VICTORIA UNIVERSITY
MELBOURNE AUSTRALIA

An agile vehicle-based dynamic user equilibrium scheme for urban traffic signal control

This is the Published version of the following publication

IET Intelligent Transport Systems

Special issue Call for Papers

Be Seen. Be Cited. Submit your work to a new IET special issue

Connect with researchers and experts in your field and share knowledge.

Be part of the latest research trends, faster.

Read more

IET The Institution of Engineering and Technology

IET Intelligent Transport Systems

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

An agile vehicle-based dynamic user equilibrium scheme for urban traffic signal control

Wei-Li Liu[1,4] | **Yue-Jiao Gong**[1] | **Wei-Neng Chen**[1] | **Jun Zhang**[2] | **Zhi Dou**[3]

[1] School of Computer Science and Engineering, South China University of Technology, Guangzhou, China

[2] University of Victoria, Melbourne, Australia

[3] College of Computer and Information Engineering, Henan Normal University, Xinxiang, China

[4] School of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, China

Correspondence

Yue-Jiao Gong, School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China.
Email: gongyuejiao@gmail.com

Abstract

Traffic Signal Control (TSC) is a fundamental task in modern intelligent transport systems. TSC is often formulated as a bi-level optimization problem, comprised by the signal timing at the upper level and the Dynamic User Equilibrium (DUE) traffic assignment at the lower level. Since DUE is non-convex, existing methods either formulate approximation models or adopt traffic simulators. However, approximation models may oversimplify the practical situations, while traffic simulators are usually time-consuming. This paper formulates a vehicle-based DUE (vDUE) model and proposes an agile method that can simultaneously maintain the computational simplicity and the traffic dynamics for the traffic assignment. Further, an agile TSC system is built by combining the vDUE at the lower level for the traffic assignment with an adaptive differential evolution algorithm at the upper level for the signal timing optimization. To enhance the effectiveness of optimization, the TSC problem formulation is also improved to make it better characterize the practical requirements. In the experiments undertaken, comparisons of different TSC methods are carried out on both real-world and synthetic transportation networks. The experimental results validate the effectiveness of the proposed agile TSC system in various traffic situations.

1 | INTRODUCTION

Traffic Signal Control (TSC) plays a crucial role in the modern intelligent transportation systems, especially considering the issues such as traffic congestion, inefficient resource utilization, and vehicle emission pollution [1]. In urban traffic systems, the area TSC is attracting more and more attention, which generally contains two major components: signal timing and traffic assignment. The signal timing aims to optimize parameters of traffic signals, such as cycle time, offset time, and green time. Meanwhile, the traffic assignment tries to appropriately allocate the given traffic demands to roads of a transportation network. It is noted that the signal timing optimization of the TSC is ordinarily conducted under constraints of the traffic assignment. Commonly, the traffic assignment adopts a Dynamic User Equilibrium (DUE) model, where vehicle drivers selfishly pursue the maximization of their own benefits until the dynamic flow equilibrium is reached [2].

In the literature, the area TSC problem is usually formulated as a bi-level optimization problem, where the upper level is the signal timing problem and the lower level is the DUE traffic assignment problem [2], [3]. For such a bi-level problem, the reliability of the dynamic flow equilibrium should greatly influence the quality of the TSC result, especially if the DUE has multiple optima [4]. That is because, within a bi-level TSC system, a solution to the DUE problem at the lower level provides equilibrium constraints to evaluate the solution to the signal timing problem at the upper level. At the same time, it is still challenging to obtain reliable dynamic flow equilibrium for existing bi-level TSC systems, which mostly adopt either approximation-based or simulation-based DUE solvers. On one hand, approximation-based DUE solvers aggregate the

intersection throughput in a static way while neglecting detailed vehicle movements [2]. Though simple enough to describe the macroscopic traffic status, the approximation models may oversimplify the practical situation, which reduces the reliability of the results. On the other, simulation-based DUE solvers provide microscopical description of vehicle behaviors and restore the realistic scenes by simulation. As the DUE problem is usually non-convex and difficult to be mathematically modeled, simulation-based models, compared with approximation ones, can help guarantee more accurate and reliable traffic dynamics [5]. However, the side effect is that they are rather time-consuming, which is difficult to be deployed in a real-time TSC system [6].

To address above issues, this paper formulates the vehicle-based DUE (vDUE) model and proposes an agile method for the traffic assignment of the bi-level TSC problem. Unlike traditional DUE solvers, the vDUE model combines the macroscopical description of the traffic status and the microscopical description of the vehicle behaviors for the sake of both simplicity and dynamism. Subsequently, an agile method is designed for the vDUE model to reroute traveling vehicles probabilistically according to the approximated saturation of their journeys ahead. Thus, the agile method can comply with the Wardrop's user equilibrium principle, which states that no vehicle can reduce its travel cost by alternating its route. Based on the vDUE model and the agile method for the lower-leveled traffic assignment, an agile TSC system is further proposed by adopting an adaptive differential evolution algorithm to effectively solve the signal timing optimization algorithm at the upper level. With the vDUE traffic assignment nested in the evaluation process, the adaptive differential evolution simultaneously optimizes cycle length, offset time, and green time ratios at all intersections, in order to minimize the average travel time of all vehicles. Besides, several improvements are made to facilitate the TSC optimization, including normalizing the value range of the cycle length to simplify the optimization algorithm, introducing the penalty coefficient into the fitness function to accelerate the solution optimization, and defining more patterns of signalized intersections to adapt to various complicated transportation networks. Experiments are conducted on various transportation networks to compare the agile TSC system with both classic and state-of-the-art TSC methods, and results validate the effectiveness of our proposed agile TSC system.

To summarize, this paper mainly has three contributions.

- First, a vDUE model is formulated. The vDUE model combines the macroscopical description of the traffic status and the microscopical description of the vehicle behaviors. The benefit is that the model can quickly respond to the practical dynamics while maintaining the computational simplicity.
- Second, an agile method is proposed for the traffic assignment based on the vDUE model. The proposed method can flexibly deal with different traffic situations (especially the crowded ones and those with emergencies), by probabilistically rerouting traveling vehicles based on the saturation approximation of their unfinished routes.

- Further, an agile TSC system is built by adopting a bi-level optimization model. The system fully considers the coordination relationship between traffic assignment and signal timing. The above agile traffic assignment scheme is executed at the lower level, while an adaptive differential evolution algorithm is developed to accomplish the signal timing optimization task at the upper level.

The rest of this paper is organized as follows. Section 2 reviews the related work, whilst Section 3 formulates the bi-level TSC optimization problem. Then, Section 4 introduces the framework and algorithms of the agile TSC system. Section 5 carries out experiments and comparisons to validate the effectiveness of our proposed TSC system, followed by conclusions drawn in Section 6.

2 | RELATED WORK

So far, various efforts have been made to solve the TSC problem. This section is going to review the related work from three aspects, namely, the problem models of TSC, the control methods of TSC, and the bi-level TSC with DUE.

2.1 | Problem models of TSC

To model the TSC problem, we mainly consider the control range, the signal parameters, and the traffic flow model.

In terms of the control range, works on TSC can be classified into the isolated control, the arterial control and the area control. Among them, the isolated control is the most fundamental type, which focuses on a single intersection and neglects its relationships with other intersections in a transportation network [7]. The arterial control focuses on several continuous intersections along an arterial road but neglects minor traffic flow on other roads [8]. Differently, the area control considers the coordination control of all intersections in a transportation network [9]. Since ignoring the coordination among different traffic signals can affect the overall performance of the network and even incur congestions, researchers have paid increasing attentions to the area control [3].

When signalized, an intersection provides different time slices for allowing its surrounding traffic flow in different directions to go when the green light is on. The traffic flow sharing the same time slice is termed as a *signal phase*. The sum of the green time of all signal phases is the *cycle length* of the related intersection. The ratio of the green time to the cycle length is the *green time ratio* of a signal phase. Besides, there is an *offset time* to activate the signal lights of each intersection. When performing the TSC on a network, the green time ratios are adjusted to distributively control the traffic flow at each intersection, whilst the cycle length and the offset time can be optimized to coordinate different intersections [10]. As a result, the signal parameters of an intersection involved in the TSC problem are typically the cycle length, the offset time, and the green time ratios [3].

The traffic flow models are generally classified into macroscopic and microscopic. On the one hand, the macroscopic models describe the relationship among amount, speed and density of the traffic flow, and they are mostly used for the traffic flow assignment on a transportation network. Examples of the macroscopic models include the mathematical function model, the Lighthill–Whitham–Richards model [11], the platoon dispersion model [12], and a newly proposed Physarum model [13]. On the other hand, the microscopic models describe the movements of individual vehicles, and they are mainly used for the traffic simulation to calculate the delay at the intersection. Examples of the microscopic models include the car-following model [14], the cell transmission model [15]. Based on pioneering work, this paper hybrids the macroscopic and microscopic models to describe the traffic flow. On the one hand, the traffic situation of the whole network is macroscopically described for the control efficiency, such as adopting a mathematical function to calculate the link travel time. On the other hand, the behavior of each vehicle is microscopically traced to better reflect the flow dynamism, such as rerouting a traveling vehicle if needed.

2.2 | Control methods of TSC

There are three major classes of control strategies for TSC, namely, the fixed-time control, the actuated control and the adaptive control [16]. The fixed-time control presets signal control parameters based on historical traffic conditions, whilst the actuated control detects current traffic conditions and adjusts signal parameters accordingly. The fixed-time control is easy and cheap to implement, whereas the actuated control is responsive to the traffic dynamics [1]. However, neither of them considers both historical and current traffic conditions at the same time [17]. The adaptive control is a more approved control method, where signal timing settings are initially generated based on prevailing traffic conditions and dynamically changed according to the feedback traffic conditions like the traffic flow [18].

To implement the adaptive control, the TSC are usually formulated into NP-hard optimization problems like the mixed-integer linear programming [15] . Thus, computational intelligence such as evolutionary computation, reinforcement learning, and fuzzy theory are often used to obtain satisfying solutions [19], [20]. For an example of the evolutionary computation, Jovanović et al. [9] developed an area-wide urban traffic control system based on the Bee Colony Optimization, by optimizing traffic signal parameters to minimize the total travel time of all network users. By exploiting reinforcement learning for traffic congestion reduction, Tan et al. [21] proposed a cooperative deep reinforcement learning framework for large-scale traffic grid signal control, and more efforts of this kind can also be found in other works like [22]. As for the fuzzy theory, Jin et al. [23] introduced a fuzzy intelligent traffic signal control system with simulation-based evaluation, which exhibits superior performance than several other controllers when a few detectors are out-of-order.

Evolutionary computation and reinforcement learning can both be considered as optimization techniques, whereas fuzzy theory is mainly used for decision-making and classification.

Owing to its natural superiority in global optimization and empirical evaluation, we adopt the evolution computation as the optimization approach for the TSC problem in this paper.

2.3 | Bi-level TSC for dynamic user equilibrium

As a common framework to solve the TSC problem, the bi-level TSC has received lots of attention from researchers. Typically, the upper level optimizes several signal parameters to achieve some overall objective of the network. For example, Chiou et al. [24] proposed a data-driven bi-level program to minimize overall travel delay and mitigate stochastic risk on links and proved its reliability against a high-consequence of exposure risk. As well, Zhang et al. [10] proposed a bi-level optimization strategy based on genetic algorithm on the cloud computing platform and proved its capability to avoid local optima and its fast solution efficiency.

Meanwhile, the lower level performs the DUE traffic assignment and provides a fitness value to evaluate the solution at the upper level. To settle the optimal traffic signal setting problem, Li et al. [3] integrated a genetic algorithm with a microscopic-traffic-simulation-based dynamic user equilibrium traffic assignment, to decouple the bi-level TSC problem into tractable single-level problems. To find equilibrium under the interaction between signal setting and traffic assignment, Chen et al. [5] adjusted the signal setting parameters through pretimed and actuated signal controls, and generated the user equilibrium dynamic traffic assignment flows through a simulation-based dynamic traffic assignment model. Han et al. [2] formulated a bi-level model for traffic signal control and solved it as a mathematical programming with equilibrium constraints, where the lower level embeds a dynamic network loading subproblem.

Although above works on the bi-level TSC for DUE have implemented the flow dynamism, they are either too time-consuming to perform the traffic simulation or not consistent enough to the reality by neglecting the vehicle movements. This paper proposes an agile method to conduct a both quick and realistic simulation for the DUE.

3 | PROBLEM FORMULATION

In a transportation network, the basic task of the TSC is to appropriately and dynamically set parameters of the traffic signals at several intersections, so as to achieve certain goals under some pre-specified constraints. This section elaborates the problem formulation of TSC based on virtual vehicles, including the basic terms, the solution representation, the fitness function, and the traffic assignment model.

3.1 | Basic terms

This paper aims to minimize the overall travel time of the network, by optimizing different kinds of signal parameters at each signalized intersection, namely, the cycle length and the offset
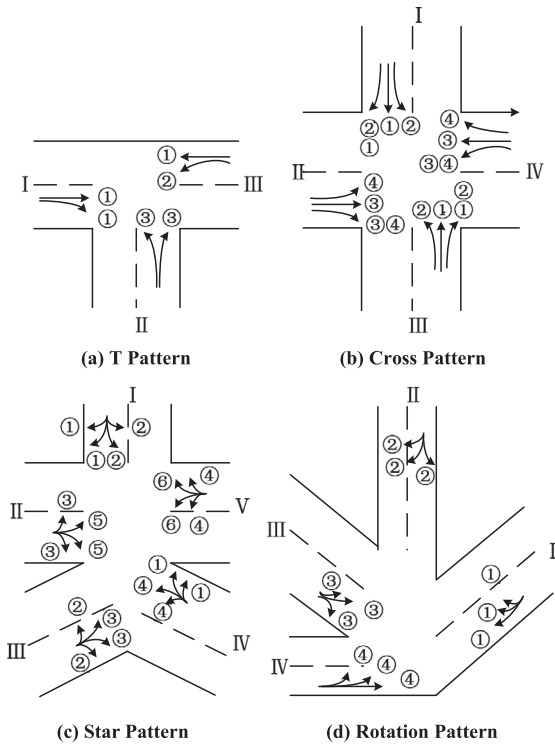
FIGURE 1 Patterns of signalized intersections

time ratio of a signalized intersection, the green time ratio of a signal phase at the intersection.

Definition 1 (Transportation Network). A transportation network is a directed graph $(\mathbf{N}, \mathbf{L})$, where $\mathbf{N}$ is the set of nodes representing intersections or terminals of roads, and $\mathbf{L} = \{i \to j | i, j \in \mathbf{N}\}$ is the set of links representing all roads in the network.

Definition 2 (Signalized Intersection). Intersections $\mathbf{N}^* \subseteq \mathbf{N}$ are a set of nodes connecting more than three links. Since traffic bottlenecks usually occur at intersections, traffic signals are installed to alleviate the traffic congestion and to smooth the traffic flow [17]. When signalized, an intersection is controlled with some signal lights such as green and red colors to indicate "go" and "stop" signs respectively.

Definition 3 (Signal Phase). A signal phase of a signalized intersection combines several nonconflicting turning movements at the intersection. In other words, all turning movements of a signal phase are active/inactive at the same time. Specifically, the turning movements of a signal phase are active during its green time but inactive during its red time. Besides, the offset time is also considered for each signal phase to capture the traffic dynamism. Note that the yellow time is not considered in this paper because it can be subtracted from the green time as stated in [3].

The set of signal phases $\mathbf{P}_n = \{\mathbf{P}_{n,k}\}$ defines the pattern of a signalized intersection n. As shown in Figure 1, this paper sum-

marizes signalized intersections of different shapes into four patterns to deal with an arbitrary complicated transportation network. Note that an intersection connecting more than six links can be set as a roundabout intersection. In Figure 1, every signalized intersection pattern is comprised of all connected links denoted with English letters and all possible turning movements described with arrowed lines. The circle digits beside an arrowed line represents the index of the signal phase where the relative movement belongs. For example, the intersection of T pattern in Figure 1(a) connects three links I, II, and III. The possible turning movements of the intersection can be expressed as $\{I \to III, I \to II, III \to I, III \to II, II \to I, II \to III\}$. These movements are classified into three signal phases, namely, $\{I \to III, I \to II, III \to I\}$, $\{III \to II\}$, and $\{II \to I, II \to III\}$, respectively.

Definition 4 (Time Slot). A time slot is the unit of time to perform the traffic assignment. Without further explanations, one time slot is viewed as 0.01 hours in this paper, as stated by Stabler [25]. Note that this definition provides a base to describe the flow dynamism in the traffic assignment. For example, properties of each link l such as the flow capacity f_l^{max} and the free-flow travel time t_l^{min} are initially known, but its flow amount f^{T_l} and travel time t^{T_l} may both change with the varied time slot T.

3.2 | Solution representation

A solution to TSC $\mathbf{s}$ is comprised of three vectors related with the signal parameters of all signalized intersections by

$$\mathbf{s} = (\mathbf{c}, \mathbf{o}, \mathbf{g}), \tag{1}$$

where the length of each vector should be equal to the number of signalized intersections $|\mathbf{N}^*|$ and the three vectors are respectively detailed into

$$\mathbf{c} = (c_1, \dots, c_n, \dots, c_{|N^*|}), \tag{2}$$

$$\mathbf{o} = (o_1, \dots, o_n, \dots, o_{|N^*|}), \tag{3}$$

$$\mathbf{g} = (\mathbf{g}_1, \dots, \mathbf{g}_n, \dots, \mathbf{g}_{|N^*|}). \tag{4}$$

For a signalized intersection n, Equations (2)–(4) defines $\mathbf{c}$, $\mathbf{o}$, and $\mathbf{g}$ with the cycle length rate c_n, the offset time ratio o_n, and the green time vector $\mathbf{g}_n$, respectively.

Then, $\mathbf{g}_n$ is further defined by

$$\mathbf{g}_n = (g_{n,1}, \dots, g_{n,k}, \dots), \tag{5}$$

where $g_{n,k}$ denotes the green time ratio for the kth signal phase of n. Since the phase number is mainly dependent on the pattern of the related intersection n, the size of its green time vector $|\mathbf{g}_n|$ may be different from that of another intersection.

Definition 5 (Cycle Length Rate). The cycle length rate c_n of a signalized intersection n is a decision variable of TSC within [0, 1]. It is mainly used to calculate the cycle length C_n, which refers to the total time that all its signal phases $\{\mathbf{P}_{n,k}\}$ sequentially complete their signal controls by switching lights.

Instead of the pre-defined bounds in pioneering work, this paper proposes to use the free-flow travel time values of the connected links to adaptively determine the value range of the cycle length for each signalized intersection. Specifically, C_n is limited within the range between the minimum and maximum travel time of all the connected links of n (denoted as $\mathbf{L}_n$). Hence, c_n is formulated as

$$c_n = \frac{C_n - \min_{l \in \mathbf{L}_n} t_l^{\min}}{\max_{l \in \mathbf{L}_n} t_l^{\min} - \min_{l \in \mathbf{L}_n} t_l^{\min}}. \tag{6}$$

Accordingly, C_n can be calculated by

$$C_n = \min_{l \in \mathbf{L}_n} t_l^{\min} + c_n \times \left(\max_{l \in \mathbf{L}_n} t_l^{\min} - \min_{l \in \mathbf{L}_n} t_l^{\min} \right). \tag{7}$$

Definition 6 (Offset Time Ratio). The offset time ratio o_n is also a decision variable of TSC, and its value is limited within a pre-defined range of $[o^{\min}, o^{\max}]$. It refers to the ratio of the relative offset time O_n to the related cycle length C_n. As a result, o_n is formulated as

$$o_n = \frac{O_n}{C_n}. \tag{8}$$

Accordingly, O_n can be calculated by

$$O_n = o_n \times C_n. \tag{9}$$

Definition 7 (Green Time Ratio). The green time ratio $g_{n,k}$ for the kth signal phase of n is another decision variable of TSC, and its value is limited within a pre-defined range of $[g^{\min}, g^{\max}]$. It refers to the ratio of the relative green time to the related cycle length. As a result, $g_{n,k}$ is formulated as

$$g_{n,k} = \frac{G_{n,k}}{C_n}. \tag{10}$$

Accordingly, $G_{n,k}$ can be calculated by

$$G_{n,k} = g_{n,k} \times C_n. \tag{11}$$

Furthermore, the cycle length of an intersection is actually the sum of the green time of all its signal phases, when considering only the green time and the red time for each signal phase. In other words, the green time ratios of all signal phases of an intersection n should satisfy

$$\sum_k g_{n,k} = 1. \tag{12}$$

3.3 | Fitness function

In this paper, the fitness function of TSC is defined as the average travel time of all vehicles that are generated according to the flow demands of the transportation network.

Definition 8 (Flow Demand). The flow demands of a network $\mathbf{F}$ consist of the trips of all Origin-Destination (OD) pairs [26], as expressed by

$$\mathbf{F} = \{F_{o,d} \mid o, d \in \mathbf{N}\}. \tag{13}$$

Without further explanations, this paper follows Stabler [25] to describe the flow demands with the hourly flow.

Definition 9 (Virtual Vehicle). For each OD pair (o, d), virtual vehicles $\mathbf{V}_{o,d}$ are generated according to the flow demand $F_{o,d}$ to simulate the traffic assignment of the network. As a result, the number of vehicles denoted by $|\mathbf{V}_{o,d}|$ is proportional to the related flow demand, as expressed by

$$|\mathbf{V}_{o,d}| = \frac{F_{o,d}}{K}, \tag{14}$$

where K is a positive integer pre-defined for the network.

Key properties of a vehicle v include the OD pair (O_v, D_v), the starting time slot at the origin T^{ori_v}, the time slot arriving at the destination T^{des_v}, and the route $\mathbf{R}_v = (O_v, ..., D_v)$. Although (O_v, D_v) and T^{ori_v} are known in advance, T^{des_v} can only be obtained after constructing and finishing $\mathbf{R}_v$. If $\mathbf{R}_v$ can not be constructed or finished before the end of the simulation period $\mathbf{T}$, a maximum value of time with penalty will be provided to mark its unacceptability. In other words, the travel time of v can be calculated at the end of the simulation by

$$t_v = \begin{cases} \delta \times |\mathbf{T}|, & \text{if } T^{\text{des}_v} \text{ is invalid} \\ T^{\text{des}_v} - T^{\text{ori}_v}, & \text{otherwise} \end{cases}, \tag{15}$$

where $\delta \geq 1$ is a penalty coefficient.

In summary, the fitness function of TSC based on vehicles can be expressed by

$$f(\mathbf{s}) = \frac{\sum_v t_v}{|\mathbf{V}|}. \tag{16}$$

3.4 | Model of traffic assignment: vDUE

The traffic assignment is usually used to provide the fitness value of a TSC solution, by considering the interactions among the link flow amounts, the link travel time, and the vehicle routes. On the one hand, the link flow amount is an essential element to calculate the link travel time. On the other hand, the link travel time has an important impact on vehicles to construct and finish their routes, and it hence possibly causes the

inverse change of the link flow amounts. This subsection aims to describe the model of the vehicle-based Dynamic Use Equilibrium (vDUE), where virtual vehicles generated according to the flow demands of the transportation network.

Definition 10 (Link Flow Amount). The flow amount of a link l at a time slot T is denoted as f_l^T and directly related with the number of vehicles on the link, as expressed by

$$f_l^T = K \times \sum_{o,d \in \mathbf{N}} \sum_{v \in \mathbf{V}_{o,d}} y_{v,l}^T, \tag{17}$$

where K is the same as in Equation (14). Besides, $y_{v,l}^T = 1$ if the vehicle v is traveling on the link l at the time slot T, and $y_{v,l}^T = 0$ otherwise.

Definition 11 (Link Travel Time). The link travel time t_l^T refers to, on average, how soon vehicles can finish traveling on the link l at a time slot T. This paper calculates t_l^T by following the link performance function proposed by the Bureau of Public Roads(BPR function)[27], as expressed by

$$t_l^T = t_l^0 \times \left[1 + 0.15 \times \left(\frac{f_l^T}{f_l^{\max}} \right)^4 \right], \forall l \in \mathbf{L}. \tag{18}$$

Definition 12 (Vehicle Route). The vehicle route $\mathbf{R}_v$ of the vehicle v is comprised of all the nodes and links which v has traveled from the origin O_v to the destination D_v during the simulation of the traffic assignment. As shown in Equation (16), the travel time of each vehicle has an influence on the fitness function of the TSC solution. Hence, the vehicle route plays a key role in solving the TSC problem, and especially in simulating the traffic assignment.

Definition 13 (Vehicle-Based Dynamic User Equilibrium). Since travelers non-cooperatively choose vehicle routes to minimize their own traveling costs (e.g. the travel time), the traffic assignment on a network is expected to reach a user equilibrium state when no travelers can reduce their traveling costs by shifting to other alternative routes [13]. Based on virtual vehicles generated according to the flow demands, we build a vDUE model for simulating the traffic assignment, as expressed by

$$\min z(f) = \sum_{l \in \mathbf{L}} \int_0^{f_l^T} t_l^T(f) df, \tag{19}$$

s.t.

$$f_l^T = K \times \sum_{o,d \in \mathbf{N}} \sum_{v \in \mathbf{V}_{o,d}} y_{v,l}^T, \forall l \in \mathbf{L}, \tag{20}$$

$$F_{o,d} = K \times |\mathbf{V}_{o,d}|, \forall o, d \in \mathbf{N} \tag{21}$$

$$|\mathbf{V}_{o,d}| \geq 0. \tag{22}$$

As shown in Equation (19), $z(f)$ is the objective function of the DUE traffic assignment, and $t_l^T(f)$ is a cost function like Equation (18) to calculate the link travel time according to the link flow amount f_l^T. Equation (20) makes the first constraint that the link flow demand should be dynamically proportional to the number of the vehicles traveling on the link, no matter what OD pairs these vehicles have. Meanwhile, Equation (21) means that the flow demand of an arbitrary OD pair (o, d) should be proportional to the number of the vehicles with the origin as o and the destination as d. At last, Equation (22) means that the number of the vehicles generated for each OD pair should not be smaller than zero.

4 | SYSTEM AND ALGORITHMS

To effectively solve the bi-level TSC problem formulated above, we build an agile TSC system by specially proposing an agile method for the vDUE model and adopting an adaptive differential evolution algorithm for the signal timing optimization. This section is going to introduce the system framework and detailed implementations of the related algorithms.

4.1 | System framework

Within a bi-level TSC system, there is a strong interaction between the upper level control model and the lower level assignment model. On one hand, the upper level determines the signal timings of all signalized intersections to perform the traffic assignment. On the other, the lower level provides a feedback to help optimize the signal control.

As shown in Figure 2, our proposed agile TSC system is comprised of two levels corresponding to the bi-level structure of the TSC problem. The upper level adopts an adaptive differential evolution algorithm [13] that evolves a population of TSC solutions, each of which is comprised of the signal timings (i.e. the offset, the cycle length, and all phase green ratios) of all considered intersections according to Equation (1). Meanwhile, the lower level executes our proposed agile method for the vDUE traffic assignment, where virtual vehicles are generated based on the flow demands according to Equation (14). The two levels exchange data mainly through the evaluation of each TSC solution, where the upper level provides the signal timings of all signalized intersections within the solution and the lower level returns the simulated travel time of all vehicles to calculate fitness value of the solution. At last, the best-so-far TSC solution is output from the upper level as the final result of the agile TSC system.

At the upper level, the adaptive differential evolution is expected to gain a high effectiveness to solve the signal timing optimization. As we know, differential evolution is a famous stochastic method for global optimization over continuous spaces, with advantages of easier implementation, fewer control parameters, faster convergence than many other acclaimed global optimization methods [28]. Meanwhile, the signal timing optimization in our TSC model is non-convex and rather
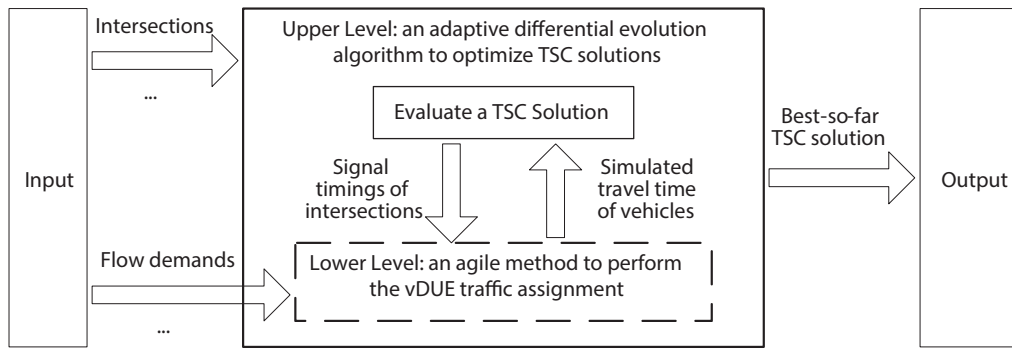
FIGURE 2 Framework of our agile TSC system

suitable for an evolutionary algorithm to solve. As a result, the differential evolution has a natural superiority to optimize TSC solutions with a complicated continuous structure as in Equations (1)–(5). Moreover, the differential evolution has a lower space complexity than some of the most competitive real parameter optimization methods, which facilitates its extension to solve large scale TSC problems [29].

At the lower level, the proposed agile method for the vDUE model is expected to efficiently simulate the practical traffic dynamics. For the sake of the computational simplicity, the traffic flow is used to describe the status of the transportation network. Meanwhile, virtual vehicles are generated to reflect the practical dynamics through their flexible routing/rerouting during the traffic assignment process. Though starting vehicles are always considered for routing, a traveling vehicle may also need to be rerouted if it finds its previously planned route not suitable to continue due to the high traffic dynamism in practice. As a result, our agile method should greatly help the traffic network to approach the dynamic user equilibrium state, by considering both starting and traveling vehicles for routing.

Next we will introduce the two levels in details.

4.2 | Adaptive differential evolution to optimize TSC solutions

To accomplish an optimization task, the differential evolution encodes a possible solution into a vector of variables and iteratively carries a series of evolvable operations on a population of solutions **S**. Theses operations traditionally include mutation, crossover and selection, and they are sequentially performed on each solution. Particularly, the mutation generates a mutant $\mathbf{s}'$ for each solution **s**, by adding some form of differential variation to an existing solution. The amplification of the differential variation is controlled by a real constant, denoted as M in this paper. To increase the diversity of the population, the crossover is performed on each pair of a solution **s** and its mutant $\mathbf{s}'$. The outcome of the crossover is a trial variant $\mathbf{s}''$, whose variables come from either **s** or $\mathbf{s}'$. The probability to set a variable of $\mathbf{s}''$ according to either **s** or $\mathbf{s}'$ is mainly controlled by another real constant, denoted as X in this paper. Finally, the selection aims to determine the members of a new population for a next
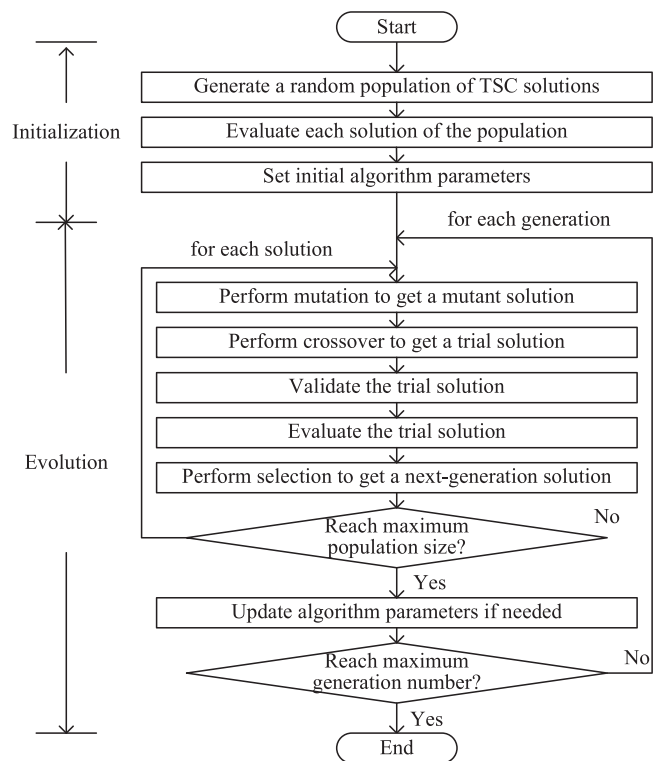

FIGURE 3 Adaptive differential evolution to optimize TSC solutions

generation (i.e. iteration). After evaluated, each trial variant $\mathbf{s}''$ is compared to the original solution **s** and the winner is selected into the new population.

This paper adopts an adaptive differential evolution to optimize the TSC solutions, by integrating with a greedy mutation strategy to accelerate the convergence and a parameter adaptation scheme to improve the robustness [13]. As described in Figure 3, the adaptive differential evolution mainly contains two parts and their details are given as below.

In the initialization part, a population of solutions $\mathbf{S} = \{\mathbf{s}\}$ are firstly generated, by randomly generating values of the cycle length rate, the offset time ratio, and the green time ratio within the ranges of $[0, 1]$, $[o_{\min}, o_{\max}]$, $[g_{\min}, g_{\max}]$, respectively. Then, each newly generated solution is evaluated by calling the agile

method to perform the vDUE traffic assignment. Moreover, since archives are commonly used in adaptive evolutionary algorithms like [30], an empty external archive $\mathbf{S}^{\cup}$ is set for the greedy mutation to gain a promising progress direction from recently explored inferior solutions. As for the parameter adaptation, the initial values of M and X for each solution are randomly generated within [0,1] and stored in two sets $\mathbf{M}$ and $\mathbf{X}$, respectively. Correspondingly, the initial mean values μ_M and μ_X are both set as 0.5 for $\mathbf{M}$ and $\mathbf{X}$, respectively.

In the evolution part, each solution of the population is updated in every generation through a series of operations, including mutation, crossover, validation, evaluation, and selection. The crossover and the selection follow strategies of the traditional differential evolution, whilst the evaluation is implemented at the lower level. As for the validation, each solution is guaranteed to have variable values of three kinds respectively within their pre-defined ranges, by setting invalid variable values to be the nearest bound. Meanwhile, the mutation uses the "current-to-pBest" strategy as expressed by

$$\mathbf{s}' = \mathbf{s} + M \times \left(\mathbf{s}^{p\text{Best}} - \mathbf{s} \right) + M \times \left(\mathbf{s}_{r1} - \mathbf{s}_{r2}^{\cup} \right), \quad (23)$$

where $\mathbf{s}$ and $\mathbf{s}'$ denote the target solution in $\mathbf{S}$ and its mutant variant, and M is the mutation factor. $\mathbf{s}^{p\text{Best}}$ is randomly one of the top $100p\%$ solutions in $\mathbf{S}$ with $p \in (0, 1]$, whilst $\mathbf{s}_{r1}$ is another solution different from $\mathbf{s}$ in $\mathbf{S}$. Meanwhile, $\mathbf{s}_{r2}^{\cup}$ is selected from $\mathbf{S} \cup \mathbf{S}'$, and it is different from $\mathbf{s}$ and $\mathbf{s}_{r1}$.

In the traditional differential evolution, the mutation factor M and the crossover factor X usually have great impacts on the algorithm performance and need some empirical knowledge to set appropriate values for different application cases. For the sake of the high efficiency and the extensive applicability, two algorithm parameters M and X are adapted at the end of every generation by following

$$M = randc(\mu_M, 0.1), \quad (24)$$

$$X = randn(\mu_X, 0.1), \quad (25)$$

where $randc(\mu_M, 0.1)$ means to generate a random number according to a Cauchy distribution with mean μ_M and standard deviation 0.1; $randn(\mu_X, 0.1)$ means to generate a random number according to a normal distribution with mean μ_X and standard deviation 0.1.

The values of μ_M and μ_X are initialized to 0.5 and updated as

$$\mu_M = (1 - b) \times \mu_M + b \times mean_L(\mathbf{M}), \quad (26)$$

$$\mu_X = (1 - b) \times \mu_X + b \times mean_A(\mathbf{X}), \quad (27)$$

where b is a positive constant within [0,1], whilst $mean_L(\mathbf{M})$ and $mean_A(\mathbf{X})$ represent the Lehmer mean function and the usual arithmetic mean function, respectively. As for $\mathbf{M}$ and $\mathbf{X}$,
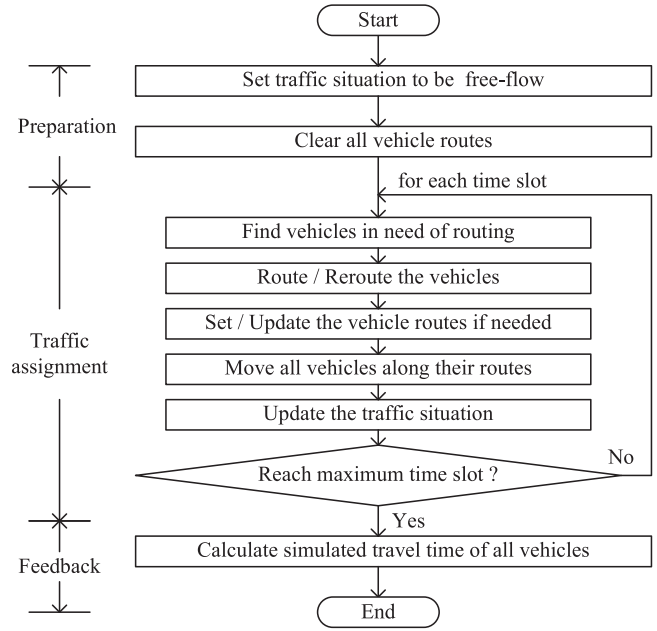

FIGURE 4　Agile method to perform vDUE traffic assignment

they contain the values of M and X that successfully bring improved solutions.

4.3 | Agile method to perform vDUE traffic assignment

When performing the traffic assignment, the traffic situation is generally updated at each time slot. Thus, the vehicle route constructed before may be measured differently later. It means that a traveling vehicle may update its scheduled route to reduce its travel cost, especially when the saturation of its remained route increases.

As a result, this paper proposes an agile method for the vDUE traffic assignment of the bi-level TSC problem, which not only routes the starting vehicles but also reroutes the traveling vehicles if needed. As shown in Figure 4, the lower level consists of three stages. The first stage is to prepare the free-flow traffic situation and the ready vehicles, and the last stage is to provide the upper level with the simulated travel time of all vehicles to calculate the fitness value of the TSC solution. Between the two stages, the traffic assignment acts as the main body, where the vehicle routes and the traffic situation are iteratively updated during a period of time slots, by following five steps.

Step 1 (Determine the vehicles in need of routing): At each time slot, our proposed method checks the status of all vehicles to determine the set of vehicles which need routing. More specifically, the starting vehicles will all be considered in need of routing, whilst a traveling vehicle v may be considered in need at the time slot T only if the saturation of its remained route θ_v^T exceeds a pre-defined threshold $\theta^{\max}$. We heuristically set the probability
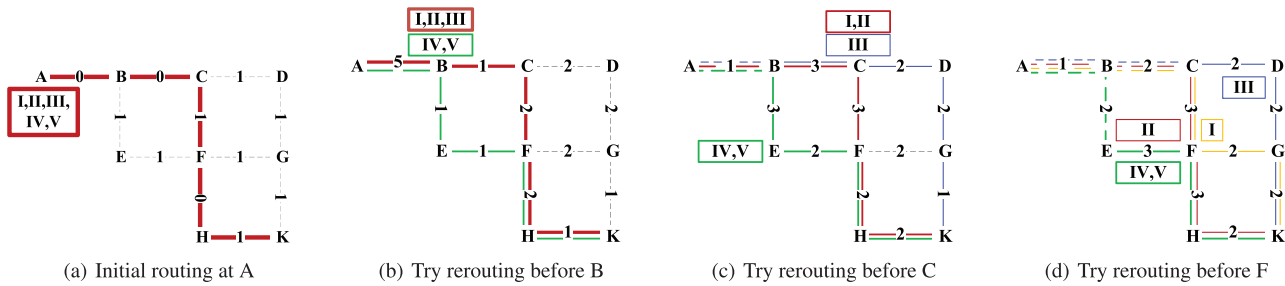
FIGURE 5 Example of the agile method to perform the vDUE traffic assignment

of the traveling vehicle v at the time slot T as

$$\rho_v^T = \begin{cases} 0, & \text{if } \theta_v^T \leq \theta^{\max} \\ \theta_v^T \times \theta^{\max}, & \text{otherwise} \end{cases}. \quad (28)$$

Considering the high dynamics of the traffic flow, we approximate θ_v^T as the average saturation of the first two links along the remained route at the time slot T, as calculated by

$$\theta_v^T = \left(\frac{f_{R_{v,1}^*}^T}{f_{R_{v,1}^*}^{\max}} + \frac{f_{R_{v,2}^*}^T}{f_{R_{v,2}^*}^{\max}} \right)/2, \quad (29)$$

where $R_{v,1}^*$ and $R_{v,2}^*$ denote the first two links along the remained route of v, respectively.

Step 2 (Route / Reroute the vehicles in need): Since the lower level is embedded in the upper level and frequently called to evaluate each TSC solution, it is highly desirable to reduce the computational cost of the vDUE traffic assignment and finally to improve the efficiency of the whole TSC system. Hence for the sake of an efficient routing, we simply build a Dijkastra shortest path for each vehicle from its current node to its destination. Note that the shortest path is measured with the link travel time instead of the link length. Besides, the current node of a starting vehicle refers to its origin, whilst that of a traveling vehicle refers the end node of the link it is traveling on.

Step 3 (Set / Update the vehicle routes if possible): This step decides whether to use the newly built shortest paths for different types of the vehicles. For a starting vehicle whose planned route is empty, this new path will be definitely adopted. For a traveling vehicle whose planned route has been started, this new path will take effect only if it is better than the unfinished part of the original vehicle route.

Since each vehicle pursues its own cost minimization, a new path will be considered better if it can reduce the remained travel cost for the related vehicle. For the sake of simplicity, we considered the travel cost of a vehicle mainly with the vehicle travel time, which can be approximated as the sum travel time of all links along its route. Hence for a traveling vehicle, a new path will replace the unfinished part of the original route only if the former has smaller sum travel time of all its links than the latter.

Step 4 (Move all vehicles along their routes): This step moves all virtual vehicles forward along their routes by simulating their pos-

sible states, including 'ready', 'moving', 'waiting', and 'finished', whose details will be described in Section 4.4.

Step 5 (Update the traffic situation): The traffic situation at a time slot is mainly dependent on the link flow amounts and the link travel time, which are updated according to Equations (17) and (18), respectively.

For a better understanding of the agile method, Figure 5 presents an example to perform the vDUE traffic assignment on the OD pair of (A,K), where each node is denoted by a capital letter and each link uses an integer to represent the number of vehicles traveling on it. For the sake of simplicity, we assume all links of the example network have the same free-flow travel time and the same flow capacity of 5, so the travel time of two links should be equal if they have the same number of vehicles traveling on it according to Equation (18). Besides, the threshold of the route saturation is set as $\theta^{\max} = 0.5$. Note that the waiting time of vehicles at signalized intersections are not considered in this example, to better focus on illustrating the rerouting idea of the agile method.

Initially in Figure 5(a), there are five vehicles {I,II,III,IV,V} starting from A, whose routes are all set to the shortest path marked in red $\mathbf{R}_r = A \rightarrow B \rightarrow C \rightarrow F \rightarrow H \rightarrow K$. Next in Figure 5(b), the five vehicles consider rerouting before they arrive at B, since the traffic flows have changed after a time period. At this time, the first two links of the remained $\mathbf{R}_r$ are $A \rightarrow B$ and $B \rightarrow C$, so the approximated saturation of the remained $\mathbf{R}_r$ is calculated as 0.6 according to Equation (29). Subsequently, the rerouting probability of each vehicle in {I,II,III,IV,V} is calculated as 0.3 according to Equation (28). Suppose that two vehicles {IV,V} are rerouted to the current shortest path marked in green $\mathbf{R}_g = A \rightarrow B \rightarrow E \rightarrow F \rightarrow H \rightarrow K$. Later in Figure 5(c), vehicles {I,II,III} consider rerouting before C, and their rerouting probability is calculated to be 0.3 with the approximated saturation of the remained $\mathbf{R}_r$ as 0.6. Suppose that one vehicle {III} is rerouted to the current shortest path marked in blue $\mathbf{R}_b = A \rightarrow B \rightarrow C \rightarrow D \rightarrow G \rightarrow K$. Then in Figure 5(d), vehicles {I,II} and {IV,V} consider rerouting before F and all get a rerouting probability of 0.3. Suppose that one vehicle {I} is rerouted to the current shortest path marked in yellow $\mathbf{R}_y = A \rightarrow B \rightarrow C \rightarrow F \rightarrow G \rightarrow K$. Finally the five vehicles {I,II,III,IV,V} with the same OD pair (A,K) are assigned to four different routes, that is, $\mathbf{R}_r$, $\mathbf{R}_g$, $\mathbf{R}_b$, and $\mathbf{R}_y$, which helps the network traffic gradually reach an equilibrium state.
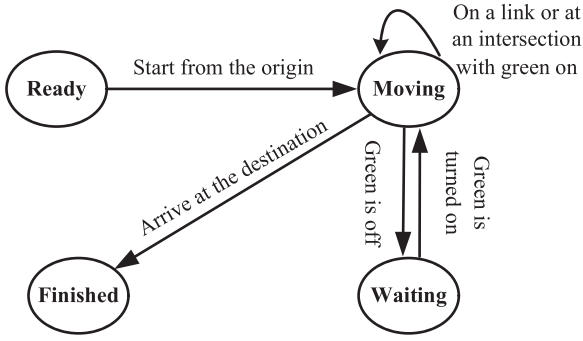
FIGURE 6 Status transition of a vehicle

4.4 | Movement simulation of virtual vehicles

From the perspective of a vehicle v, the goal of performing the traffic assignment is to successfully construct a vehicle route $\mathbf{R}_v$ and to obtain the simulated travel time t_v. As shown in Equation (15), the key of calculating t_v is T^{des_v}, which is obtained by periodically tracing the movement of the vehicle v along $\mathbf{R}_v$ until reaching the destination D_v.

To simulate the movement of each vehicle along its route, we define four types of vehicle statuses and illustrate their possible transitions in Figure 6.

(1) The first status is 'ready', which refers to a vehicle at the origin. Note that the planned starting time slots of most ready vehicles are later than the current time slot.

(2) The second status is 'moving', which refers to a vehicle moving on a link along its route. To simplify the vehicle model in the traffic dynamism, we assume that all traveling vehicles on a link l are formed into the link flow f_l and the related link travel time t_l^T can be calculated by Equation (18) at the current time slot T. As a result, the travel time needed by a traveling vehicle v to finish the link l can be calculated by

$$t_{l,v}^T = t_l^T \times \frac{d_{l,v}^T}{d_l}, \qquad (30)$$

where d_l denotes the length of the link l. Meanwhile, $d_{l,v}^T$ denotes the remained length for v at T to finish on l, and it is updated and recorded from the origin to the destination.

The vehicle will finish moving on l and change its status to either 'waiting' or 'finished', if $t_{l,v}^T < 1$. Otherwise, it should remain to be 'moving' on l and obtain $d_{l,v}^{T+1}$ by

$$d_{l,v}^{T+1} = d_{l,v}^T - \frac{d_l}{t_l^T}. \qquad (31)$$

(3) The third status is 'waiting', which means a vehicle is waiting for its green time at an intersection. This status is most possibly changed from the 'moving' status. When a vehicle v finishes a link and reaches an intersection n at the time slot T, the waiting time $\tau_{n,k}^T$ needed by the vehicle v for the kth signal phase of the intersection n can be calculated by Equation (32) with T^{test}

calculated by Equation (33).

$$\tau_{n,k}^T = \begin{cases} 0, \text{ if } T^{\text{test}} + \sum_{i=1}^{k-1} G_{n,i} \leqslant T \leqslant T^{\text{test}} + \sum_{i=1}^{k} G_{n,i} \\ \sum_{i=1}^{k-1} G_{n,i} - T^{\text{test}}, \text{ if } T < T^{\text{test}} + \sum_{i=1}^{k-1} G_{n,i} \\ \sum_{i=1}^{k-1} G_{n,i} + C_n - T^{\text{test}}, \text{ otherwise} \end{cases}, \qquad (32)$$

$$T^{\text{test}} = O_n + C_n \times \left\lfloor \frac{T - O_n}{C_n} \right\rfloor, \qquad (33)$$

where O_n and C_n are the offset and cycle length of the intersection n, whilst $G_{n,i}$ represents the green time of the ith phase of n.

(4) The last status is 'finished', which means a vehicle arriving at the destination. This status should be all changed from the 'moving' status, when a vehicle finishes the last link of its route. In this case, the vehicle v will record the current time slot T as its ending time slot T^{des_v}.

Based on the movement simulation of virtual vehicles, the agile method brings at least two advantages to the vDUE traffic assignment. One is providing the traveling vehicles with a possibility to reroute, which makes the simulation closer to both the traffic dynamism in reality and Wardrop's user equilibrium principle. The other is calculating the rerouting probability of a traveling vehicle by approximating the saturation of its remained route according to Equations (28) and (29), which improves the efficiency to deal with different traffic situations (especially the crowded ones and those with emergencies).

5 | EXPERIMENTS

In this section, we conduct experimental comparisons on different transportation networks to validate the effectiveness of both our proposed agile vDUE mechanism and the related agile TSC system. We are going to describe the test data, experimental setup, comparisons on both DUE and TSC.

5.1 | Test data

As depicted in Figure 7, three transportation networks are used in the experiments. The real-world network SiouxFalls consists of 24 nodes and 76 links, with all patterns of signalized intersections as in Figure 1. The data of this network is downloaded from a famous repository of transportation networks at [25]. Besides, two synthetic networks TN01 and TN02 are randomly generated in this paper based on a grid network. Specifically, TN01 is comprised by 14 nodes and 30 links, whilst TN02 contains 30 nodes and 70 links.

To observe influence of the traffic pressure, we provide each test network with three types of traffic situations as listed in Table 1. The first type is called 'normal' with a low flow demand, whilst the second type is called 'crowded' with a high flow
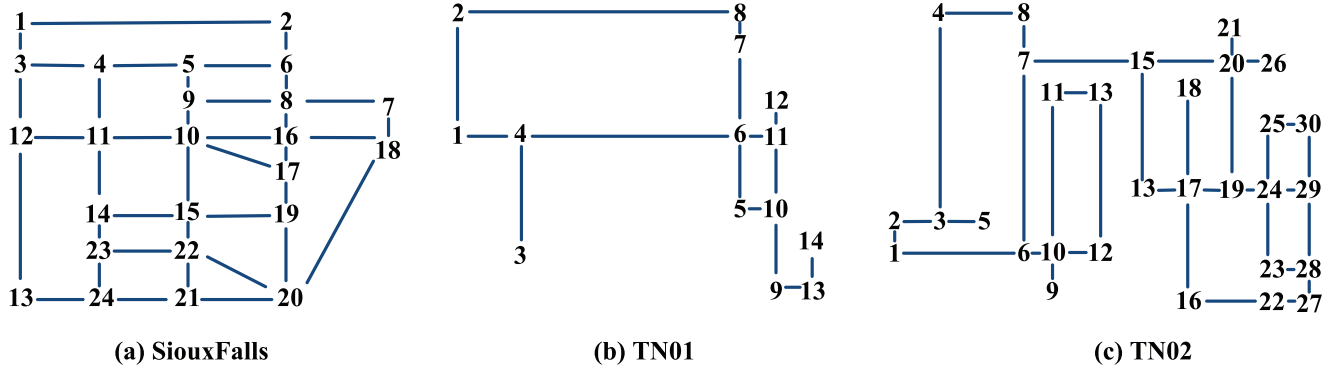
FIGURE 7 Transportation networks for test

TABLE 1 Three types of traffic situations of all test networks

Network	Traffic situation	Normal	Crowded	Severe
SiouxFalls	Flow demand (in vehicle hour)	360,600	766,400	766,400
	Number of virtual vehicles	3606	7664	7664
	Number of emergencies	0	0	2
TN01	Flow demand (in vehicle hour)	2224	8640	8640
	Number of virtual vehicles	2224	8640	8640
	Number of emergencies	0	0	3
TN02	Flow demand (in vehicle hour)	5932	22,188	22,188
	Number of virtual vehicles	5932	22,188	22,188
	Number of emergencies	0	0	4

TABLE 2 Emergencies assumed in the severe traffic situations

Network	Link	Start time	End time
SiouxFalls	13 → 24	15	35
	16 → 17	30	50
TN01	3 → 4	20	45
	6 → 7	30	55
	8 → 2	40	60
TN02	3 → 4	20	45
	6 → 7	30	55
	13 → 12	30	60
	19 → 20	40	60

demand. For the traffic simulation of each test network, vehicles are generated for each type according to the related flow demand. Considering the randomness in reality, we make the starting time of the vehicles for the traffic simulation follow a Poisson distribution. However, no emergency is assumed to happen in the simulation for either normal or crowded types. Finally, the third type is called 'severe' with the same flow demand as the Crowded. Moreover, emergencies are assumed to happen in the simulation of each test network for this type, as listed in Table 2. For the severe traffic situation of each network, the emergencies are assumed to happen on different links during different time slots, which make the flow capacity half on the related links.

5.2 | Experimental setup

The experimental comparisons in this section are conducted on both DUE and TSC. On DUE, we compare our agile method with the classic All-Or-Nothing (AON) method [31] and a revised Method of Successive Average (rMSA) [3]. AON has almost the lowest computational cost on the traffic assignment due to its simplicity, so it can act as a benchmark to evaluate other traffic assignment methods in terms of the running time. As for rMSA, it is a representative strategy for the traditional stochastic traffic assignment and has been successfully used to solve the dynamic traffic assignment of a bi-level TSC system. Therefore, rMSA can be used as a state-of-the-art counterpart to better check the performance of our proposed agile method that tries to implement the stochastics and dynamics of the traffic assignment in a more flexible way.

For the sake of fairness, we integrate each compared DUE method with a simple fixed-time signal timing method to solve the TSC problem. The fixed-time method forms a TSC solution just by picking up the median values for all decision variables within their valid ranges. Meanwhile, on TSC, our proposed agile TSC system is compared with two other TSC methods. The first is a classic TSC method, where the fixed-time signal timing method and AON are used [31]. The second is a state-of-the-art TSC method proposed by Li et al. [3], where a hybrid genetic algorithm is used to optimize the signal timings and the rMSA is used for the traffic assignment. Note that the microscopic traffic simulation of Li's TSC is replaced by calculating Equation (18) in this paper.

The solution measurements used here include objective, fitness, and running time. Among them, the objective value refers to the average vehicle travel time in a TSC solution, and it is the major indication of the solution quality. Meanwhile, the fitness value is a variant of the objective value as calculated by Equations (15) and (16), and it is expected to improve the solution quality. Finally, the running time indicates the time cost to run a TSC method on computers.

TABLE 3 Parameters of the proposed agile TSC system

Component	Notation	Value	Meaning
Model	$[c_{min}, c_{max}]$	[0, 1]	Valid range of a cycle length rate
	$[o_{min}, o_{max}]$	[0, 1]	Valid range of an offset time ratio
	$[g_{min}, g_{max}]$	[0.1, 1]	Valid range of a green time ratio
ADE	δ	5	Penalty coefficient in Equation (15)
	P^{ADE}	10	Size of the population
	G^{ADE}	50	Number of generations
	p	0.3	Mutation parameter in Equation (23)
	b	0.1	Adaptive parameter in Equations (26) and (27)
TA	θ^{max}	0.5	Saturation threshold in Equation (28)

When solving the TSC problem modeled as in Section 3, our proposed agile TSC adopts adaptive mutation and crossover factors, with other parameters listed in Table 3. Meanwhile, Li's TSC adopts the same population size and the maximum generation number as ours, with other parameters set as in [3]. For the sake of comparison fairness, each TSC method runs for 30 times independently in each test case. All compared methods are coded in C++ and executed on a computer with Intel(R) Core(TM) i7-8700 @ 3.20GHz CPU, 16 GB memory and Microsoft Windows 10 64-bit system.

5.3 | DUE comparison

As the traffic assignment plays a key role in the bi-level TSC system, this subsection carries out comparison experiments among AON, rMSA and our agile method, mainly on the overall performance of the traffic assignment and the saturation change during the traffic simulation.

5.3.1 | Overall performance

The overall performances of the three DUE methods in all test cases are compared in terms of objective value and running time.

Figure 8 compares the average objective values of three DUE methods in different traffic situations. Overall, the bins of three DUE methods become higher from the normal to the severe traffic situation on each network, which indicates that the problem difficulty increases as the traffic situation intensifies. In the normal situation, the bins of three DUE methods almost have the same height on each network. However in the crowded and severe situations, the bins of our agile method are obviously lower than those of AON and rMSA on each network. This hence validates the effectiveness of our agile method in various traffic situations (especially in the crowded ones and those with emergencies).

Table 4 compares the absolute running time of the three DUE methods, along with the relative incremental percentages of the rMSA method and our agile method over the AON method in brackets. Note that the running times listed in this table are mainly for the traffic assignment process, where the three compared DUE methods all adopt a simple fixed-time signal timing method. In respects of the incremental percentage, we notice that rMSA and our agile method always have larger running time costs than the AON method with the positive increments in all test cases, most probably due to their stochastic. Besides, our agile method always has smaller running time increments than rMSA in the normal and crowded situations. In the severe situation, our larger running time increments seem still worthy by outperforming the rMSA method so much as shown in Figure 8. As for the absolute value, we notice that the running time of each DUE method obviously increases as the traffic situation intensifies, which is compliant with the increase of the problem difficulty. Note that the experiments are currently performed on a single core of a personal computer, and they can be further accelerated by code optimization and parallelization techniques. As a result, the running time of our agile method is generally acceptable for the dynamic traffic assignment.

5.3.2 | Saturation change

To look into the inherent difference of the three DUE methods, we make a further analysis on the saturation change of the links on a transportation network.

Figure 9 illustrates the average saturation of all links on the SiouxFalls network during the simulation. As we can see, the saturation curves of the three DUE methods are overlapped

(a) SiouxFalls   (b) TN01   (c) TN02
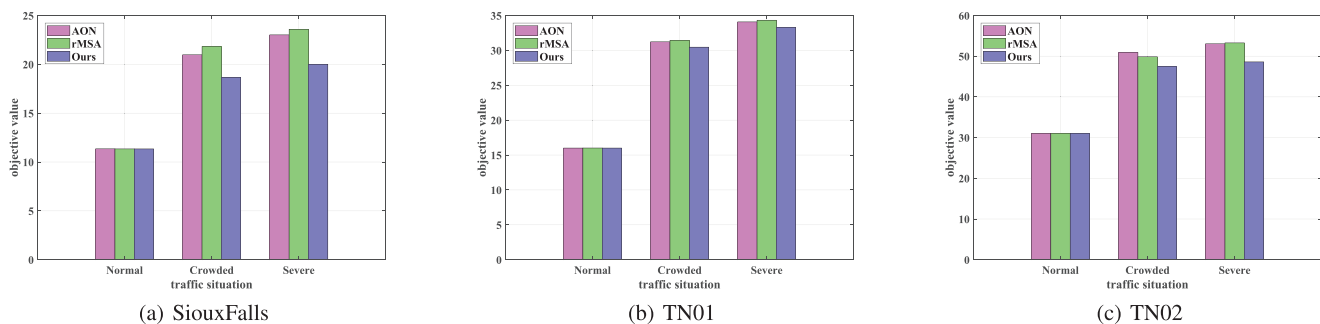
FIGURE 8 Performance comparison of three DUE methods in terms of the objective value

TABLE 4 Performance comparison of three DUE methods in terms of the running time (in second)

	Normal			Crowded			Severe		
	AON	rMSA	Ours	AON	rMSA	Ours	AON	rMSA	Ours
SiouxFalls	0.18	0.30 (65%)	0.22 (21%)	0.38	0.67 (75%)	0.50 (31%)	0.42	0.71 (67%)	0.86 (103%)
TN01	0.12	0.18 (52%)	0.13 (9%)	0.33	0.70 (108%)	0.44 (32%)	0.37	0.76 (107.%)	1.35 (267%)
TN02	0.45	0.71 (56%)	0.51 (12%)	1.42	2.74 (93%)	2.13 (50%)	1.54	2.43 (58%)	10.32 (572%)

(a) Normal   (b) Crowded   (c) Severe
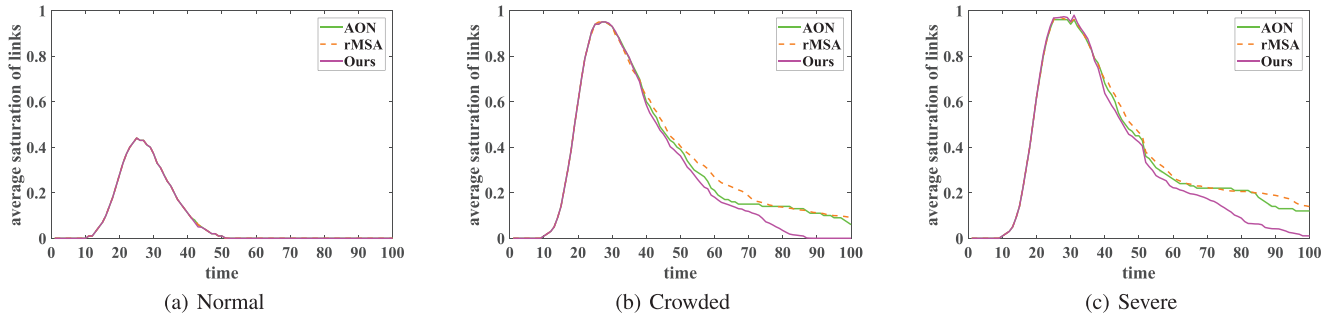
FIGURE 9 Comparing average saturation of all links on SiouxFalls network for three DUE methods

before the peak around the 30*th* time slot in all three traffic situations. However, the peak in the normal situation is almost half of those in the crowded and severe saturations, which indicates that the problem difficulty increases as the traffic situation intensifies. In the normal situation, the three curves continue to overlap after the peak, because our agile method considers nearly no traveling vehicle in need of routing with the peak smaller than $\theta_{max} = 0.5$. However in the crowded and severe situations, our curve obviously declines faster than the other two curves after the peak, because there are quite a few traveling vehicles considered in need of routing with the peaks far larger than θ_{max}. In a word, our proposed agile method performs better than AON and rMSA to decrease the average saturation of all links by making full use of the saturation approximation.
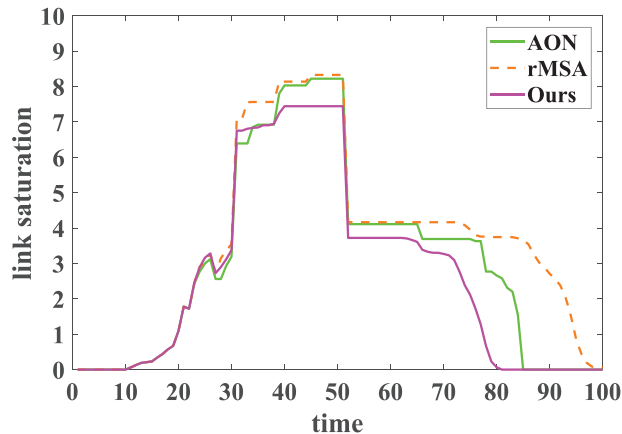

FIGURE 10 Analyzing saturation of a link with emergence on SiouxFalls network in the severe situation for three DUE methods

Figure 10 further illustrates the saturation change of a typical link 16 → 17 with emergency on SiouxFalls network. As we can see, the three DUE methods all have increased the saturation of this link during the period of emergency [30], [50]. Nevertheless, the saturation curve of our agile method is obviously lower than those of its counterparts during the emergency period. That is mainly because our agile method continuously traces the movement of each vehicle and reroutes a traveling vehicle according to the approximated saturation of its remained route. Since the intersection 16 is a cross-shaped intersection in Figure 7(a), the vehicles can be routed to other alternative links in time before they enter the emergence link 16 → 17. Meanwhile, AON and rMSA both determine the routes for vehicles mainly at their origins, so they possibly need more time to respond to the sudden saturation change of a link with emergency. In a word, our agile method is able to make a quicker response to the sudden saturation change of a link with emergence than AON and rMSA, by adopting a more flexible way to implement the dynamism and stochastic during the traffic assignment.

In summary, the effectiveness of our proposed agile method has been validated with an acceptable running time through the overall performance comparison with AON and rMSA. Besides, our agile method has also proven the significance of probabilistically rerouting traveling vehicles according to the approximated saturation of their remained routes through the analysis on the saturation change.

5.4 | TSC comparison

This subsection compares three TSC methods in two aspects. First, we compare the quality of their average solutions on all test networks.
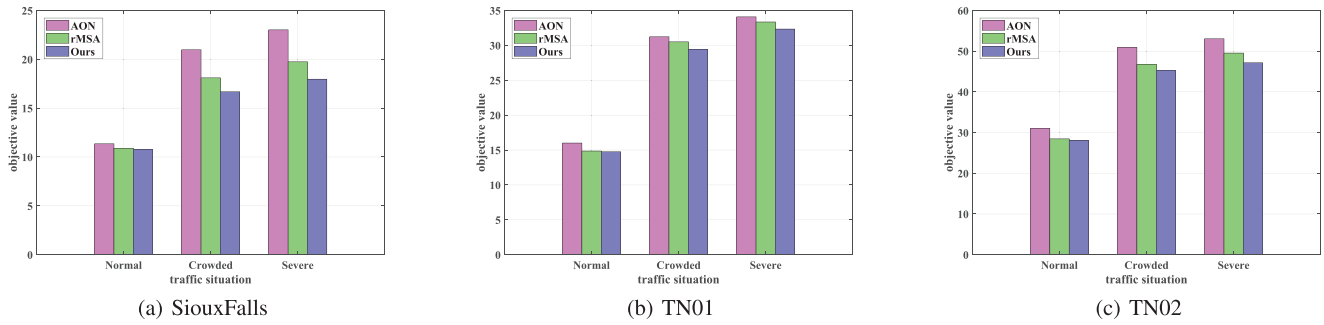
FIGURE 11 Performance comparison of three TSC methods in terms of objective value
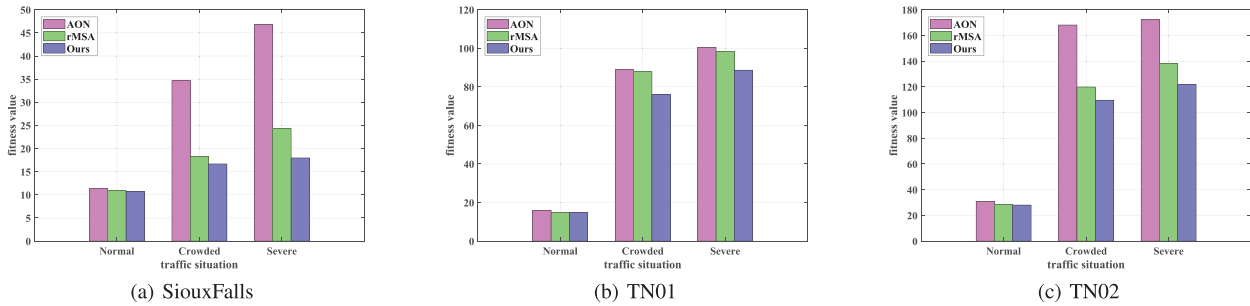

FIGURE 12 Performance comparison of three TSC methods in terms of fitness value

5.4.1 | Overall performance

The average solutions of the three TSC methods on all three networks are compared in terms of both objective and fitness values. We notice three following points from the overall result comparison.

Figure 11 compares average objective values of three TSC methods in three traffic situations. In each situation of each network, the bin heights of three TSC methods decrease in the order of the classic, Li's, and ours, which indicates that ours outperforms Li's and they both outperform the classic. Meanwhile, Figure 12 compares average fitness values of three TSC methods in three traffic situations. The fitness values calculated according to Equation (16) are proven effective to solve the TSC problem, because the fitness curves of the three TSC methods have similar trends but greater extents in comparison with the relative objective bins on all three networks.

Table 5 shows the objective decrements of ours over its two counterparts, to quantify the improvement of our agile TSC. In

respects of compared algorithm, the objective decrement always has a larger positive value over the classic than over Li's, which is consistent to the curve positions as shown in Figure 11. In respects of test network, the objective decrement has the lowest values on TN01 with the smallest flow demands and the simplest network structure, and the highest value on SiouxFalls network with the largest flow demands but the fewest virtual vehicles. In respects of traffic situation, the objective decrement generally has a higher value as the situation intensifies, which indicates that our agile TSC has a more obvious advantage in crowded and severe situations. Overall, our agile TSC outperforms its two counterparts by obtaining the average travel time of all vehicles decreased up to 21.74% in various traffic situations on all test networks.

5.4.2 | Algorithm convergence

Figures 13–15 compare the convergence curves of Li's and our TSC methods on all test cases, in terms of the fitness value. We

TABLE 5 Objective decrements of our agile TSC over the classic TSC and Li's TSC

	Normal		Crowded		Severe	
	Ours versus classic	Ours versus Li's	Ours versus classic	Ours versus Li's	Ours versus classic	Ours versus Li's
SiouxFalls	5.26%	0.92%	20.48%	7.73%	21.74%	8.63%
TN01	7.50%	0.67%	5.77%	3.61%	5.28%	3.00%
TN02	9.35%	1.40%	11.20%	3.42%	11.30%	4.85%

(a) SiouxFalls   (b) TN01   (c) TN02
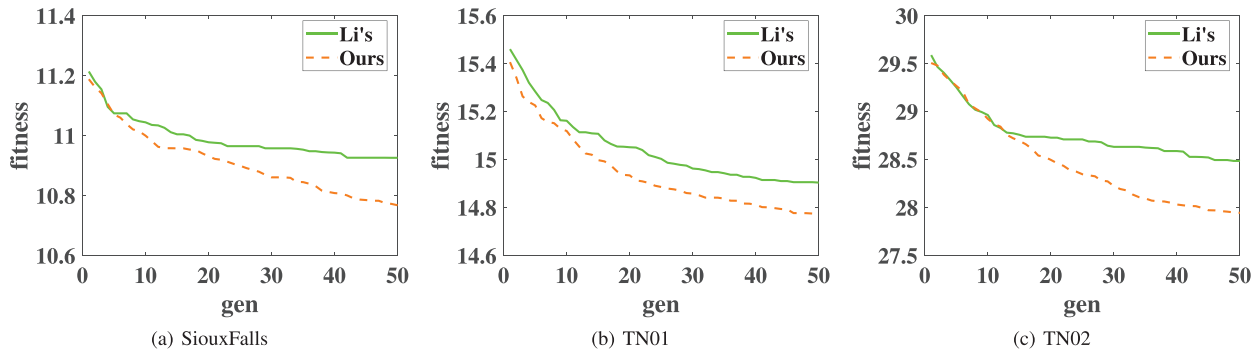
FIGURE 13   Convergence comparison of Li's and our TSC methods in the normal traffic situation
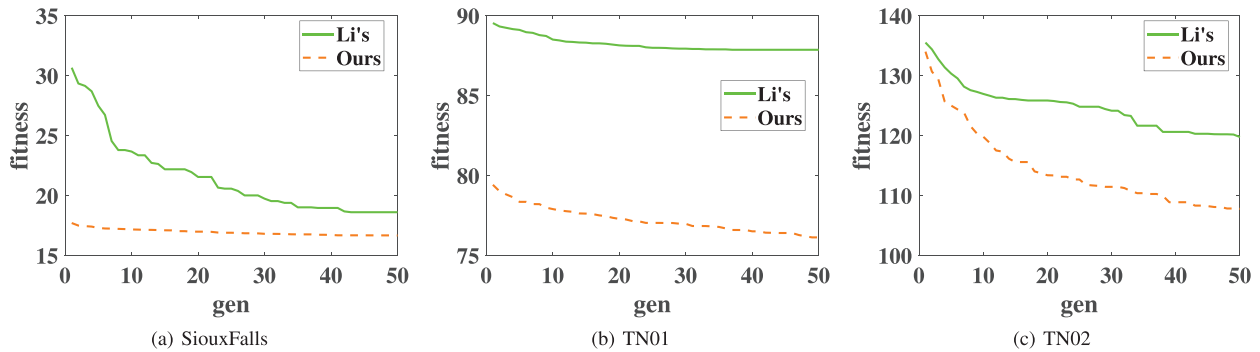

(a) SiouxFalls   (b) TN01   (c) TN02

FIGURE 14   Convergence comparison of Li's and our TSC methods in the crowded traffic situation

can notice two following points on the convergence comparison.

- The vDUE helps our agile TSC method to better solve the TSC problem, which has been demonstrated by the fitness gaps between Li's and our TSC methods at the initial generation in all test cases. Moreover, the importance of the agile method seems to increase as the traffic situation intensifies, because the initial fitness gap of the two TSC methods enlarges from the normal traffic situation, to the crowded and severe situation for all three networks.
- Our agile TSC method has a faster convergence speed than Li's TSC method, because the convergence curves of the former are all placed below those of the latter in all test cases.

In a word, our proposed agile TSC system has shown its better performance over the classic and Li's methods in various traffic situations. Hence, its effectiveness to solve the bi-level TSC problems has been validated.

6 | CONCLUSION

This paper proposes the vDUE model for the traffic assignment of the bi-level TSC problem, by combining the macroscopical description of the traffic status and the microscopical depiction of the vehicle behaviors. The vDUE considers the computational simplicity and traffic dynamics simultaneously, which is helpful for a practical TSC system. Subsequently, an agile
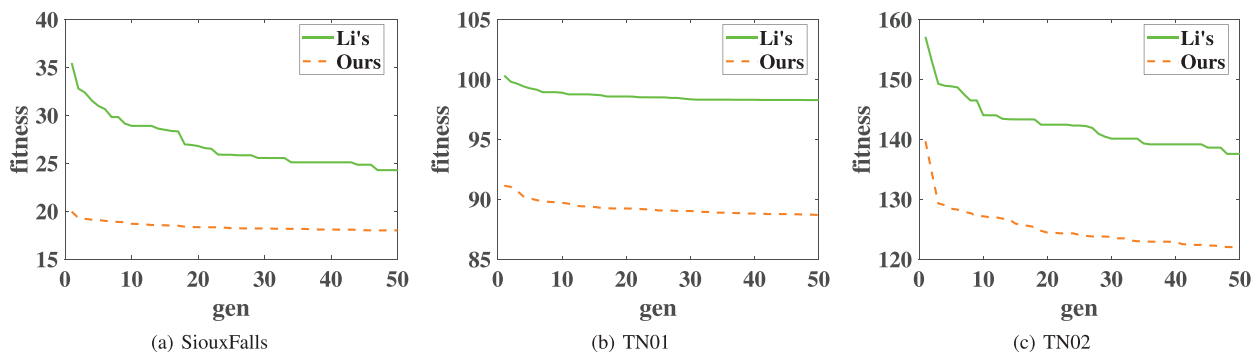

(a) SiouxFalls   (b) TN01   (c) TN02

FIGURE 15   Convergence comparison of Li's and our TSC methods in the severe traffic situation

method is also proposed for the vDUE traffic assignment, by probabilistically rerouting traveling vehicles based on the saturation approximation. In this way, different traffic situations can be flexibly handled, especially the crowded ones and those with emergencies. Further, an agile TSC system is built to solve the bi-level TSC problem, by integrating the agile method for the lower-leveled vDUE traffic assignment and an adaptive differential evolution for the upper-leveled signal timing optimization. In experiments, the DUE comparison is first conducted among AON, rMSA and ours, whose results show that our method can generally obtain better solutions than the other two traffic assignment algorithms within an acceptable running time. Then, TSC comparison is conducted on our proposed agile TSC system with a classic TSC and a state-of-the-art TSC. the results show that our agile TSC overall outperforms its two counterparts and decreases the average travel time of all vehicles up to 21.74% in various traffic situations on all test networks.

However, this study is still based on regular flow demands. Besides, it considers vehicles only in the traffic flow. Hence for future research, we will consider more dynamic traffic environments, where the flow demands may vary a lot over time. Moreover, including pedestrians into the traffic flow should be another promising direction of the TSC work.

REFERENCES
1. Zhao, Y., et al.: A novel approach for traffic signal control: A recommendation perspective. IEEE Intell. Transp. Syst. Mag. 9(3), 127–135 (2017)
2. Han, K., et al.: A bi-level model of dynamic traffic signal control with continuum approximation. Transport. Res. Part C: Emerg. Technol. 55, 409–431 (2015)
3. Li, Z., et al.: Optimizing traffic signal settings in smart cities. IEEE Trans. Smart Grid 8(5), 2382–2393 (2017)
4. Ukkusuri, S., et al.: A bi-level formulation for the combined dynamic equilibrium based traffic signal control. Procedia - Soc. Behav. Sci. 80, 729–752 (2013)
5. Chen, L., Hu, T.: Flow equilibrium under dynamic traffic assignment and signal control–An illustration of pretimed and actuated signal control policies. IEEE Trans. Intell. Transp. Syst. 13(3), 1266–1276 (2012)
6. Xie, X.F., Wang, Z.J.: Combined traffic control and route choice optimization for traffic networks with disruptive changes. Transportmetrica B-Transp. Dyn. 7(1), 814–833 (2019)
7. Xu, B., et al.: Cooperative method of traffic signal optimization and speed control of connected vehicles at isolated intersections. IEEE Trans. Intell. Transp. Syst. 20(4), 1390–1403 (2019)
8. Bi, Y., et al.: Optimal type-2 fuzzy system for arterial traffic signal control. IEEE Trans. Intell. Transp. Syst. 19(9), 3009–3027 (2018)
9. Jovanović, A., et al.: Area-wide urban traffic control: A bee colony optimization approach. Transport. Res. Part C: Emerg. Technol. 77, 329–350 (2017)
10. Zhang, Y., Zhou, Y.: Distributed coordination control of traffic network flow using adaptive genetic algorithm based on cloud computing. J. Netw. Comp. Appl. 119, 110–120 (2018)
11. Belletti, F., et al.: Expert level control of ramp metering based on multitask deep reinforcement learning. IEEE Trans. Intell. Transp. Syst. 19(4), 1198–1207 (2018)
12. Shen, L., et al.: Development of dynamic platoon dispersion models for predictive traffic signal control. IEEE Trans. Intell. Transp. Syst. 20(2), 431–440 (2019)
13. Zhang, X., Mahadevan, S.: A Bio-inspired approach to traffic network equilibrium assignment problem. IEEE Trans. Cybern. 48(4), 1304–1315 (2018)
14. Wang, W., et al.: Learning and inferring a driver's braking action in car-following scenarios. IEEE Trans. Veh. Technol. 67(5), 3887–3899 (2018)
15. Timotheou, S., et al.: Distributed traffic signal control using the cell transmission model via the alternating direction method of multipliers. IEEE Trans. Intell. Transp. Syst. 16(2), 919–933 (2015)
16. Yu, H., et al.: Performance evaluation of integrated strategy of vehicle route guidance and traffic signal control using traffic simulation. IET Intel. Transport Syst. 12(7), 696–702 (2018)
17. Yau, K., et al.: A survey on reinforcement learning models and algorithms for traffic signal control. ACM Comput. Surv. 50(3), 34:1–34:38 (2017)
18. Cai, C., et al.: Vehicle-to-infrastructure communication-based adaptive traffic signal control. IET Intel. Transport Syst. 7(3), 351–360 (2013)
19. Zhao, D., et al.: Computational intelligence in urban traffic signal control: A survey. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) 42(4), 485–494 (2012)
20. Zhong, J., et al.: Multifactorial genetic programming for symbolic regression problems. IEEE Trans. Syst. Man Cybern.: Syst. 50(11), 4492–4505 (2018)
21. Tan, T., et al.: Cooperative deep reinforcement learning for large-scale traffic grid signal control. IEEE Trans. Cybern. 50(6), 2687–2700 (2019)
22. Mousavi, S.S., et al.: Traffic light control using deep policy-gradient and value-function-based reinforcement learning. IET Intel. Transport Syst. 11(7), 417–423 (2017)
23. Jin, J., et al.: An intelligent control system for traffic lights with simulation-based evaluation. Control Eng. Pract. 58, 24–33 (2017)
24. Chiou, S.W.: A data-driven bi-level program for knowledge-based signal control system under uncertainty. Knowl.-Based Syst. 160, 210–227 (2018)
25. Stabler, B., et al.: Transportation networks for research. https://github.com/bstabler/TransportationNetworks. Accessed Aug 8, 2020.
26. Jiao, P., et al.: Real-time traffic signal control for intersections based on dynamic O–D estimation and multi-objective optimisation: Combined model and algorithm. IET Intel. Transport Syst. 12(7), 619–630 (2018)
27. Hajbabaie, A., Benekohal, R.F.: A Program for simultaneous network signal timing optimization and traffic assignment. IEEE Trans. Intell. Transp. Syst. 16(5), 2573–2586 (2015)
28. Gong, Y., et al.: Learning multimodal parameters: A bare-bones niching differential evolution approach. IEEE Trans. Neural Networks Learn. Syst. 29(7), 2944–2959 (2018)
29. Das, S., Suganthan, P.N.: Differential Evolution: A survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15(1), 4–31 (2011)
30. Chen, Y., et al.: An adaptive archive-based evolutionary framework for many-task optimization. IEEE Trans. Emerging Top. Comput. Intell. 1–16 (2019)
31. Zaidi, A.A., et al.: Back-pressure traffic signal control with fixed and adaptive routing for urban vehicular networks. IEEE Trans. Intell. Transp. Syst. 17(8), 2134–2143 (2016)