity: Suppose vehicles in $V(T)$ are pseudo-platoonable on $T$. Consider a departure-time coordination $\{t_{v,O_v} : v \in V(T)\}$ that ensures the pseudo-platoonability of vehicles in $V(T)$. Based on the result of Part (a), we define the relative departure time $t_v^{rel}$ for every vehicle $v \in V(T)$ as $t_v^{rel} = t_{v,O_v} + \tilde{T}_{O_v,r}$. We construct an auxiliary undirected graph $G^0$. The graph $G^0$ consists of $|V(T)|$ vertices, and each represents a vehicle. An edge is created between $u$ and $v$ if and only if $R_u \cap R_v$ is non-empty. Since $T$ is non-decomposable, the graph $G^0$ must be connected. According to
the result in Part (a), if \( G^0 \) has an edge between \( u \) and \( v \), then \( t^\text{rel}_u = t^\text{rel}_v \). It follows that all vehicles in \( \mathcal{V}(\mathcal{T}) \) must have a same relative departure time denoted as \( t^\text{rel} \), and hence \( t^\text{rel} \) is in the intersection of the relative time windows of all vehicles in \( \mathcal{V}(\mathcal{T}) \).

Proof of the sufficiency: Suppose the intersection of the relative time windows of all vehicles in \( \mathcal{V}(\mathcal{T}) \) is non-empty. We can pick any time \( t^\text{rel} \) in the intersection and set the departure time of vehicle \( v \) as \( t^\text{rel}_v = t^\text{rel} - \tilde{T}_{O_v,r} \) for all \( v \in \mathcal{V}(\mathcal{T}) \). Using the result of Part (a), it implies that any two vehicles from \( \mathcal{V}(\mathcal{T}) \) that have non-empty shared path can be platooned. By definition it means the vehicles in \( \mathcal{V}(\mathcal{T}) \) are pseudo-platoonable. \( \square \)

**Corollary 1.** Let \( \mathcal{T} \) be a MST, and \( \mathcal{V}^0 \) be a subset of \( \mathcal{V}(\mathcal{T}) \). The vehicles in \( \mathcal{V}^0 \) are pseudo-platoonable if and only if the subgraph of \( \tilde{G}(\mathcal{T}) \) induced by \( \mathcal{V}^0 \) is a clique.

Proof. Lemma 1(b) indicates that if \( \mathcal{V}^0 \) are pseudo-platoonable, then the subgraph of \( \tilde{G}(\mathcal{T}) \) induced by \( \mathcal{V}^0 \) is a clique. Conversely, we can use a similar argument as in Proposition 1 to show that if the RTW’s of any two vehicles in \( \mathcal{V}^0 \) have non-empty intersection (i.e., the subgraph of \( \tilde{G}(\mathcal{T}) \) induced by \( \mathcal{V}^0 \) is a clique), then intersection set \( \cap_{v \in \mathcal{V}^0} \text{RTW}_v \) is non-empty, where RTW\(_v\) is the RTW of \( v \), indicating that the vehicles in \( \mathcal{V}^0 \) are pseudo-platoonable by Lemma 1(b). \( \square \)

**Proposition 3.** Suppose the adaptive time discretization algorithm is applied to a set \( \mathcal{C} = \{[\tau^1_k, \tau^2_k] : v \in \mathcal{V}\} \) of RTWs corresponding to the set \( \mathcal{V} \) of vehicles that share a MST. Let \( \mathcal{S} \) be the collection of sorted time buckets returned by the algorithm. Then the following bounds hold:

\[
|\mathcal{C}| \leq |\mathcal{S}| \leq 2|\mathcal{C}| - 1.
\] (4)

Proof. Without loss of generality, we assume that the RTWs of all vehicles in \( \mathcal{V} \) are distinct, and hence \( |\mathcal{C}| = |\mathcal{V}| \), otherwise let \( \mathcal{C} \) be the set of distinct RTWs. We prove the proposition by induction on \( N = |\mathcal{C}| \). We first prove the claim that at every iteration of the algorithm, the set \( \mathcal{S} \) is a collection of sorted time buckets. We prove this claim by induction on \( N \). Suppose it holds for \( N = n \) and consider the case of \( N = n + 1 \). By induction hypothesis, \( \mathcal{S} \) is a collection of sorted buckets at the end of the \( n \)th iteration. It then suffices to show that after calling the procedure \text{REFINE}(\mathcal{S}, [\tau^{n+1}_1, \tau^{n+1}_2])\), the \( \mathcal{S} \) remains to be a collection of sorted time buckets. First, at any iteration of the loop at Line 3, if the condition at Line 4 holds a time bucket in \( \mathcal{S} \) is split into two which maintains the property. Second, notice that at any iteration of the loop at Line 6, the \([q_k, p_{k+1}]\) is the gap between two consecutive time buckets \([p_k, q_k]\) and \([p_{k+1}, q_{k+1}]\) in the original \( \mathcal{C} \). Therefore, the intersection interval \([q_k, p_{k+1}] \cap [\tau^{n+1}_1, \tau^{n+1}_2] \) is still between the original \([p_k, q_k]\) and \([p_{k+1}, q_{k+1}]\), and hence the property is maintained. It is clear that the incorporation of \([\infty, p_{1}] \cap [\tau^{n+1}_1, \tau^{n+1}_2] \) at Line 10 and \([\infty, q_{m}] \cap [\tau^{n+1}_1, \tau^{n+1}_2] \) at Line 12 will also maintain this property since the added intervals appear
at two ends of \( S \). Therefore, \( S \) is a collection of sorted time buckets in the case of \( N = n + 1 \), concluding the proof of the claim.

We focus on proving the bounds. The proof for the lower bound is straightforward, since every time a new interval (distinct from all previous intervals) is added to update \( S \), the cardinality of \( S \) increases by at least one. To prove the upper bound, denote \( S^{(n)} \) as the collection of sorted time buckets at the end of iteration \( n \) of the main algorithm. We notice that \( S^{(n)} \) can be partitioned into \( r \) (for some \( r \geq 1 \)) subsets denoted as \( S^{(n)}_1, \ldots, S^{(n)}_r \), such that the union of all time buckets in any subset \( S^{(n)}_k \) is a single interval and \( (\cup_{i \in S^{(n)}_i} I) \cap (\cup_{j \in S^{(n)}_j} J) = \emptyset \) for all \( i, j \in [r], i \neq j \), where \([r] := \{1, \ldots, r\}\). Every subset \( S_i \) can then be represented as \( S^{(n)}_i = \left\{ [p_k, q_k] \right\}_{k=\alpha(i)}^{\beta(i)} \) with the indices and parameters satisfying the following conditions:

\[
\begin{align*}
\beta(i) & \geq \alpha(i) \quad \text{for } i \in [r], \\
\alpha(i + 1) & = \beta(i) + 1 \quad \text{for } i \in [r - 1], \\
q_k & = p_{k+1} \quad \text{for } k \in \{\alpha(i), \ldots, \beta(i) - 1\}, i \in [r].
\end{align*}
\]

Furthermore, for each \( i \in [r] \) there exist indices \( \sigma(i) \) and \( \lambda(i) \) such that

\[
\begin{align*}
S^{(n)}_i & = \text{ATD}\left( \left\{ [\tau_{1}^{k}, \tau_{2}^{k}] \right\}_{k=\sigma(i)}^{\lambda(i)} \right) \quad \text{for } i \in [r], \\
\lambda(i) & \geq \sigma(i) \quad \text{for } i \in [r], \\
\sigma(i + 1) & = \lambda(i) + 1 \quad \text{for } i \in [r - 1], \\
\sum_{i=1}^{r} (\lambda(i) - \sigma(i) + 1) & = n,
\end{align*}
\]

where \( \text{ATD}\left( \left\{ [\tau_{1}^{k}, \tau_{2}^{k}] \right\}_{k=\sigma(i)}^{\lambda(i)} \right) \) denotes the set of sorted time buckets generated by applying the adaptive time discretization on the set \( \left\{ [\tau_{1}^{k}, \tau_{2}^{k}] \right\}_{k=\sigma(i)}^{\lambda(i)} \) of intervals. Then the induction hypothesis implies that \( |S^{(n)}_i| \leq 2(\lambda(i) - \sigma(i) + 1) - 1 \) for \( i \in [r] \). Consider the procedure \( \text{REFINE}\left( S^{(n)}, [\tau_{1}^{n+1}, \tau_{2}^{n+1}] \right) \).

The operations within the loop at Line 6 will add at most \( r - 1 \) new intervals, and all the other operations together will increase the cardinality of \( S \) by at most 2. Therefore, we have

\[
|S^{(n+1)}| \leq |S^{(n)}| + r + 1 \leq \sum_{i=1}^{r} \left( 2(\lambda(i) - \sigma(i)) + 1 \right) + r + 1 = 2(n - r) + 2r + 1 = 2(n + 1) - 1,
\]

which concludes the proof for the upper bound. Note that the lower and upper bounds can be achieved. The lower bound is achieved when all intervals in \( C \) are disjointed. The upper bound is achieved by taking the following set of intervals: \( \left\{ [2(k - 1), 2k + 1] \right\}_{k=1}^{n} \).

**Theorem 1.** Let \( V = \{1, \ldots, N\} \) be the set of vehicles involved in a MST, \( 0 < h < H \) be two parameters, and \( r = h/H \) be the ratio of the two parameters. Suppose \( C = \left\{ [T_{1}^{k}, T_{2}^{k}] \right\}_{k=1}^{N} \), where \( \{T_{1}^{k} : k \in [N]\} \) are i.i.d. random parameters following the uniform distribution on \([0, H]\). Suppose
\{T^k_i - T^k_j : k \in [N]\} are i.i.d. random parameters independent of anything else that follow the uniform distribution on \([0, h]\). Let \(S\) be the output of Algorithm 1 applied to \(S\). The expectation \(\mathbb{E}[|S|]\) of the total number of time buckets has the following lower and upper bounds:

\[
N + \left(1 - 1/\beta - 1/6\beta^2\right) N \leq \mathbb{E}[|S|] \leq N + \frac{qe}{2(1-q)} N,
\]

where \(q = e(N + 1)r/2\) and \(\beta = (N - 1)r\).

Proof. (a) We first prove for the upper bound. It is clear that the count is invariant under rescaling of the coordinate. To simplify the presentation, we rescale the coordinate in the way that \(H \to 1\), \(h \to h/H = r\). We first sort the sequence \(\{T^k_i : k \in [N]\}\) in ascending order, and let \((k)\) be the vehicle index of the \(k\)th element in the sorted sequence. Then the RTW of vehicle \((k)\) is denoted as \([T^k_1, T^k_2]\). Consider the process of calling \textsc{Refine}(\(S, [T^k_1, T^k_2]\)) for \(k = 1, \ldots, N\) in Algorithm 1. Excluding a certain zero-measured set of events, it can be seen that when a new RTW \([T^k_1, T^k_2]\) is incorporated to refine the collection \(S\) of sorted time buckets, \(|S|\) increases by either one or two, and the increment is two if and only if at least one RTW from \(\{[T^{(i)}_1, T^{(i)}_2] : i \in [k - 1]\}\) has a proper intersection with \([T^k_1, T^k_2]\). This observation leads to the following representation of \(|S|\) when Algorithm 1 terminates:

\[
|S| = N + \sum_{k=2}^{N} I_k,
\]

where \(I_k\) is the indicator of the event \(\bigvee_{i=2}^{k} \{[T^{(k)}_1, T^{(k)}_2] \cap [T^{(i)}_1, T^{(i)}_2] \text{ is proper}\}\). It follows that

\[
\mathbb{E}[|S|] = N + \sum_{k=2}^{N} \mathbb{E}[I_k] = N + \sum_{k=2}^{N} \mathbb{P}(I_k = 1) = N + \sum_{k=2}^{N} \sum_{i=1}^{k-1} \mathbb{P}(T^{(k)}_1 \in [T^{(i)}_1, T^{(i)}_2])
\]

\[
\leq N + \sum_{k=2}^{N} \sum_{i=1}^{k-1} \mathbb{P}(T^{(k)}_1 \in [T^{(i)}_1, T^{(i)}_2])
\]

\[
= N + \sum_{k=2}^{N} \sum_{i=1}^{k-1} \mathbb{P}(T^{(k)}_1 - T^{(i)}_1 \leq rU(i))
\]

where in the third equality we use the assumption that the first end points \(\{T^{(i)}_1\}_{i=1}^{n}\) are sorted in ascending order, indicating that the event \(\{T^{(k)}_1 \in [T^{(i)}_1, T^{(i)}_2] \text{ or } T^{(k)}_2 \in [T^{(i)}_1, T^{(i)}_2]\}\) is equivalent to \(\{T^{(k)}_1 \in [T^{(i)}_1, T^{(i)}_2]\}\), and in the last equality the uniformly distributed random variable \(U(i) \sim U(0, 1)\) is independent of other random parameters. So it amounts to derive an upper bound for \(\mathbb{P}(T^{(k)}_1 - T^{(i)}_1 \leq rU(i))\). It is well known that the ascending sequence \(\{T^{(k)}_1\}_{k=1}^{N}\) follows the order statistics (Durham University 2017, DasGupta 2008). From (DasGupta 2008), the random parameters \((T^{(i)}_1, T^{(k)}_1)\) have the following joint density function

\[
f_{i,k}(u,v) = \frac{N!}{(i-1)!((N-k)!(k-i-1)!} u^{i-1}(1-v)^{N-k}(v-u)^{k-i-1},
\]

where \(u = T^{(i)}_1\) and \(v = T^{(k)}_1\).
and hence the probability density function of the gap \( T^{(k)}_1 - T^{(i)}_1 \) can be derived:

\[
g_{i,k}(t) = \int_0^{1-t} f_{i,k}(u,u+t) \, du
\]

\[
= \frac{N!}{(i-1)!((N-k)!(k-i-1)!)} \cdot t^{k-i-1} \cdot \int_0^{1-t} u^{i-1}(1-u-t)^{N-k} \, du
\]

\[
= \frac{N!}{(i-1)!((N-k)!(k-i-1)!)} \cdot (1-t)^{N-k+i} \cdot t^{k-i-1}(1-t)^{N-k+i} \quad \text{(integrate by parts)}
\]

\[
= \frac{N!}{(N-k+i)!(k-i-1)!} \cdot t^{k-i-1}(1-t)^N.
\]

Therefore, we find out that

\[
\mathbb{P}(T^{(k)}_1 - T^{(i)}_1 \leq s) = \int_0^s g_{i,k}(t) \, dt = \int_0^s \frac{N!}{(N-k+i)!(k-i-1)!} \cdot (1-t)^{N-k+i} \cdot t^{k-i-1}(1-t)^N \, dt
\]

\[
\leq \int_0^s \frac{N!}{(N-k+i)!(k-i-1)!} \cdot t^{k-i-1} = \frac{N!}{(N-k+i)!(k-i-1)!} \cdot s^{k-i} = \left( \frac{N}{k-i} \right) s^{k-i}.
\]

It then follows that

\[
\mathbb{P}(T^{(k)}_1 - T^{(i)}_1 \leq rU_i) = \int_0^1 \mathbb{P}(T^{(k)}_1 - T^{(i)}_1 \leq rs) \, ds \leq \left( \frac{N}{k-i} \right) \frac{1}{(N+1)(k-i+1)} \int_0^1 (rs)^{k-i} \, ds
\]

\[
= \left( \frac{N}{k-i} \right) \frac{1}{(N+1)(k-i+1)} \cdot \frac{1}{r^{k-i+1}} \cdot \frac{1}{(N+1)r} \cdot \sum_{i=1}^{k-1} \left( \frac{e(N+1)r}{k-i+1} \right)^{k-i+1}
\]

\[
= \frac{q(1-q^{k-1})e}{2(1-q)} \quad \text{for } q \neq 1,
\]

where we define \( q := e(N+1)r/2 \) and we use the upper bound \( \binom{n}{k} \leq \left( \frac{en}{k} \right)^k \) for the binomial coefficient to get the last inequality. An upper bound for \( \mathbb{E}[|S|] \) follows as

\[
\mathbb{E}[|S|] = N + \sum_{k=2}^N \mathbb{P}(I_k = 1) \leq N + \sum_{k=2}^N \frac{q(1-q^{k-1})e}{2(1-q)}
\]

\[
= N + \frac{(N-1)qe}{2(1-q)} - \frac{q^2(N-q^N)e}{2(1-q)^2}
\]

\[
\leq N + \frac{qe}{2(1-q)} N,
\]

which concludes the proof for the upper bound.

(b) Now we prove for the lower bound. Let \( I_k' \) be the indicator of the event \( \{ [T^{(k)}_1, T^{(k)}_2] \cap [T^{(k-1)}_1, T^{(k-1)}_2] \text{ is proper} \} \) for \( k \in \{2, \ldots, N\} \). Clearly, we have \( I_k \geq I_k' \) almost surely, which leads to the following lower bound of \( \mathbb{E}[|S|] \):

\[
\mathbb{E}[|S|] \geq N + \sum_{k=2}^N \mathbb{P}(I_k' = 1) = N + \sum_{k=2}^N \mathbb{P}(T^{(k)}_1 - T^{(k-1)}_1 \leq r(U^{(k-1)})) \quad \text{(EC.14)}
\]
From the joint probability density (EC.9), we can compute the probability

$$\mathbb{P}(T_1^{(k)} - T_1^{(k-1)} \leq s) = \int_0^s g_{k-1,k}(t) = 1 - (1 - s)^N.$$ 

It then follows that

$$\mathbb{E}[|S|] \geq N + \sum_{k=2}^{N} \int_0^1 \mathbb{P}(T_1^{(k)} - T_1^{(k-1)} \leq rs) ds$$

$$= N + \sum_{k=2}^{N} \int_0^1 [1 - (1 - rs)^N] ds$$

$$= 2N - 1 - \frac{N - 1}{(N + 1)r} [1 - (1 - r)^{N+1}]$$

$$= 2N - 1 - \frac{N - 1}{(N + 1)r} + \frac{N - 1}{(N + 1)r} (1 - r)^{N+1}$$

$$\geq 2N - 1 - \frac{N - 1}{(N + 1)r} + \frac{N - 1}{(N + 1)r} \left[ 1 - (N + 1)r + \left( \frac{N + 1}{2} \right) r^2 - \left( \frac{N + 1}{3} \right) r^3 \right]$$

$$= N + \frac{N - 1}{(N + 1)r} \left[ \left( \frac{N + 1}{2} \right) r^2 - \left( \frac{N + 1}{3} \right) r^3 \right]$$

$$= N + \frac{1}{2} N(N - 1) r - \frac{1}{6} N(N - 1)^2 r^2$$

$$= N + \left( \frac{1}{2} \beta - \frac{1}{6} \beta^2 \right) N,$$

which concludes the proof. \( \square \)

**Theorem 2.** The mapping \( \varphi \) is surjective. Let \( \pi \) be any feasible schedule to the FRCVP problem. Then \( \text{Obj}^{\text{FRCVP}}(\pi) = \text{Obj}^{\text{VA}} \circ \varphi(\pi) \), where \( \text{Obj}^{\text{FRCVP}} \) and \( \text{Obj}^{\text{VA}} \) are objective function of the FRCVP problem and (VA), respectively. Let \( \chi^* \) be an optimal solution of (VA), then any schedule \( \pi^* \in \varphi^{-1}(\chi^*) \) is an optimal solution of the FRCVP problem if it satisfies that \( \pi_u = \pi_v^* \) when \( u \) and \( v \) are assigned to the same time bucket by \( \chi^* \).

**Proof.** For any feasible time-bucket assignment \( \chi = \{ \chi_v : v \in \mathcal{V} \} \), \( \chi_v \) indexes a feasible time bucket of vehicle \( v \). Let \( \pi_v \) be the middle point of the time bucket \( \chi_v \). It is clear that the schedule \( \pi = \{ \pi_v : v \in \mathcal{V} \} \) is feasible to the FRCVP problem and \( \varphi(\pi) = \chi \), which shows that the mapping \( \varphi \) is surjective. Consider a feasible schedule \( \pi \). By Lemma 1, any two vehicles \( u \) and \( v \) are pseudo-platoonable if and only if \( \pi_u = \pi_v \). Since \( \pi_u = \pi_v \) implies that \( \varphi_u(\pi) = \varphi_v(\pi) \). This means that the feasible platooning configurations are preserved under the mapping \( \varphi \), and hence \( \text{Obj}^{\text{FRCVP}}(\pi) = \text{Obj}^{\text{VA}} \circ \varphi(\pi) \). To prove the last assertion, let \( \chi^* \) be an optimal time-bucket assignment to (VA) and \( \pi^* \in \varphi^{-1}(\chi^*) \) be any schedule that is mapped to \( \chi^* \) under \( \varphi \) satisfying \( \pi_u = \pi_v^* \) if \( \chi_u = \chi_v^* \). Suppose \( \pi^* \) is not an optimal schedule to the FRCVP and let \( \pi' \) be an optimal schedule. The non-optimality of \( \pi^* \) indicates that \( \text{Obj}^{\text{FRCVP}}(\pi^*) < \text{Obj}^{\text{FRCVP}}(\pi') \). Since we have proved that \( \text{Obj}^{\text{FRCVP}}(\pi^*) = \text{Obj}^{\text{VA}} \circ \varphi(\pi^*) \) and \( \text{Obj}^{\text{FRCVP}}(\pi') = \text{Obj}^{\text{VA}} \circ \varphi(\pi') \), we conclude that \( \text{Obj}^{\text{VA}} \circ \varphi(\pi^*) < \text{Obj}^{\text{VA}} \circ \varphi(\pi') \).
Therefore, the time-bucket assignment \( \varphi(\pi') \) must be different from \( \chi^* = \varphi(\pi^*) \), and it achieves a better objective value for (VA), contradicting to the optimality of \( \chi^* \). \( \square \)

**Theorem 3.** The FRCVP problem with the route graph being a tree and with no limit on the platoon size is NP-hard.

Proof. We prove the theorem by reducing the scheduling problem of unit execution time tasks with unbounded number of machines (Sched-UET-\( \infty \)) to the tree-based FRCVP problem. The Sched-UET-\( \infty \) problem is known to be NP-complete (Ullman 1973, Papadimitriou and Yannakakis 1979), and it states as follows: Given \( n \) tasks, each has a time window for execution and the two end coordinates of every time window are non-negative integers. Every task can be completed in one unit of time on a machine, and there are unlimited amount of machines at every time instant. This means if a time unit is feasible for multiple tasks, these tasks can be assigned to the corresponding number of machines and get all completed in one unit of time. The goal is to find the minimum number of machine operations to complete all tasks. Note that if multiple machines are working in parallel, it counts as a single operation. Note that the Sched-UET-\( \infty \) is equivalent to the minimal clique partition of an interval graph.

For any instance of the Sched-UET-\( \infty \) problem, we can construct an instance of the tree-based FRCVP problem such that solving the tree-based FRCVP instance can give a solution to the Sched-UET-\( \infty \) instance. The construction is as follows: The \( n \) tasks are mapped to \( n \) vehicles. We can construct the route of each vehicle such that they only share exactly the same edge \( e \) along their routes. Picking the origin node of an arbitrary vehicle as the root. For every vehicle, we can make up the starting and destination time as well as the length of other edges along its route such that the RTW of the vehicle is the same as the time window of the corresponding task. We have the following claim hold.

**Claim.** If \( \sigma^l = 0 \) and \( \sigma^f > 0 \), solving the tree-based FRCVP instance constructed above can induce an optimal solution to the Sched-UET-\( \infty \) instance.

**Proof.** When the tree-based FRCVP instance has been solved to optimality, the solution can give \( m \) different time units (time buckets) such that the set of vehicles can be divided into \( m \) groups, and vehicles within a same group form a platoon on the shared edge \( e \). The total amount of fuel saving is equal to \( (n - m)\sigma^f C_e \) since a lead vehicle cannot save fuel by the parameter setting. Therefore, maximizing the total fuel saving is equivalent to minimizing the number of partitions of the set of vehicles. Based on the construction, the number of partitions in the FRCVP instance is equal to the number of machine operations in the Sched-UET-\( \infty \) instance. This concludes the proof. \( \square \)

**Proposition 4.** Let \( V_t \) be the subset of vehicles for which the time bucket \( t \) is feasible. Suppose the inequality \( H(V_t) \leq \frac{1}{4}\text{OPT} \) hold for all \( t \in [T] \). Then there exists a polynomial algorithm with complexity \( O(|V||T||E|) \) that can identify an optimal solution of the tree-based FRCVP problem.
Proof. Let \( V_i^* \) be the subset of vehicles assigned to \( t \) in an optimal solution. Since \( \sum_{t \in \mathcal{T}} H(V_i^*) = \text{OPT} \) by the definition of optimal solution and \( H(V_i^*) \leq H(V_i) \leq \frac{1}{T} \text{OPT} \) by assumption, it follows that \( H(V_i^*) = H(V_i) = \frac{1}{T} \text{OPT} \) for all \( t \in \mathcal{T} \). Notice that \( V_i^* \) is a subset of \( V_i \) for all \( t \). The above equalities indicate that assigning vehicles from the set \( V_i \setminus V_i^* \) to \( t \) does not increase any fuel saving as compared to \( V_i^* \). It further implies that for any three distinct vehicles \( u \in V_i \setminus V_i^*, v \in V_i \setminus V_i^* \) and \( w \in V_i^* \), any mutual intersection of the three sets \( \mathcal{R}_u, \mathcal{R}_v \) and \( \mathcal{R}_w \) is an empty set. For every \( t \), we define the following two subsets \( U_t := \{ u : \exists v \in V_i, v \neq u, \text{s.t. } \mathcal{R}_u \cap \mathcal{R}_v \neq \emptyset \} \) and \( W_t := \{ w : \forall v \in V_i, v \neq w, \mathcal{R}_w \cap \mathcal{R}_v = \emptyset \} \). It is straightforward to check that \( U_t \) and \( W_t \) are two disjoint classes and \( U_t = U_t \cup W_t \) for all \( t \). By definition, we have \( H(U_t) = H(U_t \cup A) = \frac{1}{T} \text{OPT} \) for any subset \( A \subseteq V_i \setminus U_t \) and for all \( t \). Furthermore, the subsets \( \{U_t : t \in \mathcal{T}\} \) are disjoint. Indeed, if there exists a \( v \in U_t \cap U_{t'} \) for two time buckets \( t \) and \( t' \), consider where \( v \) has been assigned in the optimal solution \( \{V_i^* : t \in \mathcal{T}\} \). Clearly, at least one of \( V_i^* \) and \( V_i^* \) does not contain \( v \). Without loss of generality, assume \( v \notin V_i^* \). Then we conclude that

\[
H(V_i^*) = H(V_i^* \cap U_t) \leq H(U_t \setminus \{v\}) < \frac{1}{T} \text{OPT},
\]

which leads to a contradiction. Note that each \( U_t \) can be constructed with computational complexity \( \mathcal{O}(|\mathcal{V}||\mathcal{E}|) \) and the overall computational complexity is \( \mathcal{O}(|\mathcal{V}||\mathcal{E}||\mathcal{T}|) \) for constructing \( U_t \) for all \( t \in \mathcal{T} \).

For every \( v \in \mathcal{V} \setminus (\cup_{t=1}^{T} U_t) \), it can be assigned to an arbitrary feasible time bucket without changing the objective value. \( \square \)

**Lemma 2.** Let \( t_1 \) and \( t_2 \) be two different time buckets. Let \( V_1 \) (resp. \( V_2 \)) be a subset of vehicles that are only feasible at \( t_1 \) (resp. \( t_2 \)). Let \( V_3 \) be a subset of vehicles that are feasible at both \( t_1 \) and \( t_2 \). Let \( V_3 = U_1 \cup U_2 \) be any 2-partition of \( V_3 \). Let \( \mathcal{E}_0 = \{ e \in \cup_{v \in V_3} \mathcal{R}_v : n_e(V_3) \geq 2 \} \) be the subset of edges that are shared by at least two vehicles in \( V_3 \). Then the following property holds:

\[
H(V_1 \cup U_1, \mathcal{E}_0) + H(V_2 \cup U_2, \mathcal{E}_0) - H(V_1, \mathcal{E}_0) - H(V_2, \mathcal{E}_0) \leq 2H(V_3, \mathcal{E}_0). \tag{11}
\]

Proof. Let \( \mathcal{E}_1 = \mathcal{E}_0 \cap (\cup_{e \in U_1} \mathcal{R}_v) \) and \( \mathcal{E}_2 = \mathcal{E}_0 \cap (\cup_{e \in U_2} \mathcal{R}_v) \). Notice that \( \mathcal{E}_0 = \mathcal{E}_1 \cup \mathcal{E}_2 \). For any \( e \in \mathcal{E}_1 \setminus \mathcal{E}_2 \), we should have \( n_e(U_1) \geq 2 \), since \( n_e(U_1 \cup U_2) \geq 2 \) but \( e \notin \cup_{e \in U_2} \mathcal{R}_v \). The difference on the left side of (11) can be evaluated as follows

\[
H(V_1 \cup U_1, \mathcal{E}_0) + H(V_2 \cup U_2, \mathcal{E}_0) - H(V_1, \mathcal{E}_0) - H(V_2, \mathcal{E}_0) = \sum_{e \in \mathcal{E}_1} C_e[\phi \circ n_e(V_1 \cup U_1) - \phi \circ n_e(V_1) + \phi \circ n_e(V_2 \cup U_2) - \phi \circ n_e(V_2)]
\]

\[
= \sum_{e \in \mathcal{E}_1 \setminus \mathcal{E}_2} C_e[\phi \circ n_e(V_1 \cup U_1) - \phi \circ n_e(V_1)] + \sum_{e \in \mathcal{E}_2 \setminus \mathcal{E}_1} C_e[\phi \circ n_e(V_2 \cup U_2) - \phi \circ n_e(V_2)] + \sum_{e \in \mathcal{E}_1 \cap \mathcal{E}_2} C_e[\phi \circ n_e(V_1 \cup U_1) - \phi \circ n_e(V_1) + \phi \circ n_e(V_2 \cup U_2) - \phi \circ n_e(V_2)], \tag{EC.17}
\]
where the last equality is due to the fact that for any \( e \in \mathcal{E}_1 \setminus \mathcal{E}_2 \), \( n_e(\mathcal{U}_2) = 0 \) and hence \( n_e(\mathcal{V}_2 \cup \mathcal{U}_2) - n_e(\mathcal{V}_2) = 0 \). To evaluate the first term in (EC.17) for a \( e \in \mathcal{E}_1 \), there are only three cases on \( n_e(\mathcal{V}_1) \) and \( n_e(\mathcal{U}_1) \): (1) \( n_e(\mathcal{V}_1) \leq 1 \), \( n_e(\mathcal{V}_1) + n_e(\mathcal{U}_1) \geq 2 \); (2) \( n_e(\mathcal{V}_1) \geq 2 \), \( n_e(\mathcal{V}_1) + n_e(\mathcal{U}_1) \geq 2 \). For each case, we have

\[
\phi \circ n_e(\mathcal{V}_1 \cup \mathcal{U}_1) - \phi \circ n_e(\mathcal{V}_1) = \begin{cases} 
\sigma' + \sigma'[n_e(\mathcal{V}_1) + n_e(\mathcal{U}_1) - 1] & \text{Case (1)}, \\
\sigma' n_e(\mathcal{U}_1) & \text{Case (2)}. 
\end{cases}
\tag{EC.18}
\]

Notice that since \( e \in \mathcal{E}_1 \setminus \mathcal{E}_2 \), we have \( n_e(\mathcal{U}_1) = n_e(\mathcal{V}_3) \), and hence the quantity \( \sigma' + \sigma'[n_e(\mathcal{V}_1) + n_e(\mathcal{U}_1) - 1] \) in Case (1) is upper bounded by \( \sigma' + \sigma'[n_e(\mathcal{V}_1) - 1] = \phi \circ n_e(\mathcal{V}_3) \). Then the following inequality holds for the two cases

\[
\phi \circ n_e(\mathcal{V}_1 \cup \mathcal{U}_1) - \phi \circ n_e(\mathcal{V}_1) \leq \frac{2\sigma_f}{\sigma_f + \sigma_f} \phi \circ n_e(\mathcal{V}_3),
\tag{EC.19}
\]

where the equality holds at Case (2) when \( n_e(\mathcal{V}_1) \geq 2 \) and \( n_e(\mathcal{U}_1) = n_e(\mathcal{V}_3) = 2 \). Evaluation of the second term in (EC.17) leads to a similar inequality as (EC.19). The third term in (EC.17) can be evaluated in the same manner for different cases of the \([n_e(\mathcal{V}_1), n_e(\mathcal{U}_1), n_e(\mathcal{V}_2), n_e(\mathcal{U}_2)]\)-combination. One can show that the following inequality holds for all cases

\[
\phi \circ n_e(\mathcal{V}_1 \cup \mathcal{U}_1) - \phi \circ n_e(\mathcal{V}_1) + \phi \circ n_e(\mathcal{V}_2 \cup \mathcal{U}_2) - \phi \circ n_e(\mathcal{V}_2) \leq 2\phi \circ n_e(\mathcal{V}_3), \tag{EC.20}
\]

where the equality holds at \( n_e(\mathcal{V}_1) = n_e(\mathcal{U}_1) = n_e(\mathcal{V}_2) = n_e(\mathcal{U}_2) = 1 \). Notice that since \( H(\mathcal{V}_3, \mathcal{E}_0) = \sum_{e \in \mathcal{E}_0} \phi \circ n_e(\mathcal{V}_3) \), combining the inequalities (EC.19) and (EC.20) concludes the proof. \( \Box \)

**Lemma 3.** Let \( t_1 \) and \( t_2 \) be two time buckets. Let the sets \( \mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3 \) and \( \mathcal{E}_0 \) be defined the same as in Lemma 2. Let \( H^* = \max_{t \in \mathcal{V}_3} H(\mathcal{V}_1 \cup \mathcal{U}) + H(\mathcal{V}_2 \cup \mathcal{V}_3 \setminus \mathcal{U}) \), and \( H_0 \) be the value returned by the function \( \text{Assignment}(t_1, t_2) \) in Algorithm 2. Then the following bounds hold:

\[
H^* - H_0 \leq 2 \left( 1 + \frac{\sigma'}{\sigma_f} \right) \left[ H(\mathcal{V}_1 \cup \mathcal{V}_3, \mathcal{E}_0) + H(\mathcal{V}_2 \cup \mathcal{V}_3, \mathcal{E}_0) - H(\mathcal{V}_1 \cup \mathcal{U}_1^*, \mathcal{E}_0) - H(\mathcal{V}_2 \cup \mathcal{U}_2^*, \mathcal{E}_0) \right], \tag{12}
\]

where \( \mathcal{U}_1^* \) be an optimal subset of \( \mathcal{V}_3 \) that leads to the optimal value \( H^* \), and \( \mathcal{U}_2^* = \mathcal{V}_3 \setminus \mathcal{U}_1^* \).

Proof. Let \( \mathcal{E}_0^c \) be the complement set of \( \mathcal{E}_0 \) with respect to the set \( \mathcal{E} \) of all edges. The value \( H^* \) can be decomposed as follows:

\[
H^* = H(\mathcal{V}_1 \cup \mathcal{U}_1^*) + H(\mathcal{V}_2 \cup \mathcal{U}_2^*) \tag{EC.21}
\]

Let \( \mathcal{U}_1 \) and \( \mathcal{U}_2 \) be the two subsets constructed in the function \( \text{Assignment}(t_1, t_2) \). The value \( H_0 \) can then be decomposed in a similar way:

\[
H_0 = H(\mathcal{V}_1 \cup \mathcal{U}_1) + H(\mathcal{V}_2 \cup \mathcal{U}_2) \tag{EC.22}
\]
Due to the definition of $\mathcal{E}_0$, for any $v \in \mathcal{V}_3$ there is no other vehicles from $\mathcal{V}_3$ that travel on any edge of $\mathcal{E}_v := \mathcal{R}_v \setminus \mathcal{E}_0$ (defined in Assignment). Therefore, for any subset $\mathcal{U}$ of $\mathcal{V}_3$ the following additive property holds on the edge set $\mathcal{E}_0^c$:

$$
H(\mathcal{V}_1 \cup \mathcal{U}, \mathcal{E}_0^c) - H(\mathcal{V}_1, \mathcal{E}_0^c) = \sum_{v \in \mathcal{U}} [H(\mathcal{V}_1 \cup \{v\}, \mathcal{E}_0^c) - H(\mathcal{V}_1, \mathcal{E}_v^c)] = \sum_{v \in \mathcal{U}} [H(\mathcal{V}_1 \cup \{v\}, \mathcal{E}_v) - H(\mathcal{V}_1, \mathcal{E}_v)],$

$$
H(\mathcal{V}_2 \cup \mathcal{U}, \mathcal{E}_0^c) - H(\mathcal{V}_2, \mathcal{E}_0^c) = \sum_{v \in \mathcal{U}} [H(\mathcal{V}_2 \cup \{v\}, \mathcal{E}_0^c) - H(\mathcal{V}_2, \mathcal{E}_v^c)] = \sum_{v \in \mathcal{U}} [H(\mathcal{V}_2 \cup \{v\}, \mathcal{E}_v) - H(\mathcal{V}_2, \mathcal{E}_v)].$

(EC.23)

With the help of (EC.23), the greedy policy of assigning vehicles to $t_1$ and $t_2$ used in the function Assignment$(t_1, t_2)$ guarantees that

$$
H(\mathcal{V}_1 \cup \mathcal{U}_1, \mathcal{E}_0^c) + H(\mathcal{V}_2 \cup \mathcal{U}_2, \mathcal{E}_0^c) \geq H(\mathcal{V}_1 \cup \mathcal{U}_1^*, \mathcal{E}_0^c) + H(\mathcal{V}_2 \cup \mathcal{U}_2^*, \mathcal{E}_0^c) \quad \text{(EC.24)}
$$

Substituting (EC.24) into (EC.21) and (EC.22) yields

$$
H^* - H_0 \leq H(\mathcal{V}_1 \cup \mathcal{U}_1^*, \mathcal{E}_0) + H(\mathcal{V}_2 \cup \mathcal{U}_2^*, \mathcal{E}_0) - H(\mathcal{V}_1 \cup \mathcal{U}_1, \mathcal{E}_0) - H(\mathcal{V}_2 \cup \mathcal{U}_2, \mathcal{E}_0) \quad \text{(EC.25)}
$$

Using Lemma 2, we have

$$
H(\mathcal{V}_1 \cup \mathcal{U}_1^*, \mathcal{E}_0) + H(\mathcal{V}_2 \cup \mathcal{U}_2^*, \mathcal{E}_0) \leq H(\mathcal{V}_1, \mathcal{E}_0) + H(\mathcal{V}_2, \mathcal{E}_0) + 2H(\mathcal{V}_3, \mathcal{E}_0). \quad \text{(EC.26)}
$$

On the other side, the following lower bound on $H(\mathcal{V}_1 \cup \mathcal{U}_1^*, \mathcal{E}_0) + H(\mathcal{V}_2 \cup \mathcal{U}_2^*, \mathcal{E}_0)$ obviously holds

$$
H(\mathcal{V}_1 \cup \mathcal{U}_1, \mathcal{E}_0) + H(\mathcal{V}_2 \cup \mathcal{U}_2, \mathcal{E}_0) \geq H(\mathcal{V}_1, \mathcal{E}_0) + H(\mathcal{V}_2, \mathcal{E}_0). \quad \text{(EC.27)}
$$

Substituting (EC.26) and (EC.27) into (EC.25) leads to

$$
H^* - H_0 \leq 2H(\mathcal{V}_3, \mathcal{E}_0). \quad \text{(EC.28)}
$$

It then suffices to bound $H(\mathcal{V}_3, \mathcal{E}_0)$ with the term on the right side of (12). We observe that

$$
H(\mathcal{V}_1 \cup \mathcal{V}_3, \mathcal{E}_0) - H(\mathcal{V}_1 \cup \mathcal{U}_1^*, \mathcal{E}_0) \geq \frac{\sigma^f}{\sigma^f + \sigma^t}[H(\mathcal{V}_3, \mathcal{E}_0) - H(\mathcal{U}_1^*, \mathcal{E}_0)],$

$$
H(\mathcal{V}_2 \cup \mathcal{V}_3, \mathcal{E}_0) - H(\mathcal{V}_2 \cup \mathcal{U}_2^*, \mathcal{E}_0) \geq \frac{\sigma^f}{\sigma^f + \sigma^t}[H(\mathcal{V}_3, \mathcal{E}_0) - H(\mathcal{U}_2^*, \mathcal{E}_0)]. \quad \text{(EC.29)}
$$

The validness of above two inequalities can be seen from the following extremal case: Suppose for an edge $e \in \mathcal{E}_0$, it is shared by three vehicles $v \in \mathcal{V}_1$, $u_1 \in \mathcal{U}_1^*$ and $u_2 \in \mathcal{V}_3 \setminus \mathcal{U}_1^*$. If only considering the two vehicles $u_1$ and $u_2$, adding $u_2$ to follow $u_1$ in a platoon leads to the amount of fuel saving increased from 0 to $(\sigma^t + \sigma^f)C_e$. On the other side, if $v$ is included in the consideration such that $\{v, u_1\}$ is a platoon, adding $u_2$ to the platoon leads to the amount of fuel saving increased from $(\sigma^t + \sigma^f)C_e$ to $(\sigma^t + 2\sigma^f)C_e$. The increment of fuel saving in the two cases is $(\sigma^t + \sigma^f)C_e$ versus
Applying Lemma 3 gives the following inequality (EC.29)

\[ H(V_1 \cup V_3, \mathcal{E}_0) + H(V_2 \cup V_3, \mathcal{E}_0) - H(V_1 \cup U_1^c, \mathcal{E}_0) - H(V_2 \cup U_2^c, \mathcal{E}_0) \geq \frac{\sigma_f}{\sigma^f + \sigma_f} [2H(V_3, \mathcal{E}_0) - H(U_1^c, \mathcal{E}_0) - H(U_2^c, \mathcal{E}_0)] \] (EC.30)

which concludes the proof. \( \square \)

**Theorem 4.** Let \( \text{OPT} \) be the optimal value of a tree-based FRCVP instance with \( T \) uniform time buckets. The objective \( H_0 \) given by Algorithm 2 satisfies the following lower bound:

\[
H_0 \geq \left( \frac{4\sigma^f + 6\sigma_f}{4\sigma^f + 5\sigma_f} \right) \frac{1}{T} \text{OPT}. \tag{13}
\]

Proof Let \( \mathcal{W}_t \) be the subset of vehicles for which \( t \) is a feasible time bucket. Let \( H_{ub} = \max_{t \in [T]} H(\mathcal{W}_t) \). Let \( \mathcal{W}_t^* \) for \( t \in [T] \) be the subset of vehicles assigned to \( t \) in an optimal solution of the FRCVP instance. Clearly, \( \mathcal{W}_t^* \) is a subset of \( \mathcal{W}_t \), and \( \text{OPT} = \sum_{t=1}^{T} H(\mathcal{W}_t^*) \) by definition. Let \( t_1 \) and \( t_2 \) be two time buckets such that \( H(\mathcal{W}_{t_1}^*) \geq H(\mathcal{W}_{t_2}^*) \geq H(\mathcal{W}_s^*) \) for all \( s \in [T] \setminus \{t_1, t_2\} \). For simplicity of notation, let \( H_1 := H(\mathcal{W}_{t_1}^*) \) and \( H_2 := H(\mathcal{W}_{t_2}^*) \). The choice of \( t_1 \) and \( t_2 \) implies that \( H_1 \geq \frac{1}{T} \text{OPT} \) and \( H_2 \geq \frac{1}{T-1} (\text{OPT} - H_1) \).

Let \( V_1, V_2, V_3 \) and \( \mathcal{E}_0 \) be defined the same as in Lemma 3 for \( t_1 \) and \( t_2 \). Let \( H^* = \max_{U \subseteq V_3} H(V_1 \cup U) + H(V_2 \cup V_3 \setminus U) \) with \( U_1^c \subseteq V_3 \) and \( U_2^c = V_3 \setminus U_1^c \) being an optimal solution of the problem. Applying Lemma 3 gives the following inequality

\[
H^* - H_0 \leq 2 \left(1 + \frac{\sigma^f}{\sigma_f}\right) [H(V_1 \cup V_3, \mathcal{E}_0) + H(V_2 \cup V_3, \mathcal{E}_0) - H(V_1 \cup U_1^c, \mathcal{E}_0) - H(V_2 \cup U_2^c, \mathcal{E}_0)]
\]

\[
= 2 \left(1 + \frac{\sigma^f}{\sigma_f}\right) [H(V_1 \cup V_3) + H(V_2 \cup V_3) - H(V_1 \cup U_1^c) - H(V_2 \cup U_2^c)]
\]

\[
- 2 \left(1 + \frac{\sigma^f}{\sigma_f}\right) [H(V_1 \cup V_3, \mathcal{E}_0^c) + H(V_2 \cup V_3, \mathcal{E}_0^c) - H(V_1 \cup U_1^c, \mathcal{E}_0^c) - H(V_2 \cup U_2^c, \mathcal{E}_0^c)]
\]

\[
= 2 \left(1 + \frac{\sigma^f}{\sigma_f}\right) [H(V_1 \cup V_3) + H(V_2 \cup V_3) - H(V_1 \cup U_1^c) - H(V_2 \cup U_2^c)]
\]

\[
\leq 2 \left(1 + \frac{\sigma^f}{\sigma_f}\right) [2H_{ub} - H_1 - H_2], \tag{EC.31}
\]

where the first inequality is directly from (12) in Lemma 3, the second equality uses the additive property given in (EC.23), and the second inequality uses the property \( \max\{H(V_1 \cup V_3), H(V_2 \cup V_3)\} \leq H_{ub} \) by the definition of \( H_{ub} \) and the definition of \( U_1^c \) and \( U_2^c \). On the other side, we have \( H^* \geq H_1 + H_2 \). Substituting this lower bound of \( H^* \) into (EC.31) leads to the following inequality

\[
H_1 + H_2 - H_0 \leq 2 \left(1 + \frac{\sigma^f}{\sigma_f}\right) [2H_{ub} - H_1 - H_2], \tag{EC.32}
\]
which further implies that
\[
H_0 \geq \left[ 1 + 2 \left( 1 + \frac{\sigma^l}{\sigma_f} \right) \right] (H_1 + H_2) - 4 \left( 1 + \frac{\sigma^l}{\sigma_f} \right) H_{ub} \\
\geq \left[ 1 + 2 \left( 1 + \frac{\sigma^l}{\sigma_f} \right) \right] \frac{1}{T-1} \text{OPT} + \left[ 1 + 2 \left( 1 + \frac{\sigma^l}{\sigma_f} \right) \right] \frac{T-2}{T-1} H_1 - 4 \left( 1 + \frac{\sigma^l}{\sigma_f} \right) H_{ub} \quad \text{(EC.33)}
\]
\[
:= f(H_1, H_{ub}),
\]
where \( H_2 \geq \frac{1}{T-1} (\text{OPT} - H_1) \) is used to get the second inequality, and \( f(H_1, H_{ub}) \) represents the term on the right side of the second inequality in (EC.33) which is a function of \( H_1 \) and \( H_{ub} \). From Line 3 of Algorithm 2 and (EC.33), the value \( H_0 \) has the following lower bound \( H_0 \geq \max\{H_{ub}, f(H_1, H_{ub})\} \). It amounts to find appropriate \( H_1 \) and \( H_{ub} \) that can minimize the lower bound. The lower bound minimization problem can be formulated as
\[
\min_{H_1, H_{ub}} \max \{H_{ub}, f(H_1, H_{ub})\} \quad \text{s.t.} \quad \frac{1}{T} \text{OPT} \leq H_1 \leq H_{ub}. \tag{EC.34}
\]
It can be verified that the optimal objective of the above problem is obtained by setting \( H_{ub} = f(H_1, H_{ub}) \) and \( H_1 = \frac{1}{T} \text{OPT} \), which gives the minimum lower bound
\[
H_0 \geq \frac{4\sigma^f + 6\sigma^l}{4\sigma^l + 5\sigma^f} \frac{\text{OPT}}{T}.
\]
This concludes the proof. \( \square \)

**Lemma 4.** Suppose Assumption 2 hold. Let \( \pi \) and \( \pi^* \) be an arbitrary schedule and an optimal schedule, respectively. For an edge \( e \in \mathcal{R} \), let \( \text{obj}(\pi, e) \) and \( \text{obj}(\pi^*, e) \) be the objective value (the amount of fuel saving) contributed by \( e \) under the schedule \( \pi \) and \( \pi^* \), respectively. If the edge \( e \in \mathcal{R} \) is shared by \( N \) vehicles, where \( N \geq T + 1 \), then \( \frac{\text{obj}(\pi, e)}{\text{obj}(\pi^*, e)} \geq 1 - \frac{T}{N} \).

Proof. Clearly, under any schedule, the objective value achieved at the edge \( e \) should be upper bounded by \( [(N-1)\sigma^f + \sigma^l] C_e \), which is corresponding to the amount of fuel saving if the \( N \) vehicles form a single platoon. For the schedule \( \pi \), suppose it leads to \( k \) single-vehicle platoons and \( r \) non-trivial platoons of size \( n_1, \ldots, n_r \), on the edge \( e \), respectively. The objective at \( e \) in this case is equal to
\[
\text{obj}(\pi, e) = r\sigma^f C_e + \sum_{i=1}^{r} (n_i - 1)\sigma^f C_e \\
= \sum_{i=1}^{r} n_i\sigma^f C_e - r(\sigma^f - \sigma^l) C_e \\
= (N-k)\sigma^f C_e - r(\sigma^l - \sigma^f) C_e,
\]
where we use the equality \( N = k + \sum_{i=1}^{r} n_i \). Since there are at most \( T \) time buckets, it implies that \( k + r \leq T \). Therefore, we get that
\[
\text{obj}(\pi, e) = N\sigma^f C_e - (k+r)\sigma^f C_e + r\sigma^l C_e \\
\geq N\sigma^f C_e - T\sigma^f C_e.
\]
Consequently, the ratio of interest can be bounded as

\[
\frac{\text{obj}(\pi, e)}{\text{obj}(\pi^*, e)} \geq \frac{N \sigma^f C_v - T \sigma^f C_v}{(N - 1) \sigma^f + \sigma^f} = \frac{(N - T) \sigma^f}{(N - 1) \sigma^f + \sigma^f} \geq 1 - \frac{T}{N},
\]

which concludes the proof. \(\square\)

**Corollary 2.** If an instance \((T, U, S)\) is inseparable, then for any vehicle \(u \in U\) there exists a vehicle \(v \in U \setminus \{u\}\) such that both \(R_u \cap R_v\) and \(S_u \cap S_v\) are non-empty.

Proof. If there exists a vehicle \(u\) such that for every vehicle \(v \in U \setminus \{u\}\) either \(R_u \cap R_v = \emptyset\) or \(S_u \cap S_v = \emptyset\), then the instance is separable with \(U_1 = \{u\}\) and \(U_2 = U \setminus \{u\}\). \(\square\)

**Lemma 5.** Suppose the regularity condition \((2)\) hold. Let \(T\) be a MST, \(U\) be the set of vehicles sharing \(T\), and \(S_v\) be the set of feasible time buckets for a vehicle \(v \in U\). Suppose \(|\mathcal{V}_v| \leq N_0\) hold for every \(e \in E(T)\), and the instance \((T, U, S)\) is inseparable. Then a vehicle-pairing algorithm applied on \(U\) can lead to an objective value \(\text{obj}(T)\) restricted on \(T\) satisfying

\[
\text{obj}(T) \geq \frac{(\sigma^f + \sigma^f) C_{\min}}{2L N_0 (L N_0 + 1)} |U|.
\]

(14)

Proof. Apply the following procedure to the instance \((T, U, S)\): (1) Let \(E' \leftarrow \emptyset, U' \leftarrow \emptyset\) and \(W \leftarrow \emptyset\). (2) Repeat the following step until it cannot be executed: Find two distinct vehicles \(v_1, v_2 \in U \setminus U'\) such that the sets \((R_{v_1} \cup R_{v_2}) \setminus E', R_{v_1} \cap R_{v_2}\) and \(S_{v_1} \cap S_{v_2}\) are all non-empty. Set \(U' \leftarrow U' \cup \{v_1, v_2\}\) and \(E' \leftarrow E' \cup R_{v_1} \cup R_{v_2}\). (3) Repeat the following step until it cannot be executed: Find two vehicles \(u \in U \setminus (U' \cup W)\) and \(v \in U'\) such that the sets \(R_u \setminus E', R_u \cap R_v\) and \(S_u \cap S_v\) are all non-empty. Set \(W \leftarrow W \cup \{u\}\) and \(E' \leftarrow E' \cup R_u \cup R_v\).

We first show that when Step (2) cannot be executed, the following property holds: For any edge \(e \in E(T) \setminus E'\) and any vehicle \(u \in \mathcal{V}_v \setminus U'\) there exists a vehicle \(v \in U'\) such that the sets \(R_u \cap R_v\) and \(S_u \cap S_v\) are all non-empty. We prove this claim by contradiction. Suppose there exist an edge \(e \in E(T) \setminus E'\) and a vehicle \(u \in \mathcal{V}_v \setminus U'\) which violate the statement. Since the instance is inseparable, there must exist a vehicle \(v \in U \setminus \{u\}\) such that the sets \(R_u \cap R_v\) and \(S_u \cap S_v\) are all non-empty. If \(v\) is not in \(U'\), the vehicle pair \((u, v)\) satisfies the executable condition at Step (2) by noting that \(e \in (R_u \cup R_v) \setminus E'\). This indicates that Step (2) can be further executed, leading to a contradiction. Therefore, we have \(v \in U'\). This proves the claim.

Next we show that when Step (3) cannot be executed, it holds that \(E' = E(T)\). We prove the claim by contradiction. Suppose there exists an edge \(e \in E(T) \setminus E'\). Since the instance is inseparable, there exist a vehicle \(u \in \mathcal{V}_v\) and a vehicle \(v \in U \setminus \{u\}\) such that the sets \(R_u \cap R_v\) and \(S_u \cap S_v\) are both non-empty. If \(v \notin U'\), vehicles \(u\) and \(v\) form a pair satisfying the conditions in Step (2), and hence that step can be further executed, which leads to a contradiction. Otherwise if \(v \in U'\), the two
vehicles form a pair satisfying the conditions in Step (3), which makes that step further executable, leading to a contradiction. Therefore, we have $E' = E(T)$ when the procedure terminates.

Notice that vehicles are added into the set $U'$ in pairs, and the two vehicles in a pair can be platooned on their shared edges. This indicates the following lower bound

$$\text{obj}(T) \geq \frac{|U'|}{2} \cdot (\sigma^l + \sigma^f)C_{\text{min}}.$$  \hspace{1cm} (EC.35)

Let $k = |U'|$ and $U' = \{v_1, \ldots, v_k\}$. When the procedure terminates, vehicles in $W$ can be partitioned into $k$ subsets denoted as $W_i$ for $i \in [k]$ such that every vehicle in $W_i$ can be (independently) platooned with $v_i$ on an edge from $R_{v_i}$. Since $|R_{v_i}| \leq L$ and every edge in $R_{v_i}$ is shared by at most $N_0$ vehicles. It follows that

$$|W| = \sum_{i=1}^{k} |W_i| \leq kLN_0 = LN_0|U'|,$$  \hspace{1cm} (EC.36)

and hence

$$|U| \leq N_0|E(T)| = N_0|E'| = N_0|\cup_{u \in U'} \cup_{W} R_u|$$

$$= (|U'| + |W|)LN_0 \leq (LN_0 + 1)LN_0|U'|.$$  \hspace{1cm} (EC.37)

Combining (EC.35) and (EC.37) gives the bound

$$\text{obj}(T) \geq \frac{(\sigma^l + \sigma^f)C_{\text{min}}}{2LN_0(LN_0 + 1)}|U'|,$$  \hspace{1cm} (EC.38)

which concludes the proof. \hspace{1cm} $\Box$

**Lemma 6.** Suppose the regularity condition (3) given at the beginning of Section 4.2 hold. When the condition at Line 15 of the Decomposition function in Algorithm 5 is satisfied, the cardinality of $E(T')$ must satisfy $N \leq |E(T')| < 3dN$.

Proof. When the condition $|E(T')| \geq N$ at Line 15 is satisfied, the edge $e$ at Line 13 satisfies either one of the two conditions: (a) $e$ is the first edge of the current path $P$; and (b) $e$ is not the first edge of $P$. We first prove for the case (a) by contradiction. Suppose $|E(T')| \geq 3dN$ In this case, and suppose the node $s_1$ has $k$ (in and out) edges that are not on $P$. Denote these edges as $e_1, \ldots, e_k$. By the regularity assumption, we should have $k \leq 2d$. The tree $T'$ in this case can be partitioned as $T' = \cup_{i=1}^{k} T_i$, where $T_i$ is the sub-tree of $T'$ containing $e_i$. Since $|E(T')| = \sum_{i=1}^{k} |E(T_i)|$ and $|E(T')| \geq 3dN$, there exists a $T_i$ satisfying $|E(T_i)| \geq N + 1$. Let $P_i$ be a maximal path of $T_i$ that contains $e_i$. Let $e_i'$ is the last edge of $P_i$ ($e_i' = e_i$ if $e$ is an in-edge of $s$). Notice that the algorithm has called Decomposition($P_i, T_i, N$) in Line 8. In Decomposition($P_i, T_i, N$), when the algorithm reached $e_i'$ as the last edge of $P_i$, the condition at Line 15 should satisfy since $|E(T_i')| \geq N$, where $T_i'$ is the sub-tree of $T_i$ that includes the starting node but excludes the end node of $e_i$. This indicates that the sub-tree $T_i'$ should have been cut from $T_i$, leading to a contradiction to the status of $T'$. 
We now consider the case (b). Since \( e \) is not the first edge of \( P \), it implies that the condition of Line 15 does not hold at any edge of \( P \) that comes before \( e \). Let \( e' \) be the edge right before \( e \), and suppose \( e' = (s_0, s_1) \) and \( e = (s_1, s_2) \). Let \( T'' \) be the maximal tree that includes the node \( s_0 \) but excludes the edge \( e' \). Since the condition of Line 15 does not hold at \( e' \), we should have \(|E(T'')| < N - 1\). For a similar argument on the sub-trees induced by edges of \( s_1 \) as in the case (a), we have
\[
|E(T')| \leq |E(T'')| + 1 + 2d(N + 1) < 3dN. \tag{EC.39}
\]
This concludes the proof. □

**Theorem 5.** Suppose Assumption 2 and regularity conditions (1)-(3) hold for a FRCVP instance \((\mathcal{T}, \mathcal{V}, \mathcal{S})\), where \( \mathcal{T} \) is a MST, \( \mathcal{V} \) is the set of all vehicles sharing \( \mathcal{T} \), and \( \mathcal{S} = \{ S_v : v \in \mathcal{V} \} \) is the collection of the feasible set of time buckets for each vehicle. Apply Algorithm 5 to the instance. If \( N_0 \) and \( N_1 \) in the algorithm are selected such that \( N_0 \geq (1 + 1/\epsilon)T \) and \( N_1 \geq 5L^4N_0^3p/\epsilon \), the objective returned by the algorithm satisfies \( \text{obj} \geq (1 - \epsilon)\text{OPT} \). The complexity of the algorithm is \(|E| \cdot T^{O(LdL^4T^4/\epsilon^5)}\), where \( E \) is the set of edges in \( \mathcal{T} \).

Proof. Let \( \mathcal{T}_{ctr} \) be the tree after running the \textsc{HeavyTrafficEdgeContraction} function on \( \mathcal{T} \). Let \( \mathcal{T}_1, \ldots, \mathcal{T}_K \) be the list of sub-trees returned by the \textsc{Decomposition} function. Since these sub-trees form a partition of \( \mathcal{T} \) in edges, there exist exact \( K - 1 \) pairs of sub-trees such that the two subtrees in the same pair should be adjacent (i.e., sharing a node) in \( \mathcal{T}_{ctr} \). Let \( \text{obj}^*(\mathcal{T}_k) \) be the objective value at the subtree \( \mathcal{T}_k \) achieved by a global optimal solution of the FRCVP instance, \( \text{obj}^\text{loc}(\mathcal{T}_k) \) be the optimal objective value of the instance \((\mathcal{T}_k, \mathcal{V}_k, \{ S_v : v \in \mathcal{V}_k \})\), where \( \mathcal{V}_k \) is the subset of vehicles of which the route intersects with \( \mathcal{T}_k \).

Let \( \text{obj}(\cdot) \) be the objective value achieved by Algorithm 5 at a given component. Note that for a given pair of adjacent sub-trees, there are at most \( N_0 \) vehicles with the route intersecting with both sub-trees in the pair, and these vehicles will lead to at most \( 2LN_0\sigma_f C_{\text{max}} \) fuel saving lost if removed from the vehicle system. We refer these vehicles as the sub-tree-crossing vehicles. It follows that
\[
\text{OPT} = \sum_{k=1}^{K} \text{obj}^*(\mathcal{T}_k) + \sum_{e \in \mathcal{T} \setminus \mathcal{T}_{ctr}} \text{obj}^*(e) \\
\leq \sum_{k=1}^{K} \text{obj}^\text{loc}(\mathcal{T}_k) + \frac{N_0}{N_0 - T} \sum_{e \in \mathcal{T} \setminus \mathcal{T}_{ctr}} \text{obj}(e) \\
\leq \sum_{k=1}^{K} \text{obj}(\mathcal{T}_k) + 2(K - 1)LN_0\sigma_f C_{\text{max}} + \frac{N_0}{N_0 - T} \sum_{e \in \mathcal{T} \setminus \mathcal{T}_{ctr}} \text{obj}(e) \tag{EC.40}
\]
where Lemma 4 is used to get the first inequality, and the term \( 2(K - 1)LN_0\sigma_f C_{\text{max}} \) is added in the second inequality to upper bound the total fuel saving made by sub-tree-crossing vehicles. Notice
that since $|E(T_k)| \geq N_1$ for all $k \in [K-1]$ by the design of the algorithm, and $T_K$ is the last (left-over) sub-tree added to the treeList with $|E(T_K)| < N_1$. We also note that for an inseparable sub-tree $T_k$, the number of vehicles sharing $T_k$ is correlated with the number of edges in $T_k$ in the sense that the following inequality holds: $|V(T_k)| \geq |E(T_k)|/L$. Combining this lower bound with the results from Lemma 5 leads to the following lower bound on $\text{obj}(T_k)$ for $k \in [K-1]$:

$$\text{obj}(T_k) \geq \frac{(\sigma^l + \sigma^f)C_{\min}}{2LN_0(LN_0 + 1)} |V(T_k)| \geq \frac{(\sigma^l + \sigma^f)C_{\min}N_1}{2L^2N_0(LN_0 + 1)}.$$

The above inequality further implies that

$$2LN_0\sigma^f C_{\max} \leq \frac{4L^3N_0^2(LN_0 + 1)}{N_1} \cdot \frac{\sigma^f}{\sigma^l + \sigma^f} \cdot \frac{C_{\max}}{C_{\min}} \text{obj}(T_k) \leq \frac{5L^4N_0^3\rho}{N_1} \text{obj}(T_k).$$

Substituting (EC.42) into (EC.40) gives

$$\text{OPT} \leq \sum_{k=1}^{K} \left(1 + \frac{5L^4N_0^3\rho}{N_1}\right) \text{obj}(T_k) + \frac{N_0}{N_0 - T} \sum_{e \in T \setminus C_{\text{tree}}} \text{obj}(e).$$

The conditions on $N_0$ and $N_1$ ensure that $N_0/(N_0 - T) \leq 1 + \epsilon$ and $5L^4N_0^3\rho/N_1 \leq \epsilon$, which leads to the inequality $\text{OPT} \leq (1 + \epsilon) \text{obj}$ and hence $\text{obj} \geq (1 - \epsilon) \text{OPT}$.

We analyze the computational complexity of the algorithm. First, the number of sub-trees in the treeList is at most $|E|/N_1$ (up to rounding), where $E$ is the set of edges in $T$. Each sub-tree has at most $3dN_1$ edges (Lemma 6), and the number of vehicles involved in each sub-tree is at most $3dN_1N_0$. The complexity of enumerating all feasible solution for each sub-tree is upper bounded by $T^{3dN_1N_0}$. Notice that the conditions satisfied by $N_0$ and $N_1$ imply that $N_1 \geq 5\rho L^4T^3(1 + 1/\epsilon)^3/\epsilon$. Therefore, the complexity of the algorithm is estimated as

$$\frac{|E|}{N_1} \cdot T^{3dN_1N_0} \leq \frac{\epsilon |E|}{5 \rho L^4T^3(1 + 1/\epsilon)^3} \left[\exp \left(15\rho d L^4 \frac{1}{\epsilon} \left(1 + \frac{1}{\epsilon}\right)^3\right)\right] \log T
\sim |E| \cdot T^{O(\rho d L^4T^3/\epsilon^3)},$$

which concludes the proof. $\square$

**Theorem 6.** Suppose the Assumption 2 and the regularity condition (1) hold. Let $V$ be the set of vehicles and $R$ be the set of edges in the MST shared by vehicles in $V$. Let $\text{obj}(x^*)$ be the objective value of the schedule obtained from the randomized algorithm, and $\text{OPT}$ be the optimal value of the instance. If the ratio $\frac{|V|}{|R|} \rightarrow \infty$, then $\frac{\mathbb{E}[\text{obj}(x^*)]}{\text{OPT}} \rightarrow 1$.

Proof. Let $[x^*, y^*, y^*, w^*]$ be an optimal solution of (LP). The optimal objective of the linear program is expressed as

$$\widehat{\text{OPT}} = \sum_{e \in R} \sum_{t \in S} (\sigma^l C_e y^*_e + \sigma^f C_e w^*_e).$$

(EC.45)
Since $\mathbb{E}[\text{obj}(x^*)] \leq \text{OPT} \leq \frac{\mathbb{E}[\text{obj}(x^*)]}{\text{OPT}}$, it suffices to show that $\frac{\mathbb{E}[\text{obj}(x^*)]}{\text{OPT}} \to 1$ as $\frac{|V|}{|R|} \to \infty$. The randomized algorithm returns a schedule in which vehicles that are assigned to a same time bucket can form a single platoon on edges that they share. Let $Y'_{e,t}$ be the indicator random variable of the event that at least two vehicles sharing the edge $e$ are assigned to the time bucket $t$. Let $W_{e,t}$ be the random variable representing the quantity $(Q_{e,t} - 1)^+$, where $Q_{e,t}$ is the number of vehicles sharing $e$ that are assigned to $t$ by the randomized algorithm, and $(\cdot)^+ := \max\{0, \cdot\}$. The expectation of $\text{obj}(x^*)$ is written as

$$
\mathbb{E}[\text{obj}(x^*)] = \sum_{e \in R} \sum_{t \in S} (\sigma^f C_e \mathbb{E}[Y'_{e,t}] + \sigma^f C_e \mathbb{E}[W_{e,t}]).
$$

(EC.46)

We can get a lower bound for $\mathbb{E}[Y'_{e,t}]$ as follows:

$$
\mathbb{E}[Y'_{e,t}] = \mathbb{P}(Y_{e,t} = 1) \geq \max_{(u,v) \in V_{e,t}} x_{u,t}^{*,v,v_t}^{*}
\geq \max_{(u,v) \in V_{e,t}} \left[ x_{u,t}^{*} x_{v,t}^{*} - (1 - x_{u,t}^{*}) (1 - x_{v,t}^{*}) \right]
= \max_{(u,v) \in V_{e,t}} (x_{u,t}^{*} x_{v,t}^{*} - 1).
$$

(EC.47)

From the constraint (LP.3) if $\max_{(u,v) \in V_{e,t}} (x_{u,t}^{*} + x_{v,t}^{*} - 1) \geq 0$, we have $y_{e,t}^{*} = \max_{(u,v) \in V_{e,t}} (x_{u,t}^{*} + x_{v,t}^{*} - 1)$, otherwise $y_{e,t}^{*} = 0$. For both cases, we have shown that $\mathbb{E}[Y'_{e,t}] \geq y_{e,t}^{*}$. We also have the following lower bound for $\mathbb{E}[W_{e,t}]$ by definition:

$$
\mathbb{E}[W_{e,t}] \geq \mathbb{E} \left[ \sum_{u \in V_{e,t}} X_{u,t} - 1 \right] = \sum_{u \in V_{e,t}} x_{u,t}^{*} - 1.
$$

(EC.48)

Let $C$ be the following subset of $(e, t)$ pairs

$$
C = \left\{ (e,t) : \sum_{u \in V_{e,t}} x_{u,t}^{*} \geq 1 \right\},
$$

and $C'$ be the complement of $C$. Notice that the following property holds

$$
\sum_{(e,t) \in C \cup C'} \sum_{u \in V_{e,t}} x_{u,t}^{*} = \sum_{e \in R} \sum_{t \in S} \sum_{u \in V_{e,t}} x_{u,t}^{*} = \sum_{e \in R} \sum_{u \in V_{t} \in S_u} \sum_{t \in S} \sum_{u \in V_{e,t}} x_{u,t}^{*} = \sum_{e \in R} |V_e| \geq |V|,
$$

(EC.49)

where the constraint (LP.2) has been used to get the third equality above. The term $\sum_{e \in R} \sum_{t \in S} \sigma^f C_e x_{e,t}^{*}$ in the objective of (LP) can be split into two parts: $\sum_{(e,t) \in C \cup C'} \sigma^f C_e x_{e,t}^{*}$ and $\sum_{(e,t) \in C'} \sigma^f C_e x_{e,t}^{*}$, which will be bounded separately using the constraint (LP.4). For the first part, we have

$$
\sum_{(e,t) \in C} \sigma^f C_e x_{e,t}^{*} \leq \sum_{(e,t) \in C} \sigma^f C_e x_{u,t}^{*},
$$

(EC.50)

and for the second part we can use the definition of $C'$ to obtain the following bound

$$
\sum_{(e,t) \in C'} \sigma^f C_e x_{e,t}^{*} \leq \sigma^f C_{\text{max}} \sum_{(e,t) \in C'} \sum_{u \in V_{e,t}} x_{u,t}^{*} \leq \sigma^f C_{\text{max}} |C'|.
$$

(EC.51)
On the other side, we get a lower bound of $E[\text{obj}(x^*)]$ using the following inequalities

\[
E[\text{obj}(x^*)] \geq \sum_{(e,t) \in C \cup C'} \sigma^e C_e E[Y_{e,t}^*] + \sum_{(e,t) \in C} \sigma^f C_e E[W_{e,t}]
\]

\[
\geq \sum_{(e,t) \in C \cup C'} \sigma^e C_e y_{e,t}^* + \sum_{(e,t) \in C} \sigma^f C_e \left( \sum_{u \in V_{e,t}} x_{u,t}^* - 1 \right)
\]

\[
\geq \sum_{(e,t) \in C \cup C'} \sigma^e C_e y_{e,t}^* + \sum_{(e,t) \in C} \sum_{u \in V_{e,t}} \sigma^f C_e x_{u,t}^* - \sigma^f \max|C|.
\] (EC.52)

Using the lower bound (EC.52) for $E[\text{obj}(x^*)]$, and substituting (EC.50) and (EC.51) into (EC.45) to get an upper bound for $\hat{\text{OPT}}$, we deduce the following lower bound on the ratio

\[
\frac{E[\text{obj}(x^*)]}{\hat{\text{OPT}}} \geq \frac{\sum_{(e,t) \in C \cup C'} \sigma^e C_e y_{e,t}^* + \sum_{(e,t) \in C} \sum_{u \in V_{e,t}} \sigma^f C_e x_{u,t}^* - \sigma^f \max|C|}{\sum_{(e,t) \in C \cup C'} \sigma^e C_e y_{e,t}^* + \sum_{(e,t) \in C} \sum_{u \in V_{e,t}} \sigma^f C_e x_{u,t}^* + \sigma^f \max|C'|}.
\] (EC.53)

Notice that the term $\sum_{(e,t) \in C} \sum_{u \in V_{e,t}} \sigma^f C_e x_{u,t}^*$ presented in both the numerator and the denominator has the following lower bound

\[
\sum_{(e,t) \in C} \sum_{u \in V_{e,t}} \sigma^f C_e x_{u,t}^* \geq \sigma^f \min \sum_{(e,t) \in C} \sum_{u \in V_{e,t}} x_{u,t}^*
\]

\[
= \sigma^f \min \sum_{(e,t) \in C} \sum_{u \in V_{e,t}} x_{u,t}^* - \sigma^f \min \sum_{(e,t) \in C} \sum_{u \in V_{e,t}} x_{u,t}^*
\]

\[
\geq \sigma^f \min (|C| - |C'|)
\] (EC.54)

where the constraint (LP.3) has been used to get the last equality. Since $|C| + |C'| \leq |R| T$, and $|R| \to \infty$, combining (EC.53) and (EC.54) shows that $\frac{E[\text{obj}(x^*)]}{\hat{\text{OPT}}} \to 1$, which concludes the proof. \(\blacksquare\)

**Proposition 5.** Let $G$ be the group generated by the elements $\{0\} \cup \{a_u, b_u : u \in V_{ext}\}$ under a finite number of plus and minus operations, where $a_u = \text{RTW}_u$ and $b_u = \text{RTW}_u$ are the coordinates of the lower and upper bounds of RTW. If $G$ is isomorphic to a one-dimensional lattice group, running Algorithm 6 on this instance will terminate in a finite number of iterations, and hence the corresponding general FRCVP instance admits a valid vehicle-to-time-bucket assignment formulation given in (GVA).

Proof. Suppose $p$ is a break point generated on a vehicle $u \in V_{ext}$ at iteration $n$ in the while loop (Line 18) of Algorithm 6. In the iteration $n + 1$, the point $p$ will either be projected in the vertical or the shifting mode to other vehicles. If the projection is in the vertical mode, the coordinate of the newly generated break points by $p$ should have the same coordinate as $p$. If the projection is in the shifting mode, consider the case of projecting $p$ on a vehicle $u'$ which is another virtual copy satisfying $u, u' \in \mathcal{M}(v)$ for some $v \in V_{ext}$. Then the projected point on RTW should have the coordinate equal to $a_{u'} + p - a_u$, where $p$ in this expression represents the coordinate of the point...
p. By the definition of the group $G$, it is easy to see that if $p \in G$, then the projected points of $p$ in the next iteration should also in $G$ under the two projection modes. Since at the beginning of the while loop in the algorithm, all break points are simply the end points of each vehicle, and hence in $G$, all break points generated from the initial set in later iterations form a subset of $G$.

Given that $G$ is isomorphic to a one dimensional lattice group, it can then be represented as $G = \{kc \mid k \in \mathbb{Z}\}$ for some constant $c \in \mathbb{R}$. Also notice that all the break points should be generated within the bounded region $\Lambda = [\min_u a_u, \max_u b_u]$ by the algorithm. Since there can only a finite number of elements from $G$ that are in $\Lambda$ (i.e., upper bounded by $\lceil \Lambda/c \rceil + 1$), the set of all break points generated by the algorithm must be finite, which implies that the algorithm should terminate in a finite number of iterations. The time buckets generated by the algorithm lead to a valid vehicle-to-time-bucket assignment formulation (GVA). 
\[\square\]