

Route assignment and scheduling with trajectory coordination

Navid Matin-Moghaddam & Jorge A. Sefair

To cite this article: Navid Matin-Moghaddam & Jorge A. Sefair (2021) Route assignment and scheduling with trajectory coordination, IISE Transactions, 53:2, 164-181, DOI: 10.1080/24725854.2020.1774096

To link to this article: <https://doi.org/10.1080/24725854.2020.1774096>



[View supplementary material](#)



Published online: 14 Jul 2020.



Submit your article to this journal



Article views: 274



[View related articles](#)



[View Crossmark data](#)



Citing articles: 1 View citing articles



Route assignment and scheduling with trajectory coordination

Navid Matin-Moghaddam and Jorge A. Sefair 

School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ, USA

ABSTRACT

We study the problem of finding optimal routes and schedules for multiple vehicles traveling in a network. Vehicles may have different origins and destinations, and must coordinate their trajectories to keep a minimum distance from each other at any time. We determine a route and a schedule for each vehicle, which possibly requires vehicles to wait at some nodes. Vehicles are heterogeneous in terms of their speed on each arc, which we assume is known and constant once in motion. Applications of this problem include air and maritime routing, where vehicles maintain a steady cruising speed as well as a safety distance to avoid collision. Additional related problems arise in the transportation of hazardous materials and in military operations, where vehicles cannot be too close to each other given the risk posed to the population or the mission in case of a malicious attack. We discuss the hardness of this problem and present an exact formulation for its solution. We devise an exact solution algorithm based on a network decomposition that exploits the sparsity of the optimal solution. We illustrate the performance of our methods on real and randomly generated networks.

ARTICLE HISTORY

Received 10 August 2019
Accepted 19 May 2020

KEY WORDS

Trajectory planning; route assignment; vehicle routing and scheduling; network optimization

1. Introduction

This article studies a route assignment and scheduling problem in which vehicles need to keep a minimum distance from each other at any time. This problem, which we refer to as RASTC (Route Assignment and Scheduling with Trajectory Coordination) is inspired by modern applications in transportation and logistics, and particularly by the emerging challenge of coordinating driverless vehicles. In RASTC, a set of vehicles travel between known origins and destinations in a directed network. Due to safety reasons (e.g., to avoid collisions or to reduce the vulnerability to adversary attacks), the Euclidean distance between any two vehicles cannot be less than a given parameter at any time. To avoid such *geographic conflict*, RASTC seeks a route and a schedule for each vehicle, specifying the departure time from each node along the route that also minimizes a function of the vehicles' travel times. RASTC includes other realistic features such as heterogeneous vehicles in terms of speed and minimum and maximum waiting times at any node.

Route assignment and scheduling problems are common in transportation and logistics applications. Related problems such as vehicle routing with time windows (Bräysy and Gendreau, 2005a, 2005b), time-dependent vehicle routing (Malandraki and Daskin, 1992), and dial-a-ride (Cordeau and Laporte, 2007), are close to RASTC, as they decide on the vehicles' departure times and seek for optimal routes and schedules. However, they focus on time-dependent demands or dynamic travel times rather than enforcing a minimum distance between vehicles (see, e.g., Pillac *et al.* (2013) and Dixit *et al.* (2019) for

comprehensive reviews on dynamic routing problems). Multiple decentralized routing models have focused on obstacle and collision avoidance. In a single-vehicle application, Hu *et al.* (2018) propose a model that selects the best path from a set of candidates to avoid static and moving obstacles while choosing the vehicle's speed and acceleration. For multiple vehicles, Jin *et al.* (2012), Kamal *et al.* (2015), and Zhu and Ukkusuri (2015) propose models and algorithms for intersection control that determine routes and departure schedules to minimize the total travel time of all vehicles across the intersection. Rios-Torres and Malikopoulos (2017) present a comprehensive survey on vehicle coordination approaches for intersections and highway on-ramps. Under the assumption of discrete time, Yu and LaValle (2012) and Ferrati and Pallottino (2013) use time-expanded networks for centralized vehicle coordination in order to construct collision-free trajectories. Yamashita *et al.* (2005) propose coordination policies to dynamically adjust vehicle routes while en route, aiming to reduce congestion in connected environments where vehicles share location and destination data.

Collision avoidance is relevant in other applications beyond road traffic. In flexible manufacturing systems, Automated Guided Vehicles (AGVs) are used for material handling and need to be safely routed across the production facility. In this area, Nishi *et al.* (2011) propose a decomposition algorithm to optimally route and schedule AGVs in discrete time that synchronizes vehicles and production schedules. The manufacturing layout is modeled as a network, thus route conflicts prevent two or more AGVs from using a node or edge at the same time. This definition of

conflict is also used by Krishnamurthy *et al.* (1993) to optimize AGV routes for known demands and by Adamo *et al.* (2018) to optimize the speed of AGVs for pickup and delivery operations with time windows. Corr  a *et al.* (2004) combine constraint programming and mixed-integer programming over a space-time network for dispatching and conflict-free routing of AGVs. Fazlollahtabar and Saidi-Mehrabad (2015) present a survey on existing methodologies for AGV scheduling and routing. Collision avoidance is also relevant in air traffic control of airplanes or Unmanned Aerial Vehicles (UAVs). Using a heuristic approach, Phung *et al.* (2017) solve a path planning problem for a single UAV in the context of infrastructure inspection with static obstacles. In a real-time setting, Frazzoli *et al.* (2001) propose a protocol for airplane conflict resolution, in which a centralized traffic controller adjusts aircraft trajectories to minimize the deviation from ideal routes sent by each pilot. Richards and How (2002) investigate the problem of finding optimal trajectories for multiple aircrafts to avoid collisions. The proposed discrete-time problem includes aircraft turning rates and speed decisions, as well as collision avoidance constraints, which are embedded into a mixed-integer program. Otto *et al.* (2018) provide a survey on optimization approaches for UAV routing. Trajectory coordination is also relevant in the area of multi-robot path planning (Hoy *et al.*, 2015). Given known origins, destinations, and fixed paths for a set of robots, Abichandani *et al.* (2013) propose a nonlinear optimization problem to determine velocity profiles under collision, kinematics, and communication constraints. Ferrera *et al.* (2013), and Ferrera *et al.* (2014) provide decentralized approaches for collision avoidance via robot coordination.

Routing and scheduling with a minimum distance requirement is also in the transportation of hazardous materials, where spills or explosions are uncommon, but have serious consequences to the environment and humans (Erkut and Verter, 1998). A common approach to mitigate the impact of hazmat accidents is to design routes that satisfy safety, equity, and operational criteria (List *et al.*, 1991; Current and Ratick, 1995). Gopalan, Batta, and Karwan (1990) and Gopalan, Kalluri, Batta, and Karwan (1990) design vehicle routes under equity considerations that balance the population exposure to hazmat shipments along the path, as it is undesirable to expose the same population to the risk of multiple hazmat shipments at the same time. Toumazis and Kwon (2016) extend the conditional value at risk ideas to develop a risk metric for the design of robust routes that minimize the worst-case consequences of a potential accident. Esfandeh *et al.* (2018) study a network design problem that includes time-dependent road closures that indirectly influence the routes chosen by hazmat vehicles, which helps to reduce a population, exposure to the risky shipments in space and time. In a closely related study to RASTC, Carotenuto *et al.* (2007) argue that if two hazmat vehicles travel too close to each other and one is involved in an accident, there is a high probability that the other will also be affected. As a result, they enforce a minimum distance between vehicles with a two-stage heuristic

approach that first identifies a candidate set of low-risk routes and then determines vehicle departure times while considering that vehicles cannot wait at intermediate nodes. To mitigate the additional risk posed by vehicles traveling too close to each other, each route is discretized and no two vehicles are allowed inside the safety (circular) area around any of the discrete points at the same time. In case of accident or malicious attack affecting a vehicle, this separation gives nearby vehicles enough time to react, limiting the negative consequences on the population. In the military context, the convoy movement problem seeks conflict-free trajectories to move military assets from sources to destinations, while satisfying spatiotemporal constraints. In this area, Thomas *et al.* (2015) propose an algorithm to route convoys across a network, assuming that convoys move as a whole and occupy an edge for some time given their length. For security reasons, a minimum inter-convoy distance is maintained for convoys traveling in the same direction. For the same problem, Chardaire *et al.* (2005) propose a discrete-time integer programming formulation that selects the best route out of a candidate set and a starting time for each convoy, assuming that once a convoy starts traversing the network it continuously moves until reaching its destination. Using an integer programming approach, Kumar and Narendran (2008) determine each convoy's route and departure time while enforcing a minimum inter-convoy headway on shared network components.

The concept of geographic conflict also arises in other *static* problems that focus on finding disjoint paths, but ignore the flow scheduling aspect. A classic problem in this area is the design of survivable networks, which aims to find a minimum cost network such that each pair (or subset) of nodes is connected by a given number of arc- or node-disjoint paths (Suurballe, 1974; Grotschel *et al.*, 1995; Kerivin and Mahjoub, 2005; Omran *et al.* 2013, Diarrassouba *et al.*, 2018). Margolis *et al.* (2018) use this concept to design a resilient supply network by selecting multiple node-disjoint distribution channels, which mitigates the risk of unsatisfied demand due to disruptions at intermediate stages. Other variants in communication problems aim to find disjoint paths that guarantee some degree of information transmission. They include finding arc-disjoint paths in the presence of resilient arcs (i.e., not subject to failure) (Żotkiewicz *et al.*, 2010) and bifurcated routing problems between an origin and a destination with node and arc use costs (De Jongh *et al.*, 1999). Other approaches focus on the design of disjoint paths under spatial failures. These failures are modeled as disks of known diameter that can be located anywhere in the continuous space. It is assumed that components overlapping with a disk completely fail (or are destroyed) (Neumayer *et al.*, 2009; Neumayer *et al.*, 2015) or lose some functionality (Sullivan and Smith, 2014). Extending the max-flow min-cut theorem, Neumayer *et al.* (2009), Kobayashi and Otsuki (2014), Neumayer *et al.* (2015), and Otsuki *et al.* (2016) study the problems of finding the maximum number of geographically disjoint paths between two nodes and the minimum number of disk failures to disconnect two nodes. In these problems, two paths

are geographically disjoint if the minimum distance between them is at least a given value (except in areas close to the origin and destination). Although close to RASTC, these *static* approaches ignore the vehicle route assignment and scheduling aspects, and can only be used when all vehicles travel between the same origin and destination. We refer to these approaches as static, as they enforce a minimum distance between paths rather than vehicles.

Applications of RASTC include air and maritime routing, where vehicles need to maintain a steady cruising speed, as well as a safety distance, to avoid collision. Additional related problems arise in the transportation of hazardous materials, where vehicles cannot be too close to each other given the risk posed to the population in case of accident or malicious attack. Moreover, RASTC is related to the convoy movement problem in the absence of congestion when the convoy can split at any point and vehicles can travel at the speed limit (or any other constant speed) along different routes. In this case, enforcing a minimum separation distance decreases the risk of losing multiple vehicles at once given an airstrike or a roadside bomb. RASTC is also related to AGV routing when the interest is to find routes and schedules for known demands and when operations require each AGV to travel between two locations only.

We focus on problems with a moderate number of vehicles and assume that cycles are undesirable, as they may unnecessarily expose a population to dangerous materials, require additional fuel consumption, and may not be even possible given the network used. We allow heterogeneous vehicles with different speeds, which remain constant once in motion. Each vehicle has a designated speed for each arc, which is not necessarily the same for all arcs. This is plausible for air and maritime routing applications, where vehicles travel at cruising speeds. For more ambitious related problems such as routing of autonomous vehicles, the solution from RASTC provides a benchmark to compare the performance of other models or solution approaches. For instance, the solution quality of a heuristic that allows variable speeds must be no worse than that provided by RASTC. Moreover, RASTC provides a benchmark for any decentralized coordination mechanism, as it assumes a centralized (and optimal) coordination between the vehicles. Although RASTC is not specifically designed to handle variable speeds while in motion, these can be approximated using our approach. Because the goal is to avoid the geographic conflict while still moving towards the destination, it is possible to add extra nodes along an arc where vehicles can wait. In this way, the combination of motion (at constant speed) and waiting can approximate a deceleration profile.

RASTC has received very little attention in the literature. There are no exact models or algorithms available to tackle emerging RASTC problems in continuous time. The following are our main contributions:

1. We introduce an NP-hard routing and scheduling problem that is relevant for current and emerging related applications and that encompasses several realistic features.

2. We develop an alternative linear formulation to the geographic conflict that avoids the Euclidean norm when calculating the distance between vehicles and propose a polynomial-time pre-processing approach to characterize the conflict.
3. We embed such conflict constraints into a Mixed-Integer Programming (MIP) formulation, which we solve using a tailored decomposition technique.

This article is organized as follows. In [Section 2](#), we study the geographic conflict and derive disjunctive linear constraints on the vehicles' departure times to represent the geographic conflict. In [Section 3](#), we embed the developments from [Section 2](#) into an MIP to solve RASTC. Although exact, the MIP's number of variables and constraints makes it computationally challenging for solving moderate- and large-scale instances. To overcome this problem, in [Section 4](#) we devise an exact solution approach based on a network decomposition. In [Section 5](#), we demonstrate the performance of our approach and explore its limits by solving real instances out of Berlin's road network, as well as other randomly generated instances. [Section 6](#) presents our final remarks and future research. [Appendices 1 and 2](#) contain all the proofs of our propositions.

2. Vehicle coordination modeling

We enforce the coordination among a set of vehicles, V , by imposing constraints on their departure times from the nodes they traverse in a directed network, $G = (N, A)$. In [Section 2.1](#), we characterize the distance between network elements, and in [Section 2.2](#) we describe the conditions that vehicles' departure times must satisfy to avoid geographic conflict. We define $\check{\delta}((i, j), k)$ and $\hat{\delta}((i, j), (k, l))$ as functions returning the shortest Euclidean distance between arc (i, j) and node k , and between arcs (i, j) and (k, l) , respectively. We assume that arcs are straight lines. Non-straight trajectories (e.g., along a winding road) can be approximated by adding intermediate nodes connected by straight arcs.

2.1. Distance between network elements

2.1.1. Arc-node distance

The arc-node distance is the minimum Euclidean distance between arc (i, j) and node k , which we denote by $\check{\delta}((i, j), k)$. To calculate $\check{\delta}((i, j), k)$, we use the coordinates of node i , $\mathbf{x}_i$, and the unit vector in the direction of arc (i, j) , $\mathbf{u}$. We then obtain the orthogonal projection of $\mathbf{x}_k$ on the line $\mathbf{x}_i + \alpha\mathbf{u}$, which is given by $\mathbf{x}_i + \alpha^*\mathbf{u}$, where $\alpha^* = \arg \min_{\alpha \in \mathbb{R}} \|\mathbf{x}_i + \alpha\mathbf{u} - \mathbf{x}_k\|$. The value of α^* is unique and can be calculated in closed form given the convexity of the Euclidean norm function. Using this projection, we have that

$$\check{\delta}((i, j), k) = \begin{cases} \|\mathbf{x}_i - \mathbf{x}_k\| & \text{If } \alpha^* \leq 0 \\ \|\mathbf{x}_j - \mathbf{x}_k\| & \text{If } \alpha^* \geq \|\mathbf{x}_j - \mathbf{x}_i\| \\ \|\mathbf{x}_i + \alpha^*\mathbf{u} - \mathbf{x}_k\| & \text{If } 0 < \alpha^* < \|\mathbf{x}_j - \mathbf{x}_i\|. \end{cases}$$

2.1.2. Arc-arc distance

We define the arc-arc distance as the minimum Euclidean distance between arcs (i, j) and (k, l) , which we denote by $\hat{\delta}((i, j), (k, l))$. We also define $\mathbf{u}_{ij}$ and $\mathbf{u}_{kl}$ as the unit vectors in the direction of arcs (i, j) and (k, l) , respectively. Following the same intuition as in the arc-node distance, we find two points along the lines $\mathbf{x}_i + \alpha_{ij}\mathbf{u}_{ij}$ and $\mathbf{x}_k + \alpha_{kl}\mathbf{u}_{kl}$ whose distance is minimum by finding $\alpha^* = \arg \min_{(\alpha_{ij}, \alpha_{kl}) \in \mathbb{R}^2} \|\mathbf{x}_i + \alpha_{ij}\mathbf{u}_{ij} - \mathbf{x}_k - \alpha_{kl}\mathbf{u}_{kl}\|$. Using $\alpha^* = (\alpha_{ij}^*, \alpha_{kl}^*)$, we have that

$$\hat{\delta}((i, j), (k, l)) = \begin{cases} \|\mathbf{x}_i + \alpha_{ij}^*\mathbf{u}_{ij} - \mathbf{x}_k - \alpha_{kl}^*\mathbf{u}_{kl}\|, & \text{If } 0 \leq \alpha_{ij}^* \leq \|\mathbf{x}_j - \mathbf{x}_i\| \text{ and } 0 \leq \alpha_{kl}^* \leq \|\mathbf{x}_l - \mathbf{x}_k\| \\ \min\{\check{\delta}((i, j), k), \check{\delta}((i, j), l), \check{\delta}((k, l), i), \check{\delta}((k, l), j)\}, & \text{Otherwise.} \end{cases}$$

Although in this case vector α^* may not be unique (e.g., when arcs are parallel), an optimal solution α^* can still be obtained in closed form given convexity of the Euclidean norm function.

2.2. Geographic conflict modeling

This section describes our approach to prevent geographic conflict by requiring vehicles to maintain a distance of at least d units at any time. We introduce two types of conflicts: *arc-arc*, used to avoid conflict when vehicles are moving, and *arc-node*, used when one of the vehicles is waiting at a node and the other is moving. To impose such conflict constraints only on relevant network components, we define $\Omega = \{((i, j), (k, l)) \subset A \times A : \hat{\delta}((i, j), (k, l)) < d\}$ and $\Psi = \{((i, j), k) \subset A \times N : \check{\delta}((i, j), k) < d\}$. As an arc-arc conflict is symmetric, we add $((i, j), (k, l))$ or $((k, l), (i, j))$ to Ω , but not both. We assume that there is no arc-node conflict involving any vehicle's origin or destination nodes.

2.2.1. Arc-arc conflict

Suppose that while traversing the network, vehicles g and h use arcs (i, j) and (k, l) , respectively, with corresponding velocity vectors $\mathbf{v}_{ij}^g$ and $\mathbf{v}_{kl}^h$, where $((i, j), (k, l)) \in \Omega$. Under these conditions, the distance between g and h at time t is given by $\|\mathbf{x}_i + \mathbf{v}_{ij}^g t - (\mathbf{x}_k + \mathbf{v}_{kl}^h(t - \delta))\|$, where δ represents the difference in the departure time of h with respect to the departure time of g , which is the reference time. For instance, $\delta < 0$ indicates that h departs δ units of time before g , whereas $\delta > 0$ indicates that h departs δ units of time after g . If $\delta = 0$, then g and h start moving at the same time. We assume that each vehicle's speed is constant while traversing an arc, thus g spends $c_{ij}^g = \frac{\|\mathbf{x}_j - \mathbf{x}_i\|}{\|\mathbf{v}_{ij}^g\|}$ time moving from i to j , and h spends $c_{kl}^h = \frac{\|\mathbf{x}_l - \mathbf{x}_k\|}{\|\mathbf{v}_{kl}^h\|}$ time moving from k to l . This means that any value of δ and t satisfying the following two conditions leads to a geographic conflict between g and h while they are in motion along arcs (i, j) and (k, l) , respectively:

$$\|\mathbf{x}_i + \mathbf{v}_{ij}^g t - (\mathbf{x}_k + \mathbf{v}_{kl}^h(t - \delta))\| < d \quad (1)$$

$$t \in [\max\{0, \delta\}, \min\{c_{ij}^g, c_{kl}^h + \delta\}] \quad (2)$$

Condition (1) reflects the occurrence of geographic conflict, whereas (2) narrows the time window for conflict analysis down to values of t when vehicles are moving. The term $\max\{0, \delta\}$ captures the earliest time at which both vehicles are in motion and the term $\min\{c_{ij}^g, c_{kl}^h + \delta\}$ captures the

time at which the first vehicle arrives at its destination. The following proposition establishes an important property of all the values (δ, t) satisfying (1) and (2), which we later use to derive some constraints for our mathematical program.

Proposition 1. *If the differences in departure times δ_1 and δ_2 , where $\delta_1 \leq \delta_2$, lead to a geographic conflict, then any $\delta \in [\delta_1, \delta_2]$ also leads to a conflict.*

Proposition 1 proves the convexity of the set of values (δ, t) satisfying (1) and (2), and suggests that the geographic conflict interval can be fully characterized by the minimum and maximum possible difference in the departure times. Based on this observation, we define $\hat{\ell}_{ijkl}^{gh} = \min\{\delta | (\delta, t) \text{ satisfies } \|\mathbf{x}_i + \mathbf{v}_{ij}^g t - (\mathbf{x}_k + \mathbf{v}_{kl}^h(t - \delta))\| \leq d \text{ and (2)}\}$ and $\hat{u}_{ijkl}^{gh} = \max\{\delta | (\delta, t) \text{ satisfies } \|\mathbf{x}_i + \mathbf{v}_{ij}^g t - (\mathbf{x}_k + \mathbf{v}_{kl}^h(t - \delta))\| \leq d \text{ and (2)}\}$, thus conflict arises if the difference between the departure times falls in the interval $(\hat{\ell}_{ijkl}^{gh}, \hat{u}_{ijkl}^{gh})$. That is, if we let τ_i^g and τ_k^h be the departure times of vehicles g and h from nodes i and k along arcs (i, j) and (k, l) , respectively, such that $((i, j), (k, l)) \in \Omega$, then the disjunction $(\tau_k^h - \tau_i^g \leq \hat{\ell}_{ijkl}^{gh}) \vee (\tau_k^h - \tau_i^g \geq \hat{u}_{ijkl}^{gh})$ avoids any conflict when the vehicles are in motion along these arcs. The optimization problems producing $\hat{\ell}$ - and $\hat{u}$ -parameters are convex due to Proposition 1 and that Slater's constraint qualification holds given that $((i, j), (k, l)) \in \Omega$. Hence, we can obtain the unique optimal values for $\hat{\ell}$ and $\hat{u}$ -parameters in closed-form by solving the corresponding system of first-order Karush-Kuhn-Tucker optimality conditions.

To illustrate the operation of our arc-arc conflict conditions, consider the situation depicted in Figure 1(a), where vehicle g travels between nodes i and j using arc (i, j) , and vehicle h travels from k to l using arc (k, l) . The xy -coordinates of nodes i, j, k , and l are $(0, 0)$, $(2, 2)$, $(3, 3)$, and $(2, 1)$, respectively. In this case, we assume $d=1$, for which $((i, j), (k, l))$ clearly belongs to Ω . Assuming speeds equal to one, we obtain $\hat{\ell}_{ijkl}^{gh} = -0.82$, which means that h must depart k at least 0.82 units of time before g to avoid conflict. Figure 1(b) illustrates the situation when $\tau_k^h - \tau_i^g = -0.82$.

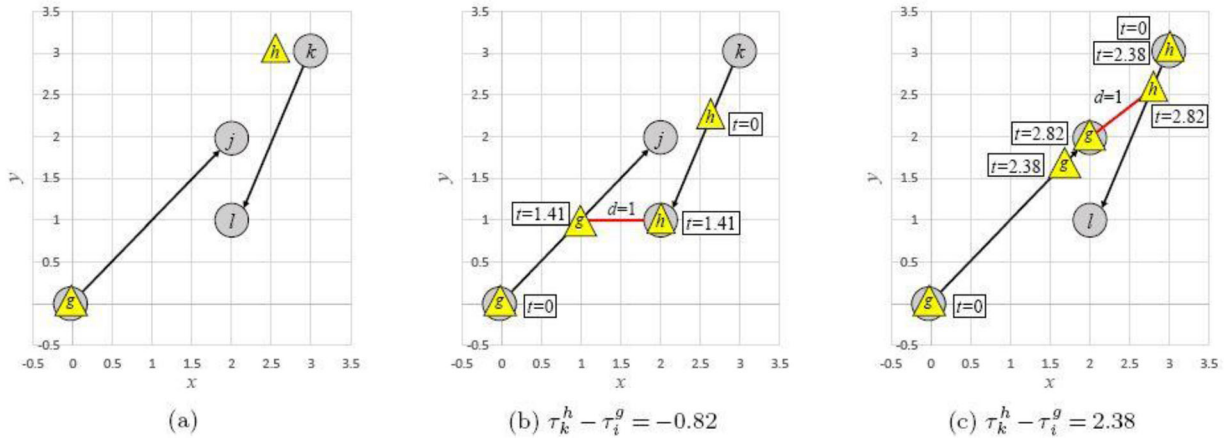


Figure 1. Arc-arc conflict analysis.

At the departure time of g (which we denote by $t=0$ for convenience), vehicle h is almost halfway of its trip to l . At $t=1.41$, g is halfway to j and h arrives at l , where the distance between vehicles is exactly d . In this case, there is no conflict between g and h , as the distance between them is no less than d at any time while in motion. Note that any departure times such that $\tau_k^h - \tau_i^g < -0.82$ also prevent conflict. The arc-arc conflict analysis also produces $\hat{u}_{ijl}^{gh} = 2.38$, meaning that h must depart k at least 2.38 units of time after g to avoid conflict. Figure 1(c) illustrates the situation when $\tau_k^h - \tau_i^g = 2.38$. At time $t=0$, g departs node i while h waits at k until time $t=2.38$. At $t=2.82$, g arrives at j and h is at coordinate $(2.8, 2.6)$, whose distance to j is exactly d . As in the previous case, this situation leads to no conflict as the vehicles are never at a distance of less than d while in motion. Indeed, any configuration of departure times such that $\tau_k^h - \tau_i^g \notin (-0.82, 2.38)$ will prevent geographic conflict.

Our arc-arc conflict conditions are general to any network topology as long as the participant arcs belong to Ω . Figure 2 depicts three special cases of arc-arc conflicts, in which we assume for simplicity $d=1$ and unit speeds. In the first case, two vehicles using the same arc (i.e., $(i, j) = (k, l)$) synchronize their departure times to avoid conflict while in motion. In Figure 2(a), our conflict analysis requires the vehicles to leave node i with a time difference of at least 1 unit, i.e., $\tau_i^h - \tau_i^g \notin (-1, 1)$. Figure 2(a) illustrates that the minimum distance d is preserved when $\tau_i^h - \tau_i^g = -1$. In Figure 2(b), vehicles transit two arcs that intersect (e.g., a road intersection with no right or left turns), requiring the vehicles to leave the departure nodes with a time difference $\tau_k^h - \tau_i^g \notin (-1.04, 1.55)$ to avoid conflict. Figure 2(b) illustrates the case where $\tau_k^h - \tau_i^g = -1.04$. At time $t=0$, g departs i while h is almost at the intersection, and at time $t=1.05$ vehicles are at their minimum distance (d). Figure 2(c) illustrates the case where $j = k$, meaning that g travels towards the location of h such that the departure times from i and j must satisfy $\tau_j^h - \tau_i^g \notin (0.59, 2)$. Figure 2(c) illustrates the case where $\tau_j^h - \tau_i^g = 0.59$, in which g departs i at $t=0$ and h waits at node j until

$t=0.59$, achieving a minimum distance of d between vehicles at time $t=1.30$.

Figures 1 and 2 illustrate how to prevent the geographic conflict while vehicles are moving. However, these constraints allow the distance between vehicles to be less than d when one of them is waiting at a node. For instance, the distance between g and h will be less than d if h waits at l after $t=1.41$ in Figure 1(b) or when g waits at j after $t=2.82$ in Figure 1(c). This situation may not be desirable in some related problems (e.g., military convoy planning), requiring additional *arc-node* constraints to prevent such conflict.

2.2.2. Arc-node conflict

Suppose that vehicle g uses arc (i, j) and vehicle h visits (and possibly waits at) node l such that $((i, j), l) \in \Psi$. In this case, the distance between the two vehicles at time t is given by $\|\mathbf{x}_i + t\mathbf{v}_{ij}^g - \mathbf{x}_l\|$, where we assume that the departure time of g is the reference time. Proposition 2 establishes an analogous result to Proposition 1 for the arc-node conflict. The proof of Proposition 2 is similar to that of Proposition 1.

Proposition 2. *If the differences in departure times t_1 and t_2 , where $t_1 \leq t_2$, lead to a geographic conflict, then any $t \in [t_1, t_2]$ also leads to a conflict.*

Because $((i, j), l) \in \Psi$, we know that geographic conflict exists at some time t such that $0 \leq t \leq c_{ij}^g$. Using Proposition 2, we can calculate a conflict interval $(\tilde{\ell}_{ijl}^{gh}, \tilde{u}_{ijl}^{gh})$ for the vehicles' departure times difference by performing the following two steps:

- Step 1: Solve $|\mathbf{x}_i + t\mathbf{v}_{ij}^g - \mathbf{x}_l| = d$ for t and obtain the roots t_1 and t_2 , where $t_1 \leq t_2$.
- Step 2: Return $\tilde{\ell}_{ijl}^{gh} = \max\{0, t_1\}$ and $\tilde{u}_{ijl}^{gh} = \min\{c_{ij}^g, t_2\}$.

The roots in Step 1 always exist and are real because $((i, j), l) \in \Psi$ and there are no constraints on t . Using this two-step procedure, we establish that no conflict arises if h

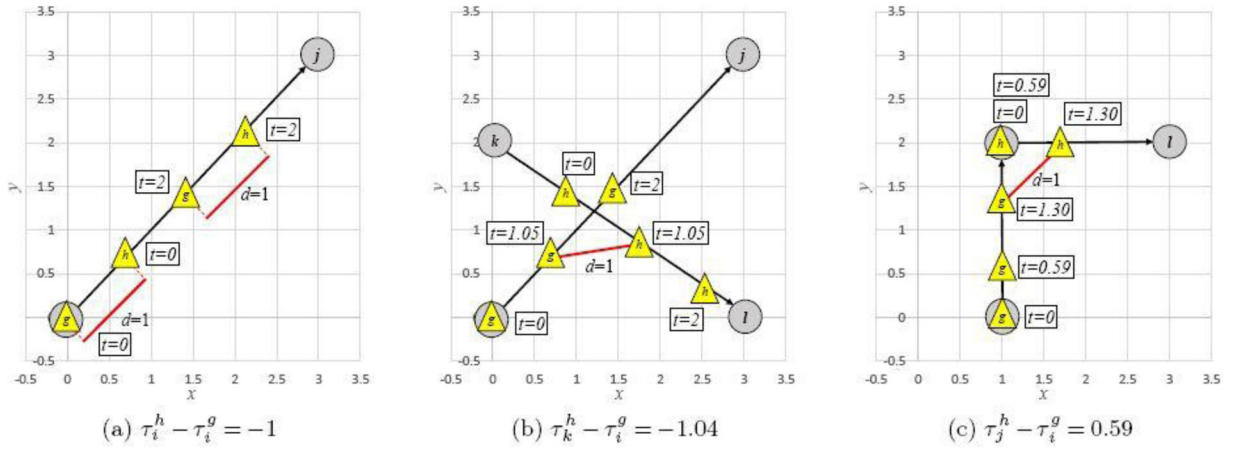


Figure 2. Special cases in the arc-arc conflict analysis.

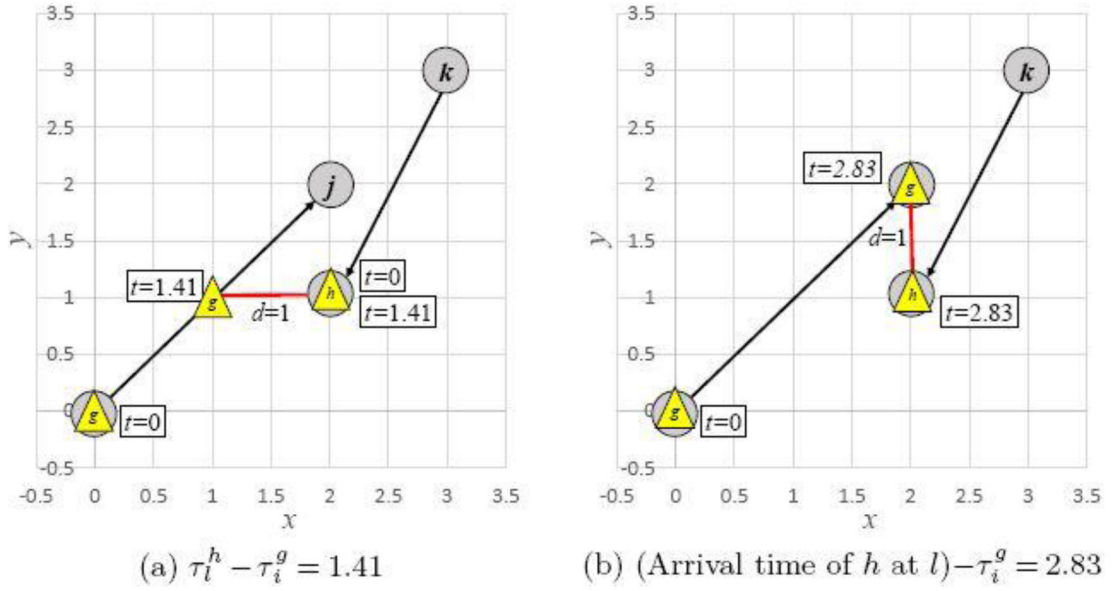


Figure 3. Arc-node conflict analysis.

leaves node l at most $\check{\ell}_{ijl}^{gh}$ units of time after g departs i , or if h arrives at l at least $\check{u}_{ijl}^{gh}$ units of time after g departs i . If we let τ_i^g be the departure time of vehicle g from node i along arc (i, j) , τ_l^h be the departure time of vehicle h from node l , and τ_k^h be the departure time of vehicle h from node k along arc (k, l) , such that $((i, j), l) \in \Psi$, then the disjunction $(\tau_l^h - \tau_i^g \leq \check{\ell}_{ijl}^{gh}) \vee (\tau_k^h + c_{kl}^h - \tau_i^g \geq \check{u}_{ijl}^{gh})$ avoids any conflict while vehicle g is moving and vehicle h is waiting after arriving into node l using arc (k, l) .

Figure 3 illustrates the arc-node conflict conditions when g is traveling from i to j , h is waiting at l , $d=1$, and unit speeds. Clearly, $((i, j), l) \in \Psi$, indicating that g and h will have conflicting locations at some time $t \in [0, \|\mathbf{x}_j - \mathbf{x}_l\|]$. We obtain that $\check{\ell}_{ijl}^{gh} = 1.41$ and $\check{u}_{ijl}^{gh} = 2.83$. Figure 3(a) illustrates the situation when $\check{\ell}_{ijl}^{gh} = 1.41$, which forces h to leave l (to any other node) no later than $t=1.41$ to avoid conflict. Figure 3(b) describes the situation when $\check{u}_{ijl}^{gh} = 2.83$, which

implies that h must arrive at l (from k or any other node) not earlier than $t = 2.83$.

3. Mathematical programming formulation

We propose an MIP formulation for RASTC that uses a directed network $G = (N, A)$, where N is the set of nodes and A is the set of arcs. A set of vehicles V travels between known origin and destination nodes denoted by s_g and p_g , respectively, for each vehicle $g \in V$. We partition V into sets V_{st} to identify those vehicles traveling from s to t . Parameters a_j^g and b_j^g denote the minimum and maximum amount of time that vehicle g is allowed to wait at node $j \in N$, if visited. There is no maximum waiting time constraint at nodes s_g and p_g , for any $g \in V$. Departure times within the interval $(\check{\ell}_{ijkl}^{gh}, \hat{u}_{ijkl}^{gh})$ result in a conflict when vehicles g and h travel on arcs (i, j) and (k, l) , respectively, such that $((i, j), (k, l)) \in \Omega$. Similarly, the interval $(\check{\ell}_{ijk}^{gh}, \hat{u}_{ijk}^{gh})$ captures the arc-node conflict when vehicle g travels along arc (i, j)

and vehicle h uses node k such that $((i, j), k) \in \Psi$. For each vehicle g , we precalculate the shortest-path time from s_g to p_g , denoted by z_{sp}^g , using arc costs given by $c_{ij}^g + a_j^g$, for all $(i, j) \in A$. We use big- M parameters M_g and M_{gh} with positive value for $g, h \in V$.

The binary variable x_{ij}^g is equal to one if and only if $g \in V$ uses arc $(i, j) \in A$, and the continuous variable τ_i^g captures the departure time of g from node $i \in N$. We reformulate the arc-arc and arc-node conflict disjunctive conditions using a big- M approach. We use the binary variable f_{ijkl}^{gh} , which is equal to one if the conflict condition involving $\hat{u}_{ijkl}^{gh}$ is satisfied by vehicles g and h , $g \neq h$, traveling on arcs (i, j) and (k, l) , respectively, such that $((i, j), (k, l)) \in \Omega$. This variable is equal to zero if the departure times of g and h satisfy the disjunctive condition that uses $\hat{e}_{ijkl}^{gh}$. The binary variable e_{ijk}^{gh} models the arc-node conflict between vehicles g and h , $g \neq h$, when g is traveling arc (i, j) and h is waiting at node k such that $((i, j), k) \in \Psi$. If the departure times satisfy the arc-node condition that uses $\hat{u}_{ijk}^{gh}$, then e_{ijk}^{gh} is equal to one. Otherwise, if they satisfy the condition using $\hat{e}_{ijk}^{gh}$, then e_{ijk}^{gh} is equal to zero. The decision variable z captures the value of the objective function, which corresponds to the maximum relative deviation from each vehicle's shortest-path time. We define the sets of indices $\Lambda = \{(g, h, i, j, k, l) | g, h \in V, g \neq h, ((i, j), (k, l)) \in \Omega\}$, $\Gamma = \{(g, h, i, j, l) | g, h \in V, g \neq h, ((i, j), l) \in \Psi\}$, and $\Upsilon = \{(g, h, i, j, k, l) | g, h \in V, g \neq h, ((i, j), (k, l)) \in A, ((i, j), l) \in \Psi\}$ to capture the possible combinations of vehicles and network elements where conflict may occur. Constraints (4)–(19) describe the feasible region of RASTC:

$$\min \quad z \quad (3)$$

$$\text{s.t.} \quad \sum_{j: (i, j) \in A} x_{ij}^g - \sum_{j: (j, i) \in A} x_{ji}^g = \begin{cases} 1, & \text{If } i = s_g \\ 0, & \text{If } i \in N \setminus \{s_g, p_g\}, \forall g \in V, i \in N \\ -1, & \text{If } i = p_g \end{cases} \quad (4)$$

$$\tau_i^g + c_{ij}^g + a_j^g \leq \tau_j^g + M_g(1 - x_{ij}^g), \quad \forall g \in V, \forall (i, j) \in A \quad (5)$$

$$\tau_i^g + c_{ij}^g + b_j^g \geq \tau_j^g - M_g(1 - x_{ij}^g), \quad \forall g \in V, \forall (i, j) \in A : i, j \notin \{s_g, p_g\} \quad (6)$$

$$\tau_k^h - \tau_i^g \leq \hat{e}_{ijkl}^{gh} + M_{gh} [f_{ijkl}^{gh} + (1 - x_{ij}^g) + (1 - x_{kl}^h)], \quad \forall (g, h, i, j, k, l) \in \Lambda \quad (7)$$

$$\tau_k^h - \tau_i^g \geq \hat{u}_{ijk}^{gh} - M_{gh} [(1 - f_{ijkl}^{gh}) + (1 - x_{ij}^g) + (1 - x_{kl}^h)], \quad \forall (g, h, i, j, k, l) \in \Lambda \quad (8)$$

$$\tau_l^h - \tau_i^g \leq \hat{e}_{ijl}^{gh} + M_{gh} [e_{ijl}^{gh} + (1 - x_{ij}^g)], \quad \forall (g, h, i, j, l) \in \Gamma \quad (9)$$

$$\tau_k^h + c_{kl}^h - \tau_i^g \geq \hat{u}_{ijk}^{gh} - M_{gh} [(1 - e_{ijl}^{gh}) + (1 - x_{ij}^g) + (1 - x_{kl}^h)], \quad \forall (g, h, i, j, k, l) \in \Upsilon \quad (10)$$

$$f_{ijkl}^{gh} \leq x_{ij}^g, \quad \forall (g, h, i, j, k, l) \in \Lambda \quad (11)$$

$$f_{ijkl}^{gh} \leq x_{kl}^h, \quad \forall (g, h, i, j, k, l) \in \Lambda \quad (12)$$

$$e_{ijl}^{gh} \leq x_{ij}^g, \quad \forall (g, h, i, j, l) \in \Gamma \quad (13)$$

$$e_{ijl}^{gh} \leq \sum_{k: (k, l) \in A} x_{kl}^h, \quad \forall (g, h, i, j, l) \in \Gamma \quad (14)$$

$$\tau_s^{g_1} \leq \dots \leq \tau_s^{g_{|V_{st}|}}, \quad \forall s, t \in N : |V_{st}| > 1 \quad (15)$$

$$\frac{\tau_{p_g}^g}{z_{sp}^g} \leq z, \quad \forall g \in V \quad (16)$$

$$\tau_i^g \geq 0, \quad \forall g \in V, i \in N \quad (17)$$

$$x_{ij}^g \in \{0, 1\}, \quad \forall g \in V, (i, j) \in A \quad (18)$$

$$f_{ijkl}^{gh} \in \{0, 1\}, \quad \forall (g, h, i, j, k, l) \in \Lambda \quad (19)$$

$$e_{ijl}^{gh} \in \{0, 1\}, \quad \forall (g, h, i, j, l) \in \Gamma \quad (20)$$

Constraints (4) impose flow-balance conditions for each node and each vehicle, similar to the classic multicommodity flow problem (Ahuja *et al.*, 1993). Constraints (5)–(6) help enforce the waiting times at each node and keep track of each vehicle's travel time. These constraints also help in eliminating cycles, as they are assumed to be infeasible for the problems of our interest. Constraints (5) guarantee that if vehicle $g \in V$ uses arc (i, j) (i.e., $x_{ij}^g = 1$), then its departure time from node j is at least a_j^g units of time after its arrival time (i.e., $\tau_i^g + c_{ij}^g$), which enforces the minimum waiting time. Similarly, Constraints (6) prevent vehicle g from waiting longer than b_j^g at node j , enforcing the maximum waiting time conditions. Due of the big- M parameters, these constraints are only active when g uses arc (i, j) , and are redundant otherwise. We strengthen the formulation by modifying the big- M values as our solution algorithm progresses and more information on the problem's optimal solution becomes available. This is discussed in Section 4.2.

Constraints (7) and (8) impose the arc-arc conflict constraints. If vehicles g and h use arcs (i, j) and (k, l) , respectively (e.g., $x_{ij}^g = x_{kl}^h = 1$), and there is an arc-arc conflict between such arcs, then the binary variable f_{ijkl}^{gh} forces the departure times to satisfy either (7) or (8). If $x_{ij}^g = 0$ or $x_{kl}^h = 0$, or both, then the corresponding arc-arc constraints are relaxed. Constraints (9) and (10) impose the arc-node conflict constraints. In this case, the binary variable e_{ijl}^{gh} forces the departure times to satisfy exactly one constraint between (9) and (10), when g is moving along arc (i, j) and h is waiting at l , and $((i, j), l) \in \Psi$. Constraints (11)–(14) strengthen the formulation by forcing the e - and f -variables to be zero when the network elements involved in the conflict are not used by vehicles g and h , as in this case the corresponding Constraints (7)–(10) will be already relaxed. When multiple identical vehicles travel between the same origin and destination, any permutation of a feasible routing and scheduling plan among vehicles is also feasible. Constraints (15) break such symmetry and reduce the

feasible space by imposing a nondecreasing order on the vehicles' departure times, which will not affect the optimal solution. In this case, imposing no more than $|V| - 1$ constraints suffices to eliminate the symmetries.

In the absence of conflict, the optimal path for any vehicle is the shortest path to the destination with arc costs given by the sum of travel and minimum waiting times. However, conflict may force a vehicle to deviate from its shortest path or to wait longer than required at one or more nodes, increasing the total travel time. For this reason, our objective function seeks a *fair* route planning and scheduling that minimizes the maximum deviation from each vehicle's shortest-path time. Constraints (16) and the objective function (3) model this situation. Constraints (17)–(19) enforce the nature of the decision variables.

Regarding RASTC's computational complexity, related static problems such as finding the maximum number of geographically disjoint paths can be solved in polynomial time (Neumayer *et al.*, 2009; Kobayashi, and Otsuki 2014; Neumayer *et al.*, 2015; Otsuki *et al.*, 2016). However, RASTC is NP-hard due to its dynamic nature (see proof in Appendix 1).

4. Network decomposition approach for RASTC

In this section, we develop a network decomposition scheme to expand the limits of our exact model. This is motivated by the prohibitively large number of decision variables and constraints in formulation (3)–(20), which makes RASTC unsolvable in reasonable time for medium- and large-scale instances using commercial solvers. Our approach is based on two observations:

1. Not all vehicles use all the network components, which means that solving RASTC on a sub-network may produce an optimal solution for the problem on the complete network.
2. Conflict may not occur on every pair of network components in Ω or Ψ , which means that enforcing a subset of conflict constraints may be enough to produce a conflict-free optimal solution for the complete problem.

4.1. Notation and additional definitions

We use the notation $\text{RASTC}(G, \Lambda, \Gamma, Y)$ to describe the problem in (3)–(20) with parameters given by G , Λ , Γ , and Y , where we assume that other parameters (e.g., V) will not change. We denote the global optimal value of $\text{RASTC}(G, \Lambda, \Gamma, Y)$ by z^* , and use $z(\cdot)$ to denote the optimal value of RASTC with parameters given by $(\cdot)$. For instance $z(\tilde{G}, \tilde{\Lambda}, \tilde{\Gamma}, \tilde{Y})$ denotes the optimal solution to RASTC $(\tilde{G}, \tilde{\Lambda}, \tilde{\Gamma}, \tilde{Y})$. We use the notation $\hat{z}^i(\cdot)$ to represent the optimal value of RASTC if the parameters in $(\cdot)$ produce an upper bound on z^* , and $\check{z}^i(\cdot)$ if they produce a lower bound at iteration i .

We introduce the following network structures that are useful in our analysis. A *reduced network*, $G^r = (N^r, A^r)$, is a subgraph of G (i.e., $N^r \subseteq N$ and $A^r \subseteq A$) that contains at

least one path from s_g to p_g for all $g \in V$. The set of *boundary nodes* B of G^r is the set of nodes in N^r with an incoming or outgoing arc in $A \setminus A^r$. That is, $B = \{i \in N^r \mid (i, j) \in A \setminus A^r \vee (j, i) \in A \setminus A^r\} \subseteq N^r$. The network G^r induces a *complement network* $G^c = ((N \setminus N^r) \cup B, A \setminus A^r)$, which contains all elements in G but not in G^r , and also includes the boundary nodes. We use G^c to search for paths between each pair of boundary nodes $i, j \in B, i \neq j$ that can be used to augment G^r . We denote the elements (nodes and arcs) in the k th shortest-path of vehicle g from i to j in G^c as $\mathcal{P}_{ijg}^k(G^c)$, which allows us to define an *augmented network*

$$G^a = \left(\bigcup_{\substack{g \in V, \\ i, j \in B, i \neq j, \\ k=1, \dots, K}} \mathcal{P}_{ijg}^k(G^c) \right) \cup G^r,$$

where $G^r \subseteq G^a \subseteq G$, and K is the maximum number of paths allowed. Figure 4(a) shows an initial network $G = (N, A)$ whereas Figure 4(b) shows a possible reduced network G^r with boundary nodes in gray. Figure 4(c) shows the corresponding complement network and Figure 4(d) shows an augmented network, which in this case contains only one path (if any exist) for each pair of boundary nodes in G^c .

We enforce the following rules when constructing G^a . For each pair of nodes $i, j \in B, i \neq j$ and each vehicle g , we augment G^r with $\mathcal{P}_{ijg}^1(G^c)$, which contains the elements of a shortest-path between i and j in G^c with arc costs given by the sum of travel and minimum waiting times for each vehicle. If such a path consists of arc (i, j) only, then we also add $\mathcal{P}_{ijg}^2(G^c)$, the *second* shortest-path between i and j in G^c , to G^r in order to allow vehicles to wait at a non-boundary node in G^c , which may be optimal. Note that if $\mathcal{P}_{ijg}^2(G^c) \neq \emptyset$, then it must contain a non-boundary node in G^c that can be used for waiting, which is not the case if the path has only one arc. We illustrate the importance of this construction in the following example, which is also relevant when discussing the correctness of our decomposition algorithm in Section 4.2.

Example 1. This example illustrates an instance of RASTC in which waiting is optimal and where the travel time for some vehicles increases with respect to their shortest path due to the geographic conflict constraint. Consider the network in Figure 5(a), with $V = \{1, 2, 3\}$, and origin-destination pairs $(s_1, t_1) = (1, 5)$, $(s_2, t_2) = (5, 1)$, and $(s_3, t_3) = (9, 6)$. We assume that vehicles are not allowed to wait at nodes 2 and 6, and that the distance to enforce arc-arc and arc-node conflicts is d . Moreover, we assume unit speeds such that the travel times displayed on each arc are equal to the distance between nodes. The optimal solution to RASTC on G is that all vehicles depart their origins at time 0, using paths $1 \rightarrow 2 \rightarrow 6 \rightarrow 8 \rightarrow 7 \rightarrow 4 \rightarrow 5$, $5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$, and $9 \rightarrow 2 \rightarrow 6$, for Vehicles 1, 2, and 3, respectively. At time $3d$, Vehicle 2 is at node 4 and Vehicle 1 is at node 8, which means that Vehicle 1 must wait $d/2$ units of time until Vehicle 2 arrives at node 3. In this case, the optimal value of RASTC is $z^* = 8d/6d = 4/3$ (given by Vehicle 1's deviation from its shortest path). Figure 5(b) shows a possible reduced network, G^r . In an optimal solution to RASTC on G^r , Vehicle 1 has to wait $6d$ units of time

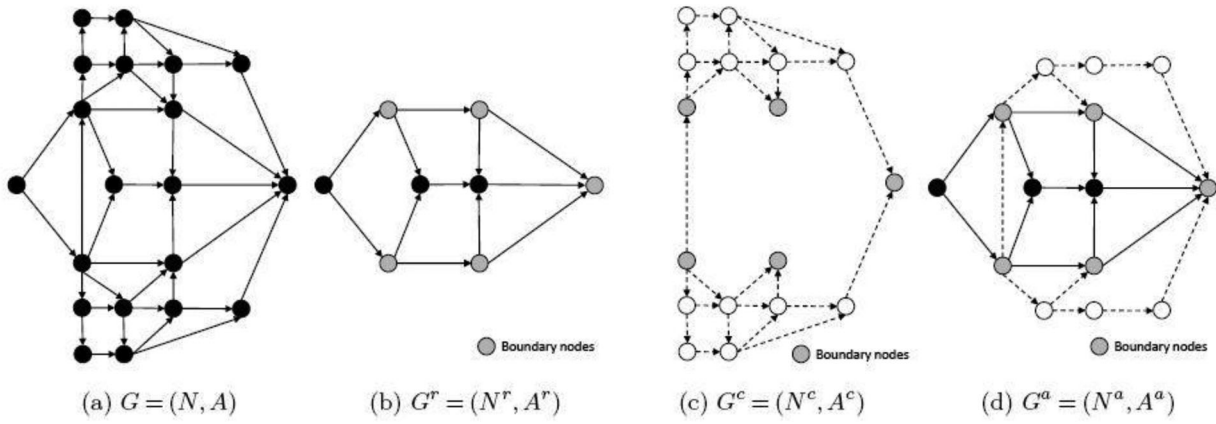


Figure 4. Network structures used to solve RASTC.

at node 1 until Vehicle 2 finishes its route, before traveling to node 5 using path $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$. As a result, $z(G^r, \Lambda, \Gamma, \Upsilon) = 12d/6d = 2$. Observe that Vehicle 1 cannot use nodes 2, 3, 4, 6 or 7 while Vehicle 2 is moving, as this will create a conflict. To create G^a , the shortest path between boundary nodes for any vehicle is arc (6, 7). If only this arc is added to G^r and RASTC is solved again on G^a , then the optimal solution will not change, as arc (6, 7) alone does not help Vehicle 1 in avoiding Vehicle 2. However, if we add the second shortest path to G^r , which is given by $6 \rightarrow 8 \rightarrow 7$, then the optimal solution on G^a is optimal to RASTC on G .

By construction, G^r is augmented with elements from G^c , thus the new elements added to G^a contain at least one non-boundary node, either from $\mathcal{P}_{ijg}^1(G^c)$ or $\mathcal{P}_{ijg}^2(G^c)$, for each vehicle g . The distance between these new nodes and some elements already in G^r may be less than d , creating a conflict. However, if both the maximum waiting time and the geographic conflict constraints are relaxed for these new nodes only, then they can be used for waiting. Using these elements, we define the lower-bound problem RASTC-R as a relaxation of RASTC in which Constraints (6)–(14) are not enforced for elements in $G^a \setminus G^r$ (e.g., white nodes and dotted arcs in Figure 4(d)). That is, there is no geographical conflict or maximum waiting time constraints when vehicles use those elements in $G^a \setminus G^r$. We use $z_R(G, \Lambda, \Gamma, \Upsilon)$ to denote the optimal solution to RASTC-R($G, \Lambda, \Gamma, \Upsilon$) and vector $(\mathbf{x}, \tau)$ to describe a feasible solution to RASTC or RASTC-R, where $\mathbf{x}$ and τ contain the values of the x - and τ -variables, respectively. We use $(\hat{\mathbf{x}}, \hat{\tau})$ and $(\check{\mathbf{x}}, \check{\tau})$ to denote optimal solutions to an upper and a lower bound problem, respectively.

The following propositions state some useful bounds for our decomposition algorithm, where we note that $z^* = z(G, \Lambda, \Gamma, \Upsilon)$.

Proposition 3. *The following conditions are satisfied for any network G , any reduced network G^r , and any conflict sets $\bar{\Lambda} \subseteq \Lambda, \bar{\Gamma} \subseteq \Gamma$, and $\bar{\Upsilon} \subseteq \Upsilon$:*

1. $z(G^r, \Lambda, \Gamma, \Upsilon) \geq z(G, \Lambda, \Gamma, \Upsilon)$
2. $z(G, \Lambda, \Gamma, \Upsilon) \geq z(G, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$

Proposition 4. *For a given network G and a reduced network G^r such that the optimal solution to RASTC($G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}$) is feasible to RASTC($G^r, \Lambda, \Gamma, \Upsilon$), where $\bar{\Lambda} \subseteq \Lambda, \bar{\Gamma} \subseteq \Gamma$, and $\bar{\Upsilon} \subseteq \Upsilon$, then $z(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}) \geq z(G, \Lambda, \Gamma, \Upsilon)$.*

Proposition 5. *For a given network G , a reduced network $G^r \subset G$, and conflict sets $\bar{\Lambda} \subseteq \Lambda, \bar{\Gamma} \subseteq \Gamma$, and $\bar{\Upsilon} \subseteq \Upsilon$, $z(G, \Lambda, \Gamma, \Upsilon) \geq z_R(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$.*

4.2. Decomposition algorithm

Algorithm 1 describes our network decomposition approach. Line 1 initializes conflict sets $\bar{\Lambda}, \bar{\Gamma}$, and $\bar{\Upsilon}$ to empty, as they will be dynamically populated when encountering conflicts. Line 1 also initializes the upper and lower bound values, UB_0 and LB_0 , and a counter i to track the number of iterations. Line 2 constructs a *feasible* reduced network that guarantees that at least T paths exists from s_g to p_g for every vehicle $g \in V$. This step is performed using Dijkstra's algorithm for $T=1$ (Dijkstra, 1959) or Yen's k shortest-path algorithm for $T>1$ (Yen, 1971). The loop in lines 3–17 is executed until convergence and consists of *upper bound* (Lines 5 and 6) and *lower bound* (Lines 7 and 8) routines. Using Proposition 4, Line 5 obtains an upper bound on z^* by solving RASTC on the reduced network using Algorithm 2 (see Section 4.2.1), where conflict constraints are added dynamically in a cutting-plane fashion. This strategy drastically reduces the number of x -, e -, and f -variables, as well as the number of conflict constraints. Line 5 also produces a feasible solution for RASTC($G, \Lambda, \Gamma, \Upsilon$), whose objective value and solution are stored in an incumbent in Line 6. In Line 7, our algorithm constructs the augmented network G^a induced by G^r . Line 8 produces a lower bound on z^* by following the rules from Section 4.1 to construct G^a . Section 4.2.2 provides a polynomial time algorithm to construct G^a . Using Proposition 5, Line 8 constructs a lower bound on z^* by solving problem RASTC-R($G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}$), whose optimal value is saved in Line 9.

Lines 10 and 11 verify a first stopping condition for our algorithm. If $(\check{\mathbf{x}}, \check{\tau})$ is feasible to RASTC (i.e., no conflict or maximum waiting time violations in $G^a \setminus G^r$), then such a

solution is optimal. Lines 13 and 14 verify an additional stopping condition that occurs when the optimal solution to $\text{RASTC-R}(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ only uses elements in G^r or when this solution uses elements in $G^a \setminus G^r$, but has the same objective function value equal to the best known upper bound. In such cases, upper and lower bound values are the same, and the incumbent $(\bar{\mathbf{x}}, \bar{\tau})$ is optimal. If none of these conditions is satisfied, then G^r is augmented in Line 15, and the algorithm goes to Line 3. Although in Line 15 we augment G^r , we only allow new elements to be used by vehicles needing them, according to $(\bar{\mathbf{x}}^i, \bar{\tau}^i)$. This reduces the number of binary variables in the problems solved in Lines 5 and 8. Algorithm 1 can stop at an iteration i in which $UB_i > LB_i$ as a result of Lines 10 and 11.

At each iteration $i > 1$ of Algorithm 1, we tighten the MIP used in Line 5 by updating the value of the big- M parameters. We use $M_g = UB_{i-1} z_{SP}^g$ and $M_{gh} = UB_{i-1} \max\{z_{SP}^g, z_{SP}^h\}$ for vehicles $g, h \in V$. We initialize the big- M values using the same expressions, but having a specific UB -parameter for vehicle $g \in V$ given by $\sum_{h \in V} z_{SP}^h / z_{SP}^g$, and for each pair of vehicles $g, h \in V$ given by $\max\{\sum_{\ell \in V} z_{SP}^\ell / z_{SP}^g, \sum_{\ell \in V} z_{SP}^\ell / z_{SP}^h\}$. These values capture the worst-case situation in which vehicles move one at a time. Using this strategy, the MIP becomes stronger as Algorithm 1 progresses because the UB -values are nonincreasing. Before proving the finite termination and correctness of Algorithm 1, we provide more details on the *upper bound* (Lines 5 and 6) and *lower bound* (Lines 7 and 8) routines.

Algorithm 1. Network Decomposition Algorithm for RASTC

- 1: Initialize $\bar{\Lambda} = \emptyset, \bar{\Gamma} = \emptyset, \bar{\Upsilon} = \emptyset, UB_0 = \infty, LB_0 = 0$, and set counter $i = 0$
 - 2: Initialize G^r with a set of T paths from s_g to p_g in G for each $g \in V$
 - 3: **while** $UB_i > LB_i$ **do**
 - 4: Set $i = i + 1$
 - 5: Solve $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ to obtain an optimal solution $(\hat{\mathbf{x}}^i, \hat{\tau}^i)$, optimal value $\hat{z}^i(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$, and updated sets $\bar{\Lambda}, \bar{\Gamma}$, and $\bar{\Upsilon}$ (see Section 4.2.1)
 - 6: Set $UB_i = \hat{z}^i(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ and update the incumbent solution $(\bar{\mathbf{x}}, \bar{\tau}) \leftarrow (\hat{\mathbf{x}}^i, \hat{\tau}^i)$ and objective $\bar{z} = UB_i$
 - 7: Calculate B and construct G^a using G^r (see Section 4.2.2)
 - 8: Solve $\text{RASTC-R}(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ to obtain an optimal solution $(\bar{\mathbf{x}}^i, \bar{\tau}^i)$ and optimal value $\bar{z}^i(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$
 - 9: Set $LB_i = \bar{z}^i(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$
 - 10: **if** $(\bar{\mathbf{x}}^i, \bar{\tau}^i)$ is feasible for RASTC **then**
 - 11: Incumbent $(\bar{\mathbf{x}}, \bar{\tau}) \leftarrow (\bar{\mathbf{x}}^i, \bar{\tau}^i)$ is optimal with objective $\bar{z} = LB_i$. Go to Step 15
 - 12: **if** $UB_i = LB_i$ **then**
 - 13: Incumbent $(\bar{\mathbf{x}}, \bar{\tau})$ is optimal with objective $\bar{z} = UB_i$. Go to Step 15
 - 14: **else** Augment G^r with the elements used in $(\bar{\mathbf{x}}^i, \bar{\tau}^i)$ that are not in G^r
 - 15: Return $(\bar{\mathbf{x}}, \bar{\tau})$ and $\bar{z}$
-

4.2.1. Upper bound

We use Algorithm 2 to solve $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ in Line 5 of Algorithm 1. After initializing i , Line 2 solves RASTC over G^r using a subset of conflicts. Line 3 stores an optimal solution and its objective value in an incumbent. The loop in Lines 4–9 iterates until the incumbent solution has no conflict violations. Line 6 identifies such violations using $\hat{\ell}$ -, $\hat{u}$ -, $\hat{\ell}$ -, and $\hat{u}$ -parameters. This can be done in $O(|N|^2|V|^2)$ steps by comparing all arcs in the paths of every pair of vehicles, where $|N|$ bounds the number of arcs in any path. Line 7 solves RASTC using the updated conflicts sets and Line 8 updates the incumbent with the resulting optimal objective value and solution. Line 10 returns an optimal solution to $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$, its objective value, and the updated conflict sets. Algorithm 2 finishes in a finite number of iterations, due to the finite size of the conflict sets. The solution obtained upon termination is feasible to $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ and also optimal given Proposition 4. This means that $z(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}) = \hat{z}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$.

4.2.2. Lower bound

We revisit the importance of the rules to construct G^a . In Figure 5(b), the shortest path between boundary nodes is arc (6, 7). If only this arc is added to G^r , then Algorithm 1 erroneously stops with a suboptimal solution, as the solution to RASTC-R in Line 8 would be the same as the solution to $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$. In this case, $z_R(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ is not a lower bound on z^* . If G^a follows the rules from Section 4.1, then the optimal solution to $\text{RASTC-R}(G^a, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ is optimal to $\text{RASTC-R}(G, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$.

Algorithm 2. Upper Bound Algorithm for $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$

- 1: Set counter $i = 0$
 - 2: Solve $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ to obtain an optimal solution $(\mathbf{x}^i, \tau^i)$ and optimal value $z^i(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$
 - 3: Set $\hat{z}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}) = z^i(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ and update the incumbent solution $(\hat{\mathbf{x}}, \hat{\tau}) \leftarrow (\mathbf{x}^i, \tau^i)$
 - 4: **while** $(\mathbf{x}^i, \tau^i)$ induces geographic conflict **do**
 - 5: Set $i = i + 1$
 - 6: Identify all the violated arc-arc and node-arc conflicts in $(\mathbf{x}^i, \tau^i)$ and update $\bar{\Lambda}, \bar{\Gamma}$, and $\bar{\Upsilon}$
 - 7: Solve $\text{RASTC}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ to obtain an optimal solution $(\mathbf{x}^i, \tau^i)$ and optimal value $z^i(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$
 - 8: Set $\hat{z}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}) = z^i(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon})$ and update the incumbent solution $(\hat{\mathbf{x}}, \hat{\tau}) \leftarrow (\mathbf{x}^i, \tau^i)$
 - 9: Return $(\hat{\mathbf{x}}, \hat{\tau}), \hat{z}(G^r, \bar{\Lambda}, \bar{\Gamma}, \bar{\Upsilon}), \bar{\Lambda}, \bar{\Gamma}$, and $\bar{\Upsilon}$
-

Using G^r as input, in Line 7 of Algorithm 1 we construct the set of boundary nodes in $O(|A^c|)$ steps and construct G^a by calculating the K (≥ 2) shortest-path between every pair of boundary nodes for each vehicle. This can be done in $O(|V||N^r|^2|N^c|(|N^c| \log |N^c| + |A^c|))$ steps using Yen's algorithm (Yen, 1971) and Fibonacci heaps (Fredman and Tarjan, 1987). In practice, we calculate paths between boundary nodes only for those vehicles that have used such nodes. Proposition 6 describes a *filtering* process that avoids

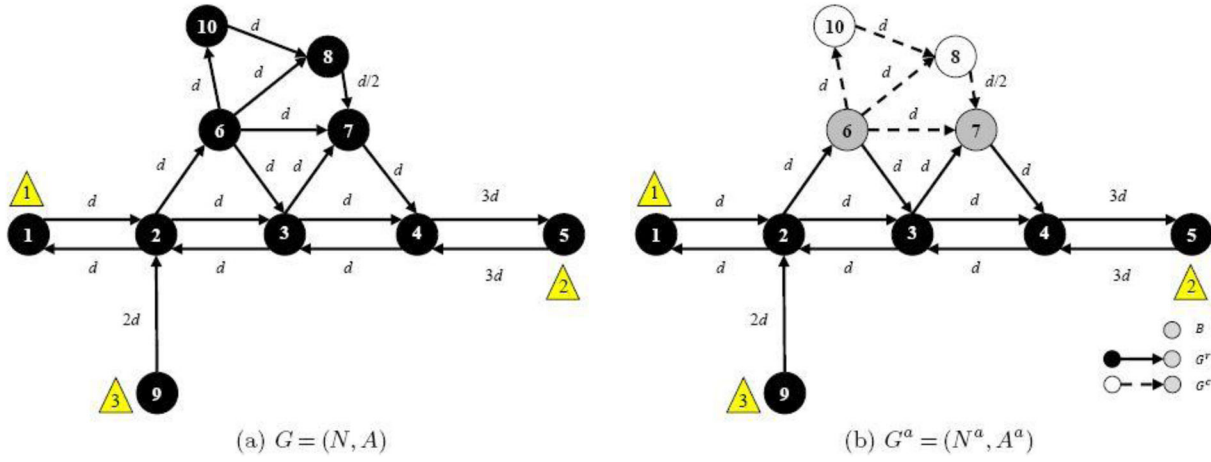


Figure 5. Construction of G^r .

adding unnecessary paths to G^a , reducing the number of decision variables in the problem solved in Line 8 of Algorithm 1. We use the function $c(\mathcal{P})$ to denote the total travel and minimum wait time along path $\mathcal{P}$. Proposition 7 proves the finite termination and correctness of Algorithm 1.

Proposition 6. Consider that at any given iteration, Algorithm 1 has constructed a reduced network G^r and its corresponding complement network G^c , and that the current upper bound is UB . Then, a shortest-path $\mathcal{P}_{ijg}(G^c)$, with $i, j \in B, i \neq j$, and $g \in V$, will not improve UB if for every $g \in V, c(\mathcal{P}_{sg,ig}(G)) + c(\mathcal{P}_{ijg}(G^c)) + c(\mathcal{P}_{j,p_{g,g}}(G)) \geq z_{sp}^g UB$.

Proposition 7. Algorithm 1 terminates in a finite number of iterations with an optimal solution to $RASTC(G, \Lambda, \Gamma, Y)$.

5. Computational results

We illustrate the features of RASTC and examine the performance of our network decomposition approach on real and randomly generated networks. To perform our computations, we use C++ with CPLEX 12.7 on a desktop computer with an Intel Core i7 2.40 GHz processor and 8.0 GB RAM. We set a solution time limit of 2 hours in all the experiments. With no enhancements, CPLEX cannot solve to optimality any of the proposed instances and sometimes cannot even find an integer feasible solution, while the optimality gaps are very large for those instances where an incumbent is available within the time limit.

5.1. Berlin's road network instances

We use a directed road network from the *Friedrichshain* district in east Berlin, Germany (Transportation Networks for Research Core Team, 2018), with 224 nodes and 523 arcs. The maximum Euclidean distance between any two nodes is 2.25 miles, which we denote by d_{max} . We allow vehicles to wait at nodes an unlimited amount of time and assume a speed of 35 mph for every vehicle. Moreover, we define parameter $\bar{d}$ to limit the maximum Euclidean distance that a vehicle can travel between origin and destination. Using this

network, Section 5.1.1 illustrates an optimal routing and scheduling plan for $|V| = 15$. Section 5.1.2 describes a procedure to create random instances out of this network and Section 5.1.3 summarizes the performance of our approach on such instances.

5.1.1. Illustrative example

In this section we describe in detail the features of a RASTC's optimal solution with $|V| = 15, \bar{d} = 2d_{max}/3$, and $d = 1050$ ft. Figure 6(a) shows the network and the randomly generated origins (labeled as $\blacktriangle$) and destinations (labeled as $\blacktriangledown$). The label next to the triangles is the vehicle index.

We depict the vehicles' position at various times t , where $t=0$ is the time at which the first vehicle starts moving. We illustrate the conflict by drawing a circle of diameter equal to d around each vehicle such that any conflict results in overlapping circles. Figure 6(b) shows that 10 vehicles start traveling at $t=0$, indicating that some have to wait into its journey avoid conflict. Figure 6(c) shows that Vehicle #10 only starts traveling at $t=6$, when Vehicle #8 is far enough to avoid conflict. As expected, the minimum distance requirement leads to a deterioration in the travel time for some vehicles, due to some having to wait at intermediate nodes or need to deviate from a shortest path to avoid conflict. Algorithm 1 solves this instance in 55 seconds, whereas the MIP formulation could not solve it within 2 hours.

Figure 7 illustrates other features of RASTC. The label next to a node is the node index. Figures 7(a)–7(c) show that waiting at a node is optimal for some vehicles. Vehicle #3 departs from Node 93 in Figure 7(a) and due to the road (directed) network structure, it has to make a U-turn visiting Nodes 125, 92, and 59 on the way to its destination. From $t=50$ to $t=95$, Vehicle #2 waits at Node 121 to avoid conflict with Vehicle #3 and the approaching Vehicle #9. Figures 7(d)–7(f) illustrate the case where Vehicle #2 is not allowed to wait at Node 121 (i.e., $b_{121}^2 = 0$). Vehicle #2 visits Node 121 at $t=50$ and continues to its destination while Vehicle #3 waits at Node 93. Vehicle #3 starts moving at $t=95$ when Vehicle #9 is far enough. Figures 7(g)–7(i) illustrate the case of heterogeneous vehicle speeds. We increase Vehicle #3's speed on arc (93, 125) to twice the speed in

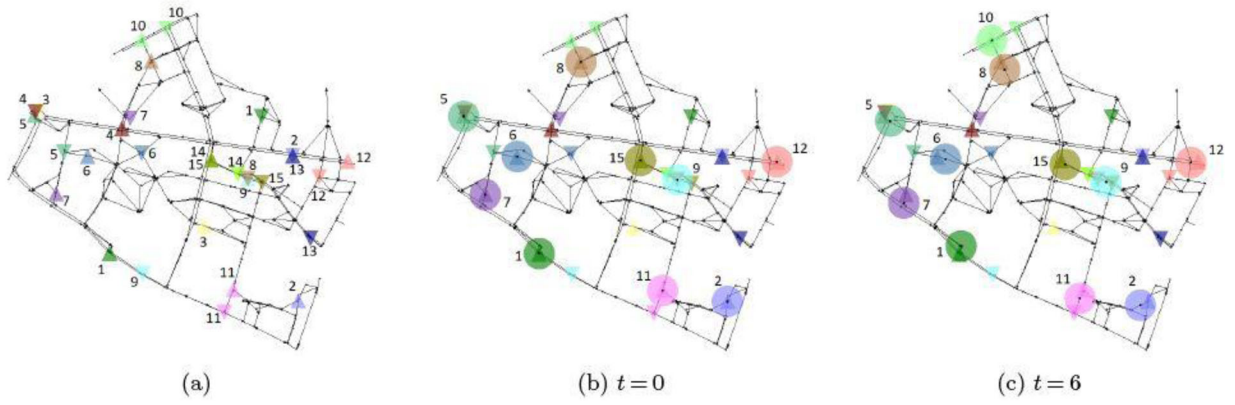


Figure 6. Origin-Destination pairs and waiting at origin nodes.

other arcs. As a result, Vehicle #3 departs Node 93 at some time $t < 50$ and by $t = 50$ is already at Node 59, which allows Vehicles #9 and #2 to freely travel without waiting.

5.1.2. Random instance generation

We create instances with 5, 10, 15 and 20 vehicles and with randomly generated origin and destination nodes. To induce various trip lengths, we use $\bar{d} \in \{d_{max}/3, 2d_{max}/3, d_{max}\}$ to represent short (S), medium (M), and long (L) trips, respectively. For each combination of $|V|$ and $\bar{d}$, we generate five random instances (replications). Moreover, for each trip length we solve problems with $d \in \{105, 210, 420\}$ (in feet), representing short (S), medium (M), and large (L) distance requirements for the geographic conflict.

5.1.3. Results

In this section, we compare the performance of our decomposition algorithm and the MIP in (3)–(20). The first three columns of Table 1 describe the instance solved, including the number of vehicles ($|V|$) and the different levels of $\bar{d}$ and d . We vary $\bar{d}$ to control the instance difficulty. Increasing $\bar{d}$ leads to longer trips, making it more likely for vehicles to encounter conflict. Longer trips also result in larger G^r and G^a networks, increasing the difficulty of the subproblems. Likewise, increasing d induces more conflicts, which will likely increase the number of iterations needed by our approach.

In Table 1, r is the percentage of instances (out of five) solved to optimality within a 2 hour time limit using the proposed decomposition approach. Additionally, t_{min} , t_{avg} , and t_{max} are the minimum, average, and maximum running times (in seconds) across solved instances for each combination of $|V|$, $\bar{d}$ and d . For those instances that timed out, g_{min} , g_{avg} , and g_{max} report the minimum, average, and maximum optimality gap calculated as $100(UB_k - LB_k)/UB_k$, where k is the last iteration before timing out. The optimality gap when $r = 100$ is zero, which we report as “-”. We do not report the solution time when $r = 0$. As expected, the solution time increases as $|V|$, $\bar{d}$, and d increase.

Our approach can solve all instances with $\bar{d}$ of type S to optimality, regardless of the number of vehicles and value of

d . Table 1 also reports the time to obtain and the quality of the first optimality gap. These values are given by t_{avg}^1 and $g_{avg}^1 = 100(UB_1 - 1)/UB_1$, where t_{avg}^1 is the average time required to solve the first upper bound problem. Note that the objective function value in RASTC is always at least equal to one. We also report the average number of iterations to solve an instance to optimality or before time-out, which is given by i_{avg} . On average, our decomposition algorithm requires few iterations and provides an initial feasible solution (UB) and initial gap in relatively short time.

We calculate the proportion of x -variables used in the decomposition strategy with respect to the MIP. We report the average value of this metric across vehicles and replications (times 100) as a proxy of the average number of network components used by each vehicle. We report these values for the upper and lower bound problems in the last iteration, which we denote by $|x|_{avg}^u$ and $|x|_{avg}^l$, respectively. We also report the proportion of τ -variables, which we denote by $|\tau|_{avg}^u$ and $|\tau|_{avg}^l$. The values c_{avg} and v_{avg} are the proportion of constraints and variables in the last lower bound subproblem (the largest problem solved), relative to the MIP. These metrics show that our approach significantly reduces the number of variables and constraints used. For example, instances 20-S-L require on average only 0.2% of the constraints, 0.3% of the variables, no more than 3.4% of the arc variables, and no more than 6.21% of the continuous variables required in the MIP.

5.2. Random network instances

Section 5.1 is focused on analyzing the performance of our approach for several instances out of the same road network. In this section, we study the performance of our approach on randomly generated networks. Section 5.2.1 describes a procedure to such instances and Section 5.2.2 summarizes the performance metrics.

5.2.1. Random instance generation

We generate layered networks in order to control the distance between nodes and arcs, which directly affects the

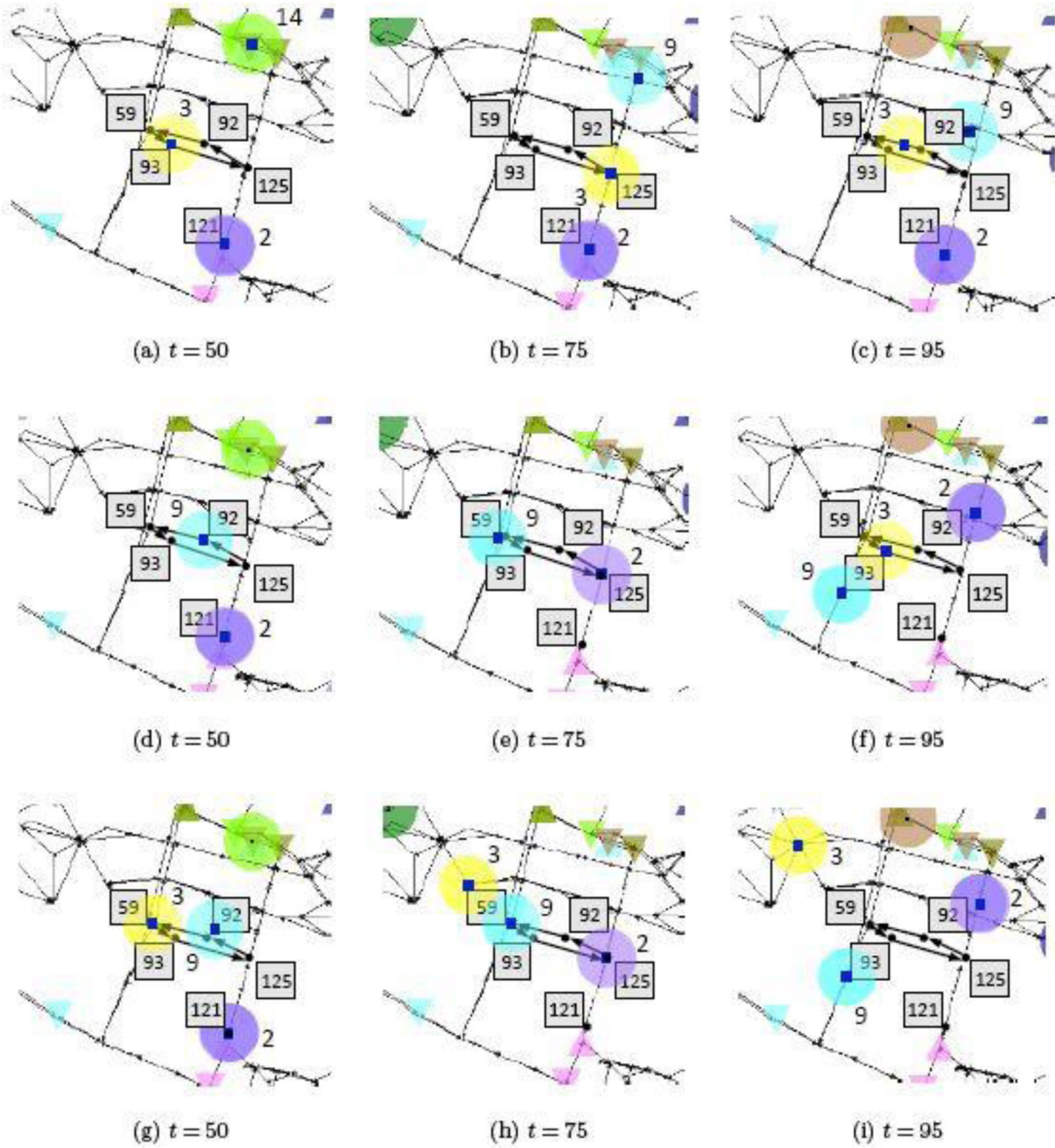


Figure 7. Waiting at intermediate nodes and heterogeneous vehicle speeds.

existence of conflict. These layered networks also provide many alternative paths for each vehicle, testing the limits of our decomposition approach, as many iterations may be needed to find useful network components. We create layered networks having n layers and n nodes per layer, resulting in $|N| = n \times n$ nodes. Nodes are arranged in a square of dimension 200×200 such that layers are evenly separated. The position of each node within a layer is chosen randomly. The i th node (from top to bottom) of the q th layer (from left to right) is connected to the i th node of layers $q+1$ and $q-1$ and to nodes $i-1$ and $i+1$ in the same layer, whenever these nodes exist. The resulting arrangement of nodes and arcs creates multiple conflicts when vehicles move. Figure 8 shows random layered networks with $|N| =$

6×6 , $|N| = 10 \times 10$, and $|N| = 14 \times 14$ nodes. All vehicles are assumed to have a unit speed.

We create networks with sizes ranging from 4×4 to 14×14 nodes. For each size, we create instances with 10, 15, 20, and 25 vehicles with randomly generated origin and destination nodes for each vehicle and with $d \in \{2, 8\}$, representing short (S) and large (L) distance requirements. We impose no restriction on the trip's length of any vehicle (i.e., $\bar{d} = \infty$) and assume that there are no minimum or maximum waiting times on any node. For each combination of $|N|$, $|V|$, and d , we generate three random replications. Because of the number of alternative paths and the number of arcs, we expect these layered networks to be more difficult than those from Berlin's road network.

Table 1. Computational performance of the proposed decomposition approach on Berlin's road network instances ($T = 1$, $K = 2$).

$ V $	$\bar{d}$	d	r	t_{min}	t_{avg}	t_{max}	g_{min}	g_{avg}	g_{max}	t_{avg}^1	g_{avg}^1	i_{avg}	$ x _{avg}^u$	$ x _{avg}^l$	$ t _{avg}^u$	$ t _{avg}^l$	C_{avg}	V_{avg}
5	S	S	100	0.45	1.07	1.84	-	-	-	0.95	0.00	1.00	1.12	1.12	3.07	3.07	0.40	0.31
		M	100	0.81	1.31	1.80	-	-	-	0.76	3.32	1.00	1.12	1.35	3.07	3.34	0.37	0.32
		L	100	0.04	0.54	1.16	-	-	-	0.70	5.01	1.00	1.12	1.39	3.07	3.39	0.27	0.24
	M	S	100	0.68	4.72	18.25	-	-	-	1.45	4.48	1.20	2.36	3.59	5.86	7.46	0.64	0.80
		M	100	0.20	7.91	34.59	-	-	-	1.15	5.34	1.00	2.19	3.66	5.57	7.50	0.57	0.71
		L	100	0.07	9.28	42.55	-	-	-	1.54	7.99	1.60	2.42	3.08	5.91	6.73	0.40	0.48
10	S	S	100	0.38	15.67	73.00	-	-	-	1.47	6.95	1.40	2.36	4.52	5.91	9.16	0.75	1.02
		M	100	0.28	11.36	51.09	-	-	-	0.99	7.32	1.20	2.39	4.59	5.96	9.19	0.68	0.92
		L	100	0.41	125.25	610.51	-	-	-	1.25	10.27	2.00	2.94	5.18	6.91	10.03	0.53	0.75
	M	S	100	2.55	18.40	36.52	-	-	-	2.49	20.81	2.20	1.54	2.60	3.88	5.43	0.40	0.57
		M	100	12.31	29.35	70.90	-	-	-	2.78	23.96	2.40	1.57	2.99	3.89	5.99	0.38	0.54
		L	100	24.99	42.71	85.71	-	-	-	4.04	28.16	3.20	1.87	3.29	4.41	6.38	0.28	0.41
15	M	S	100	4.98	52.17	153.41	-	-	-	5.46	16.73	2.20	2.53	4.14	6.09	10.83	0.49	0.73
		M	100	6.46	99.06	333.53	-	-	-	6.90	23.32	2.00	2.95	5.89	6.86	10.83	0.57	0.87
		L	80	10.30	52.36	116.99	43.17	43.17	43.17	8.40	28.48	1.60	2.65	7.31	6.36	13.14	0.47	0.74
	L	S	100	22.37	90.43	198.98	-	-	-	7.29	24.85	2.20	3.10	6.21	7.27	11.33	0.59	0.91
		M	100	48.05	997.42	4294.11	-	-	-	8.82	26.62	3.00	3.43	6.51	7.78	11.66	0.56	0.85
		L	80	61.32	789.72	2846.64	31.91	31.91	31.91	10.60	28.92	3.40	3.51	7.86	7.95	13.78	0.47	0.73
20	S	S	100	6.87	59.40	157.52	-	-	-	9.16	14.36	2.00	1.53	2.77	3.89	5.56	0.32	0.48
		M	100	6.55	97.15	264.72	-	-	-	11.39	18.94	2.40	1.73	3.19	4.25	6.17	0.31	0.47
		L	100	7.74	190.50	337.20	-	-	-	15.30	27.52	3.20	1.93	3.71	4.53	6.75	0.25	0.38
	M	S	100	11.52	324.80	800.78	-	-	-	19.20	24.43	2.40	2.62	5.01	6.12	9.34	0.46	0.73
		M	80	15.96	209.61	369.95	32.91	32.91	32.91	24.18	27.16	2.20	2.74	6.33	6.35	11.19	0.46	0.73
		L	60	24.27	1224.10	3430.73	13.96	27.04	40.12	34.82	35.17	2.00	2.67	6.86	6.27	12.03	0.34	0.55
20	L	S	60	199.89	796.85	1933.25	4.92	23.77	42.61	40.19	30.16	2.40	3.18	7.06	7.33	12.82	0.54	0.85
		M	40	269.35	1327.51	2385.67	1.51	26.58	45.82	65.45	35.53	2.60	3.27	8.31	7.55	14.35	0.52	0.80
		L	20	323.40	323.40	323.40	2.63	18.75	46.68	80.13	37.58	2.60	3.56	9.76	8.12	16.61	0.42	0.63
	S	S	100	10.41	99.50	154.74	-	-	-	16.82	18.75	1.80	1.44	2.68	3.66	5.29	0.27	0.42
		M	100	73.91	144.85	191.03	-	-	-	23.13	28.15	2.00	1.56	2.85	3.87	5.54	0.25	0.39
		L	100	167.70	551.37	1631.14	-	-	-	27.98	33.62	3.60	2.08	3.41	4.65	6.21	0.20	0.31
20	M	S	20	587.78	587.78	587.78	20.19	32.33	36.85	87.07	30.26	1.40	2.22	8.01	5.54	13.84	0.46	0.74
		M	40	398.44	484.54	570.64	0.99	25.27	37.88	98.89	31.61	1.80	2.55	8.15	6.17	14.00	0.41	0.66
		L	20	1349.86	1349.86	1349.86	0.31	23.43	48.11	130.25	36.35	2.00	2.65	7.52	6.33	13.08	0.29	0.45
	L	S	40	1151.22	1379.54	1607.86	2.62	14.78	31.71	75.13	30.61	2.40	3.21	9.44	7.48	16.14	0.47	0.75
		M	40	859.81	1319.64	1779.46	1.41	28.77	45.52	224.56	41.23	2.20	3.31	9.91	7.71	16.87	0.46	0.70
		L	0	-	-	-	7.25	31.17	45.69	333.58	47.28	1.50	3.23	10.99	7.64	18.34	0.36	0.54

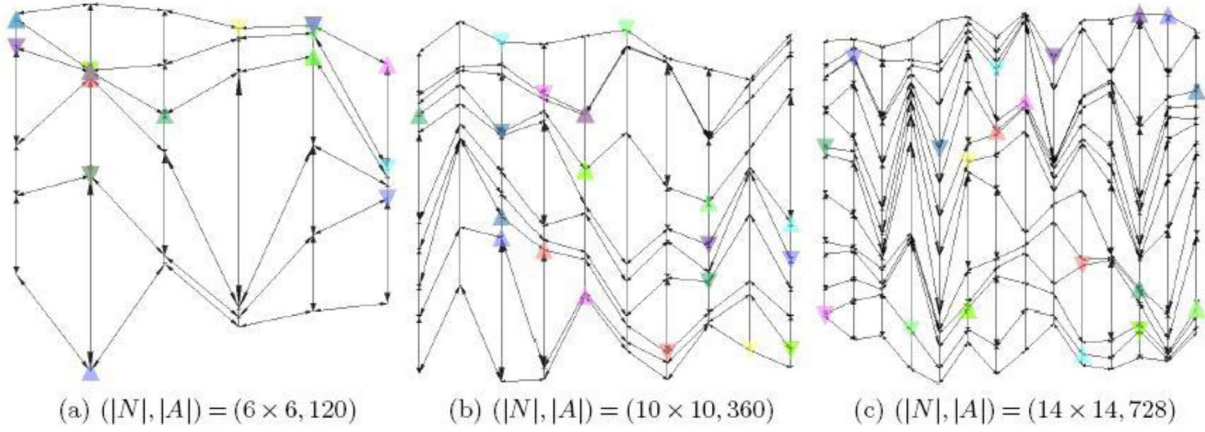


Figure 8. Random layered networks of different sizes.

5.2.2. Results

Table 2 shows the performance of our method on random layered networks. The first three columns characterize the instance, including $|N|$, $|A|$, $|V|$, and d . We compare our Decomposition Approach (DA) with the MIP solved with CPLEX on the complete network but adding the conflict constraints as encountered via cutting planes. That is, using Algorithm 2 with call $\text{RASTC}(G, \bar{\Lambda} = \emptyset, \bar{\Gamma} = \emptyset, \bar{Y} = \emptyset)$. The columns t_{DA} and t_{MIP} are the average solution times (in seconds) for our approach and the MIP, respectively, for instances solved within the time limit, whose number is shown in parenthesis. The column g_{DA} is the average optimality gap for the decomposition approach calculated as in Table 1. Our decomposition approach solves instances of up to 10 vehicles on 14×14 networks, whereas the largest instance solved with the MIP via cutting planes has 10 vehicles on the easiest setup of a 6×6 network. These results show that the proposed decomposition approach outperforms the MIP. The value of z^* is the average optimal objective function value for instances solved to optimality. If $z^* > 1$, then the geographic conflict requires some vehicles to wait at intermediate nodes or to deviate from their shortest path.

As expected, the performance of our decomposition algorithm depends on multiple factors, including $|N|$, $|V|$, $\bar{d}$, and d . For instance, in Berlin's Friedrichshain network (224 nodes and 523 arcs), we optimally solve some of the 20-vehicle instances with large $\bar{d}$ and medium d , whereas in the layered networks, we solve instances of 25 vehicles, 100 nodes, and 360 arcs but a small d . Our approach can handle more vehicles for some parameter combinations (e.g., small d or $\bar{d}$) or under other problem setups such as imposing conflict constraints only in some parts of the network, which is allowed in our modeling.

6. Final remarks and future work

We present an approach to impose geographical conflict conditions in a route assignment and scheduling problem. Using a polynomial-time pre-processing step, we identify regions in the network where geographic conflict may occur and provide conditions on the departure times from each

node that avoid conflict. By using a big-M approach to model disjunctive constraints, we reformulate this problem into a mixed-integer program that is very challenging to solve. To improve the solution time, we introduce a decomposition algorithm that takes advantage of the problem's network structure. Instead of solving the problem on the initial (complete) network, we limit our search to the most important sub-networks for each vehicle, which we dynamically construct as the optimization problem is solved. Solving the problem on a reduced network provides an upper bound on the optimal objective function value, which is helpful to eliminate network components that are not used in any optimal solution. We obtain a lower bound by allowing vehicles to use elements outside the reduced network, ignoring conflict or maximum waiting times. This strategy, combined with an iterative procedure to prevent conflicts as they are encountered, helps us maintain a small-sized problem which translates into favorable solution times. Our algorithm is able to solve instances that the MIP formulation cannot solve.

Our approach takes advantage of the sparsity of an optimal solution to construct a reduced network that is sufficient to identify an optimal solution for the complete network. Our approach can be generalized to other problems, where the solution is sparse and possibly without a network structure. The proposed decomposition approach follows the same principle of other classic methods: generate useful problem elements as needed while keeping a "master problem" small. In RASTC, we generate path segments aiming to improve the incumbent solution in the reduced network, guaranteeing the existence of a feasible solution at any iteration. Our method can be initialized using any set of candidate paths, for instance using the low-risk routes from Carotenuto *et al.* (2007). Moreover, our approach preserves the structure of RASTC at every iteration, which is advantageous because there is always a connection between integer and continuous variables, a known problem in other approaches such as Benders decomposition. Our proposed decomposition approach does not rely on duality theory, avoiding the challenges of potentially weak linear relaxations. However, our approach can be coupled with other methods to solve large problems (e.g., column generation,

Table 2. Computational performance of the proposed decomposition approach versus MIP on random layered network instances ($T = 1, K = 2$).

(N , A)	$ V $	d	t_{MIP}	t_{DA}	g_{DA}	z^*	(N , A)	$ V $	d	t_{MIP}	t_{DA}	g_{DA}	z^*
$(4 \times 4, 48)$	10	S	22 (3)	16 (3)	–	1.40	$(6 \times 6, 120)$	10	S	1951 (1)	21 (3)	–	1.07
		L	21 (3)	11 (3)	–	1.42			L	–	19 (3)	–	1.15
	15	S	26 (3)	11 (3)	–	1.44		15	S	–	31 (3)	–	1.15
		L	216 (3)	21 (3)	–	1.45			L	–	62 (3)	–	1.19
	20	S	431 (3)	23 (3)	–	1.42		20	S	–	59 (3)	–	1.23
		L	54 (1)	76 (3)	–	1.42			L	–	233 (3)	–	1.27
	25	S	1788 (1)	108 (3)	–	1.46		25	S	–	977 (3)	–	1.35
		L	–	169 (3)	–	1.49			L	–	1878 (2)	20	1.37
	10	S	–	49 (3)	–	1.08	$(10 \times 10, 360)$	10	S	–	61 (3)	–	1.03
		L	–	53 (3)	–	1.19			L	–	68 (3)	–	1.08
$(8 \times 8, 224)$	15	S	–	63 (3)	–	1.10		15	S	–	174 (2)	35	1.09
		L	–	250 (3)	–	1.20			L	–	96 (1)	40	1.16
	20	S	–	104 (2)	20	1.17		20	S	–	1632 (2)	36	1.08
		L	–	252 (2)	26	1.26			L	–	1550 (1)	38	1.16
	25	S	–	321 (1)	22	1.12		25	S	–	636 (1)	39	1.14
		L	–	1396 (2)	45	1.26			L	–	–	47	–
$(12 \times 12, 528)$	10	S	–	105 (3)	–	1.23	$(14 \times 14, 728)$	10	S	–	888 (2)	28	1.02
		L	–	227 (3)	–	1.17			L	–	509 (1)	39	1.07
	15	S	–	202 (2)	25	1.20		15	S	–	–	42	–
		L	–	324 (3)	–	1.22			L	–	–	43	–
	20	S	–	236 (1)	35	1.08		20	S	–	–	47	–
		L	–	–	34	–			L	–	–	43	–
	25	S	–	–	37	–		25	S	–	–	43	–
		L	–	–	45	–			L	–	–	51	–

Benders decomposition, Lagrangian relaxation), which can be used to accelerate the solution of the problems on the reduced or augmented networks.

Although in this article we focus on two-dimensional problems where conflict is prevented everywhere in the network, the vehicle coordination analysis also applies to three-dimensional problems (e.g., aerial or underwater vehicles) and other problems where conflict needs to be prevented only in some areas. A future research path is to include a variable speed or a discrete speed profile that vehicles can choose when traversing an arc. Currently, it is possible to approximate such a variable speed profile by adding nodes along an arc where vehicles can wait. These mechanisms help approximate any acceleration–deceleration decisions along the route at the expense of additional decision variables and constraints. This is motivated by the observation that geographical conflicts may be avoided not only through route selection and scheduling but also by choosing an appropriate speed at specific times. Additionally, our centralized approach can be used as a benchmark to determine the quality of decentralized approaches that locally resolve the geographical conflicts.

Division of Earth Sciences;

Acknowledgments

We thank the three anonymous reviewers whose suggestions helped improve this manuscript.

Funding

This material is based upon work supported by the National Science Foundation under Grant No. 1740042.

Notes on contributors

Navid Matin-Moghaddam is a PhD candidate in industrial engineering at Arizona State University. He holds a MSc degree in industrial

engineering (2015) from Clemson University and a BSc in industrial engineering (2013) from Sharif University of Technology. His research interests include the development and application of novel operations research and data science methodologies. His interdisciplinary research includes collaborations with researchers from various domains such as computer science, civil engineering, and education.

Dr. Jorge A. Sefair is an assistant professor in the School of Computing, Informatics, and Decision Systems Engineering (CIDSE) at Arizona State University, where he also is a Senior Sustainability Scientist at the Julie Ann Wrigley Global Institute of Sustainability, and affiliate faculty at the Simon A. Levin Mathematical, Computational, and Modeling Sciences Center (MCMSC), the Center for Biodiversity Outcomes, and the Center for Spatial Reasoning & Policy Analytics (CSRPA). Dr. Sefair holds a PhD degree in industrial and systems engineering from the University of Florida (2015), and an MSc degree in industrial engineering (2008), BSc in industrial engineering (2006), and BA in economics (2005) from Universidad de los Andes (Colombia). His research interests include network optimization, multi-stage optimization, and integer programming. In particular, he is motivated by applications of operations research in environmental planning, public policy, and urban systems. His research has been interdisciplinary, having published academic works with colleagues from a variety of fields, including civil engineering, public health, ecology, biology, and economics.

ORCID

Jorge A. Sefair  <http://orcid.org/0000-0002-5887-8938>

References

- Abichandani P., Mallory K., Hsieh M.A. (2013) Experimental multi-vehicle path coordination under communication connectivity constraints. In: Desai J., Dudek G., Khatib O., Kumar V. (eds.), *Experimental Robotics. Springer Tracts in Advanced Robotics*, vol 88. Springer, Heidelberg.
- Adamo, T., Bektaş, T., Ghiani, G., Guerriero, E. and Manni, E. (2018) Path and speed optimization for conflict-free pickup and delivery under time windows. *Transportation Science*, 52(4), 739–755.
- Ahuja, R.K., Magnanti, T.L. and Orlin, J.B. (1993) *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, Upper Saddle River, NJ.

- Bräysy, O. and Gendreau, M. (2005a) Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation Science*, **39**(1), 104–118.
- Bräysy, O., Gendreau, M. (2005b). Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation Science*, **39**(1), 119–139.
- Carotenuto, P., Giordani, S., Ricciardelli, S. and Rismondo, S. (2007) A tabu search approach for scheduling hazmat shipments. *Computers and Operations Research*, **34**(5), 1328–1350.
- Chardaire, P., McKeown, G.P., Verity-Harrison, S.A. and Richardson, S.B. (2005) Solving a time-space network formulation for the convoy movement problem. *Operations Research*, **53**(2), 219–230.
- Cordeau, J.F. and Laporte, G. (2007) The dial-a-ride problem: Models and algorithms. *Annals of Operations Research*, **153**(1), 29–46.
- Corréa, A.I., Langevin, A. and Rousseau, L.M. (2004) Dispatching and conflict-free routing of automated guided vehicles: A hybrid approach combining constraint programming and mixed integer programming. In: Régim JC., Rueher M. (eds.), *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. CPAIOR 2004. Lecture Notes in Computer Science*, vol 3011. Springer, Berlin, Heidelberg.
- Current, J. and Ratick, S.S. (1995) A model to assess risk, equity and efficiency in facility location and transportation of hazardous materials. *Location Science*, **3**(3), 187–201.
- De Jongh, A., Gendreau, and Labbe, M. (1999). Finding disjoint routes in telecommunications networks with two technologies. *Operations Research* **47**(1) 81–92.
- Diarrassouba, I., Mahjoub, M., Mahjoub, A.R. and Yaman, H. (2018) Finding failure-disjoint paths for path diversity protection in communication networks. *Annals of Telecommunications*, **73**(1–2), 5–28.
- Dijkstra, E.W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, **1**(1), 269–271.
- Dixit, A., Mishra, A. and Shukla, A. (2019) Vehicle routing problem with time windows using meta-heuristic algorithms: A survey, in *Harmony Search and Nature Inspired Optimization Algorithms*, Springer, Singapore, pp. 539–546.
- Erkut, E., and Verter, V. (1998) Modeling of transport risk for hazardous materials. *Operations Research*, **46**(5), 625–642.
- Esfandeh, T., Batta, R. and Kwon, C. (2018) Time-dependent hazardous-materials network design problem. *Transportation Science*, **52**(2), 454–473.
- Fazlollahtabar, H. and Saidi-Mehrabad, M. (2015) Methodologies to optimize automated guided vehicle scheduling and routing problems: A review study. *Journal of Intelligent & Robotic Systems*, **77**(3–4), 525–545.
- Ferrati, M. and Pallottino, L. (2013) A time expanded network based algorithm for safe and efficient distributed multi-agent coordination, in *Proceedings of the 52nd IEEE Conference on Decision and Control*. IEEE Press, Piscataway, NJ, pp. 2805–2810.
- Ferrera, E., Castaño, A., Capitán, A.R., Marrón, P.J. and Ollero, J.A. (2014) Multi-robot operation system with conflict resolution. *ROBOT2013: First Iberian Robotics Conference*. Springer, pp. 407–419.
- Ferrera, E., Castaño, A., Capitán, A.R., Ollero, J.A. and Marrón, P.J. (2013) Decentralized collision avoidance for large teams of robots, 2013 16th International Conference on Advanced Robotics (ICAR), Montevideo, pp. 1–6.
- Frazzoli, E., Mao, Z.H., Oh, J.H. and Feron, E. (2001) Resolution of conflicts involving many aircraft via semidefinite programming. *Journal of Guidance, Control, and Dynamics*, **24**(1), 79–86.
- Fredman, M.L. and Tarjan, R.E. (1987) Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, **34**(3), 596–615.
- Gopalan, R., Batta, R.R. and Karwan, M. (1990) The equity constrained shortest path problem. *Computers and Operations Research*, **17**(3), 297–307.
- Gopalan, R., Kolluri, K.S., Batta, R. and Karwan, M. (1990) Modeling equity of risk in the transportation of hazardous materials. *Operations Research*, **38**(6), 961–973.
- Grötschel, M., Monma, C.L. and Stoer, M. (1995) Design of survivable networks, in *Handbooks on Operations Research and Management Science*, vol. 7, Elsevier Science, North-Holland, Amsterdam, pp. 617–672.
- Hoy, M., Matveev, A.S and Savkin, A.V. (2015) Algorithms for collision-free navigation of mobile robots in complex cluttered environments: A survey. *Robotica*, **33**(3), 463–497.
- Hu, X., Chen, L., Tang, B., Cao, D. and He, H. (2018) Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles. *Mechanical Systems and Signal Processing*, **100**, 482–500.
- Jin, Q., Wu, G., Boriboonsomsin, K. and Barth, M. (2012) Multi-agent intersection management for connected vehicles using an optimal scheduling approach., in *Proceedings of the International Conference on Connected Vehicles and Expo*, IEEE Press, Piscataway, NJ, pp., 185–190.
- Kamal, M.A.S., Imura, J., Hayakawa, T., Ohata, A. and Aihara, K. (2015) A vehicle-intersection coordination scheme for smooth flows of traffic without using traffic lights. *IEEE Transactions on Intelligent Transportation Systems*, **16**(3), 1136–1147.
- Kerivin, H. and Mahjoub, A.R. (2005) Design of survivable networks: A survey. *Networks*, **46**(1), 1–21.
- Kobayashi, Y. and Otsuki, K. (2014) Max-flow min-cut theorem and faster algorithms in a circular disk failure model, in *Proceedings of the 2014, in Conference on Computer Communications IEEE, Press*, Piscataway, NJ, pp. 1635–1643.
- Krishnamurthy, N.N., Batta, R. and Karwan, M.H. (1993) Developing conflict-free routes for automated guided vehicles. *Operations Research*, **41**(6), 1077–1090.
- Kumar, P.N.R. and Narendran, T.T. (2008) Integer programming formulation for convoy movement problem. *International Journal of Intelligent Defence Support Systems*, **1**(3), 177–188.
- List, G.F., Mirchandani, P.B., Turnquist, M.A. and Zografos, K.G. (1991) Modeling and analysis for hazardous materials transportation: Risk analysis, routing/scheduling, and facility location. *Transportation Science*, **25**(2), 100–114.
- Malandraki, C. and Daskin, M.S. (1992) Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms. *Transportation Science*, **26**(3), 185–200.
- Margolis, J.T., Sullivan, K.M., Mason, S.J. and Magagnotti, M. (2018) A multi-objective optimization model for designing resilient supply chain networks. *International Journal of Production Economics*, **204**, 174–185.
- Neumayer, S., Efrat, A. and Modiano, E. (2015) Geographic max-flow and min-cut under a circular disk failure model. *Computer Networks*, **77**, 117–127.
- Neumayer, S., Zussman, G., Cohen, R. and Modiano, E. (2009). Assessing the vulnerability of the fiber infrastructure to disasters. *INFOCOM, 2009 Proceedings*, IEEE Press, Piscataway, NJ, pp. 1566–1574.
- Nishi, T., Hiranaka, Y. and Grossmann, I.E. (2011) A bilevel decomposition algorithm for simultaneous production scheduling and conflict-free routing for automated guided vehicles. *Computers & Operations Research*, **38**(5), 876–888.
- Omran, M.T., Jörg-Rüdiger Sack, J.R. and Zarrabi-Zadeh, H. (2013) Finding paths with minimum shared edges. *Journal of Combinatorial Optimization*, **26**(4), 709–722.
- Otsuki, K., Kobayashi, Y. and Murota, K. (2016) Improved max-flow min-cut algorithms in a circular disk failure model with application to a road network. *European Journal of Operational Research*, **248**(2), 396–403.
- Otto, A., Agatz, N., Campbell, J., Golden, B. and Pesch, E. (2018) Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey. *Networks*, **72**(4), 411–458.
- Phung, M.D., Quach, C.H., Dinh, T.H. and Ha, Q. (2017) Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection. *Automation in Construction*, **81**, 25–33.
- Pillac, V., Gendreau, M., Guéret, C. and Medaglia, A.L. (2013) A review of dynamic vehicle routing problems. *European Journal of Operational Research*, **225**(1), 1–11.

- Richards, A. and How, J.P. (2002) Aircraft trajectory planning with collision avoidance using mixed integer linear programming. *Proceedings of the 2002 American Control Conference*, vol. 3. IEEE Press, Piscataway, NJ, pp. 1936–1941.
- Rios-Torres, J. and Malikopoulos, A.A. (2017) A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps. *IEEE Transactions on Intelligent Transportation Systems*, **18**(5), 1066–1077.
- Sullivan, K.M. and Smith, J.C. (2014) Exact algorithms for solving a Euclidean maximum flow network interdiction problem. *Networks*, **64**(2), 109–124.
- Suurballe, J.W. (1974) Disjoint paths in a network. *Networks*, **4**(2), 125–145.
- Thomas, S., Deodhare, D. and Murty, M.N. (2015) Extended conflict-based search for the convoy movement problem. *IEEE Intelligent Systems*, **30**(6), 60–70.
- Toumazis, I. and Kwon, C. (2016) Worst-case conditional value-at-risk minimization for hazardous materials transportation. *Transportation Science*, **50**(4), 1174–1187.
- Transportation Networks for Research Core Team. (2018) <https://github.com/bstabler/TransportationNetworks> (accessed 2 March 2020).
- Yamashita, T., Izumi, K., Kurumatani, K. and Nakashima, H. (2005) Smooth traffic flow with a cooperative car navigation system, in *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, Association for Computing Machinery, New York, NY, pp. 478–485.
- Yen, J.Y. (1971). Finding the k shortest loopless paths in a network. *Management Science*, **17**(11), 712–716.
- Yu, J. and LaValle, S.M. (2012) Time optimal multi-agent path planning on graphs. *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, Technical Report WS-12-10, pp. 58–59.
- Zhu, F. and Ukkusuri, S.V. (2015) A linear programming formulation for autonomous intersection control within a dynamic traffic assignment and connected vehicle environment. *Transportation Research Part C: Emerging Technologies*, **55**, 363–378.
- Żotkiewicz, M., Ben-Ameur, W. and Pióro, M. (2010) Finding failure-disjoint paths for path diversity protection in communication networks. *IEEE Communications Letters*, **14**(8), 776–778.