This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2016.2526032, IEEE Transactions on Smart Grid

Optimizing Traffic Signal Settings in Smart Cities

Zhiyi Li, *Student Member, IEEE*, Mohammad Shahidehpour, *Fellow, IEEE*, Shay Bahramirad, *Senior Member*, and Amin Khodaei, *Senior Member, IEEE*

***Abstract*—Traffic signals play a critical role in smart cities for mitigating traffic congestions and reducing the emission in metropolitan areas. This paper proposes a bi-level optimization framework to settle the optimal traffic signal setting problem. The upper-level problem determines the traffic signal settings to minimize the drivers' average travel time, while the lower-level problem aims for achieving the network equilibrium using the settings calculated at the upper level. Genetic algorithm is employed with the integration of microscopic-traffic-simulation based dynamic traffic assignment (DTA) to decouple the complex bi-level problem into tractable single-level problems which are solved sequentially. Case studies on a synthetic traffic network and a real-world traffic subnetwork are conducted to examine the effectiveness of the proposed model and relevant solution methods. Additional strategies are provided for the extension of the proposed model and the acceleration solution process in large-area traffic network applications.**

***Index Terms*—Smart cities, traffic signal setting, bi-level optimization, dynamic traffic assignment.**

I. Introduction

INTELLIGENT transportation systems (ITS) are becoming increasingly important in smart cities [1]. Rather than counting solely on developing new roads or increasing road capacities, ITS utilizes advanced information and communication technologies such as real-time vehicle-to-vehicle (V2V) [2] and vehicle to infrastructure (V2I) [3] communications to smooth out traffic flows and reduce road congestion. ITS provides drivers with the critical traffic information that would help improve road safety and traffic efficiency [4].

As a vital component of ITS, traffic signals can play a very important role in strategic traffic management that would lead to a preferred distribution of traffic flows [5]. The traffic management authority would need to consider offset time, split time (i.e., green time), cycle time and phase sequences of traffic signals in order to appropriately control traffic signals and reduce the travel time in congested roads. The lack of coordination among traffic signals can affect drivers' total travel time and incur congestions in some areas which would further prolong the total travel time. On the other hand, a proper coordination among traffic signals can also respond positively to emission control and environmental concerns.

ITS allows drivers and the traffic management authority exchange information in real time. Drivers report their trip information (i.e. origin, destination and departure time) before departure. The traffic management authority uses the drivers' information to update the traffic signal and trip demand database periodically. Given the ITS trip data, drivers are enabled to have more accurate perceptions of future network conditions and traffic management authority manages to minimize potential traffic congestion and be responsive to drivers' trip demands by appropriately setting traffic signals.

Different from other studies [6],[7] which optimized traffic signal settings conditioned on pre-defined drivers' routes, we consider interdependencies of drivers' choices and traffic signal settings in this paper. The traffic-signal-setting problem in nature is a leader-follower Stackelberg game which can be formulated as a bi-level optimization problem where the upper and the lower level problems belong to the leader (traffic management authority) and the follower (drivers), respectively. Traffic management authority wants to minimize drivers' average travel time by choosing the optimal traffic signal settings. Note that traffic management authority will be able to evaluate the effects of a particular set of traffic signal settings only by taking drivers' routes into account. In turn, drivers solve their own optimization problem to find the fastest route based on the determined traffic signal settings. As a result, the traffic-signal-setting problem is formulated as a bi-level optimization problem in our paper.

The bi-level problem can be expressed as a mathematical program with equilibrium constraints (MPEC) by converting the lower-level problem as a set of additional constraints of the upper-level problem. For instance, if the lower-level problem is convex, then this lower-level problem can be recast using its Karush–Kuhn–Tucker conditions and integrated into the upper-level problem as a set of additional complementarity constraints. In this way, the bi-level optimization problem is converted into an MPEC. The bi-level optimization problem is difficult to solve mainly due to the tight couplings between the two levels. Thus, we use the Genetic Algorithm (GA) to decouple the original bi-level problem into two single-level problems and solve them sequentially.

The main contributions of this paper are summarized as follows:

(1) The paper proposes an optimization-based framework for determining adaptive traffic signal time settings in smart cities.
(2) The paper formulates a comprehensive bi-level optimization model, which optimizes and coordinates split time, cycle time, phase sequences and offsets simultaneously for traffic signals at all intersections.
(3) The paper decouples the bi-level problem into tractable single-level problems and employs GA to find near-optimal solutions with reasonable computation efforts.
(4) The paper develops a microscopic simulation-based dynamic traffic assignment model for capturing the network equilibrium in traffic flows.

The remainder of this paper is organized as follows: Section

This project was funded by a grant from the Commonwealth Edison Company (ComEd). Zhiyi Li and Mohammad Shahidehpour are with the Robert W. Galvin Center for Electricity Innovation at Illinois Institute of Technology (zhiyi.li@hawk.iit.edu; ms@iit.edu); Mohammad Shahidehpour is also a research professor in the ECE Department at King Abdulaziz University in Saudi Arabia. Shay Bahramirad is with ComEd (Shay.Bahramirad@ComEd.com); Amin Khodaei is with the ECE Department, University of Denver (Amin.Khodaei@du.edu).

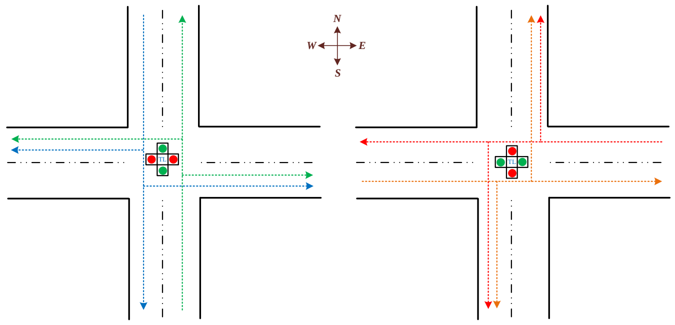
II formulates the bi-level optimization model to determine the optimal setting of traffic signals. Section III explains the solution methods for the proposed optimization problem. Section IV presents and analyzes the numerical results from case studies. Section V gives some ideas on model extension, as well as solution process acceleration strategies for real-world large-area network applications. Relevant conclusions are summarized in Section VI.

II. Model Formulation

A. *Traffic Signal Integration*

In practice, traffic lights located at any intersections are synchronized that would follow specific patterns for managing the traffic flow and safety. Accordingly, regional traffic signals at a given traffic direction can be integrated into a single traffic signal which would simultaneously regulate the flow of all vehicles approaching corresponding intersections from different directions. Such integration could reduce the number of control variables and accelerate the solution process in the traffic flow optimization problem. This paper assumes each integrated traffic signal, referred to as a traffic signal henceforth, performs traffic flow regulations in two perpendicular N-S and E-W directions.

Fig. 1 shows the two distinct signal phases regulating the traffic at an intersection (arrows refer to allowable traffic directions). The traffic light color states in these two directions are mutually exclusive in each phase. For simplicity, we only discuss the color states in the E-W direction hereinafter.



(a) Green in the N-S direction (b) Green in the E-W direction

Fig. 1 Traffic flow regulation

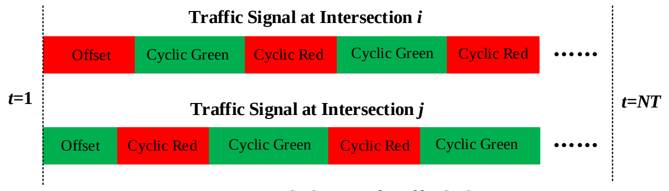


Fig. 2 Signal phases of traffic light

Each traffic signal comprises a sequence of phases (shown in Fig. 2) and each phase indicates the corresponding color state for traffic control. In this paper, each traffic signal is assumed to have three phases (i.e., offset, green and red). Following the initial offset phase, green and red phase are repeated periodically. The two phase sequence is dependent on the initial offset phase. Although the yellow phase exists in practice, its duration in the optimization process is subtracted from that of green state and only treated separately in the implementation process.

B. *Formulation of Bi-level Optimization*

This subsection describes the formulation for the optimal traffic signal setting problem. A list of notations and the corresponding definitions are given as follows.

Notations	**Definition**
	Indices
N	Index for intersections
M	Index for drivers
T	Index for time intervals
	Variables
$I_{n,t}$	Binary variable representing traffic signal at intersection *n* in time interval *t*, if equal to 1, direction E-W is green, otherwise, direction N-S is green
T_n^O	Offset of traffic signal at intersection *n*
T_n^G	Cyclic green time of traffic signal at intersection *n*
T_n^R	Cyclic red time of traffic signal at intersection *n*
r_m	Route choice for driver *m*
r_m^*	Optimal (fastest) route for driver *m*
T_{r_m}	Total travel time for driver *m* to finish the trip by choosing route r_m
	Parameters
N	Number of intersections
M	Number of drivers
NT	Number of time intervals
$I_{n,0}$	Initial status of traffic signal at intersection *n*
$T_{n,0}^G$	Initial green time of traffic signal at intersection *n*
$T_{n,0}^R$	Initial red time of traffic signal at intersection *n*
$T_{\min,n}^G$, $T_{\max,n}^G$	Minimum and maximum green time of traffic signal at intersection *n*
$T_{\min,n}^R$, $T_{\max,n}^R$	Minimum and maximum red time of traffic signal at intersection *n*
R_m	Candidate routes for driver *m*
T_m^{Depart}	Departure time for driver *m* from its origin
	Functions
$T_m^{Travel}(\cdot)$	Total travel time for driver *m* on its trip during the time periods under study
$T_m^{Trip}(\cdot)$	Total time for driver *m* to finish its trip

The objective function and constraints in the optimization problem are listed as follows.

- *Objective Function*

The objective is to minimize the average travel time for all drivers during the studied time intervals, which could also be viewed as the surrogate measure for the reduction of pollution and fuel consumption. The objective function is represented as

$$\min \frac{1}{M}\sum_m T_m^{Travel}\left(r_m^*\right) \tag{1}$$

where

$$T_m^{Travel}\left(r_m^*\right)=\begin{cases} T_{r_m^*} & ,\text{if } T_m^{Depart}+T_{r_m^*}\le NT \\ NT-T_m^{Depart} & ,\text{if } T_m^{Depart}+T_{r_m^*}> NT \end{cases},\forall m \tag{2}$$

- *Constraints on Green/Red Phases*

A very long green phase might result in prolonged waiting times for other drivers in an intersection, whereas a very short green phase might hamper the traffic safety. Thus, the green phase duration must be within a reasonable range as

$$T_{\min,n}^G \le T_n^G \le T_{\max,n}^G, \forall n \tag{3}$$

Similarly, red phase duration must be within a reasonable range as

$$T^R_{\min,n} \le T^R_n \le T^R_{\max,n}, \forall n \quad (4)$$

- *Constraints for Offset Phase Duration*

Traffic signals should be switched smoothly which could otherwise incur unexpected implications of traffic safety. This would include the offset phase (i.e., initial color state in the new settings). So

$$T^O_{\min,n} \le T^O_n \le T^O_{\max,n}, \forall n \quad (5)$$

where

$$T^O_{\min,n} = \max\left\{0, I_{n,0}\cdot\left(T^G_{\min,n} - T^G_{n,0}\right) + \left(1-I_{n,0}\right)\cdot\left(T^R_{\min,n} - T^R_{n,0}\right)\right\}, \forall n \quad (6)$$

$$T^O_{\max,n} = I_{n,0}\cdot\left(T^G_{\max,n} - T^G_{n,0}\right) + \left(1-I_{n,0}\right)\cdot\left(T^R_{\max,n} - T^R_{n,0}\right), \forall n \quad (7)$$

- *Constraints for Traffic Signal States*

The durations of traffic signal states are solely dependent on signal setting decisions (i.e. T^O_n , T^G_n and T^R_n for each intersection n). The relationships among states are represented as follows.

The color and phase duration in the initial (offset) states is determined by T^O_n, represented as

$$I_{n,t} = I_{n,0}, \text{ when } 1 \le t \le T^O_n, \forall n \quad (8)$$

After offset phase, the duration of traffic signal cycles is the sum of T^G_n plus T^R_n. Note that the cycle sequences are affected by the initial state. Accordingly, the first cycle is represented as

$$I_{n,t} = \begin{cases} 1-I_{n,0}, \text{ when } T^O_n + 1 \le t \le T^X_n \\ I_{n,0}, \text{ when } T^X_n + 1 \le t \le T^O_n + T^G_n + T^R_n \end{cases}, \forall n \quad (9)$$

where

$$T^X_n = T^O_n + T^R_n \cdot I_{n,0} + T^G_n \cdot \left(1 - I_{n,0}\right), \forall n \quad (10)$$

Then color cycles will be repeated as

$$I_{n,t} = I_{n,t-T^G_n-T^R_n}, \text{ when } T^O_n + T^G_n + T^R_n + 1 \le t \le NT, \forall n \quad (11)$$

- *Route Constraints*

Each driver will use the traffic signal information and other drivers' route decisions to look for a route with the minimum travel time, which is modeled as

$$r^*_m = \arg\min_{r_m \in R_m} T^{Trip}_m\left(T^{Depart}_m, I_{1,1},\ldots,I_{N,NT}, r^*_1,\ldots,r^*_{m-1}, r^*_{m+1},\ldots,r^*_M\right), \forall m \quad (12)$$

III. Solution Methodology

A. The Framework of the Proposed Solution

The framework of the proposed bi-level optimization problem is depicted in Fig. 3. The upper and the lower level problems are managed by the traffic management authority and drivers, respectively. The traffic management authority would minimize drivers' average travel time by optimizing traffic signal settings as it takes drivers' route choices into account. In turn, drivers look for the fastest route based on the information on the traffic signal settings.

The objective function (1) with constraints (2)-(11) forms the upper-level problem where T^O_n , T^G_n and T^R_n $(\forall n)$ are decision variables for adjusting traffic signal settings and $I_{n,t}$ $(\forall n, \forall t)$ are state variables representing traffic signal state. In essence, given the traffic signal states $I_{n,t}$ $(\forall n, \forall t)$, constraint (12) itself is an optimization problem for the fastest route r^*_m pertaining to a particular driver $(\forall m)$. Considering the behavior of drivers, on congested roads, the fastest-route problem at the lower-level aims at achieving the network equilibrium based on the traffic signal information determined in the upper-level problem. In turn, the lower-level problem passes the drivers' route information to the upper-level problem.

The procedure for approximating an network equilibrium is referred to as the dynamic traffic assignment (DTA) [10] which is detailed in Section III.D. At such equilibrium no driver can reduce its travel time by unilaterally changing its route within any time interval. Different from deterministic user equilibrium assignment which assumes drivers are identical and have complete knowledge of the network conditions, the stochastic user equilibrium assignment is capable of modeling the variations in drivers' perceptions of network conditions and in drivers' decisions on route choices. In other words, even with the help of advanced intelligent transportation systems technologies, drivers' perceptions of network conditions and their preferences of route choices are hardly identical. Accordingly, we model the lower level problem as a stochastic user equilibrium DTA problem, which is more appropriate in the environment of smart cities.

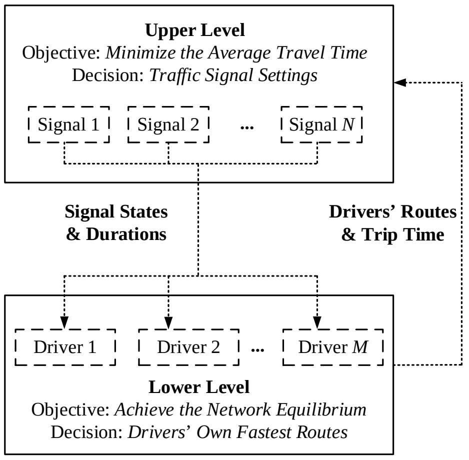

Upper Level
Objective: Minimize the Average Travel Time
Decision: Traffic Signal Settings
Signal 1
Signal 2
...
Signal N
Signal States & Durations
Drivers' Routes & Trip Time
Driver 1
Driver 2
...
Driver M
Lower Level
Objective: Achieve the Network Equilibrium
Decision: Drivers' Own Fastest Routes

Fig. 3 Bi-level optimization framework

B. Hybrid Genetic Algorithm

The bi-level optimization problem is intrinsically difficult to solve mainly due to the tight couplings between the two levels. So we employ GA to solve the proposed bi-level optimization model by selecting the optimum cycle, splits and offsets simultaneously. GA could offer a high-quality near-optimal solution with an affordable computational cost as compared with conventional optimization methods [8]. In particular, the microscopic traffic simulation is successfully integrated with the global search capability provided by GA. To differentiate it from the traditional GA, we use a Hybrid Genetic Algorithm (HGA) for converting the bi-level optimization into an integrated problem with computationally feasible solutions. The integrated problem is composed of two single-level problems which are solved sequentially.

The main process for applying HGA is illustrated in Fig. 4 and described as follows.

Step 1: *Initialization.* Form the random initial population, where the chromosome of each individual represents a candidate solution encoded as real-integer-valued string of length $3 \cdot N$. Here, subscript s refers to individual s. Each gene in the chromosome is a decision variable ($T_{s,n}^{O}$, $T_{s,n}^{G}$ or $T_{s,n}^{R}$) representing the phase duration of a traffic signal. So

$$\begin{aligned} &T_{\min,n}^{O} \le T_{s,n}^{O} \le T_{\max,n}^{O}, \forall n, \forall s \\ &T_{\min,n}^{G} \le T_{s,n}^{G} \le T_{\max,n}^{G}, \forall n, \forall s \\ &T_{\min,n}^{R} \le T_{s,n}^{R} \le T_{\max,n}^{R}, \forall n, \forall s \end{aligned} \quad (13)$$

In turn, each traffic signal's setting is jointly determined by a subgroup comprised of three successive genes. In Fig. 5, the permutation of genes corresponds to the sequences of cyclic phases.

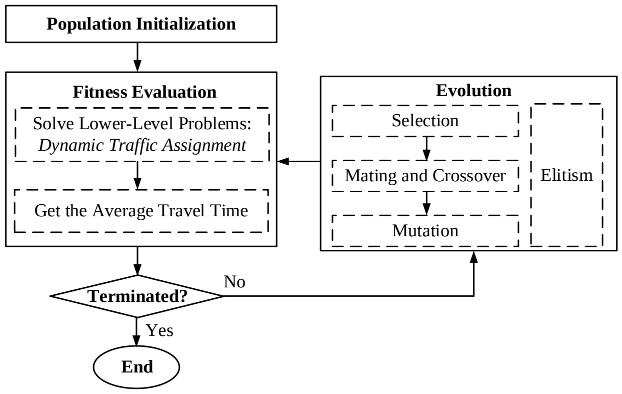

Population Initialization
Fitness Evaluation
Solve Lower-Level Problems: Dynamic Traffic Assignment
Get the Average Travel Time
Evolution
Selection
Mating and Crossover
Mutation
Elitism
Terminated?
No
Yes
End

Fig. 4 Hybrid Genetic Algorithm

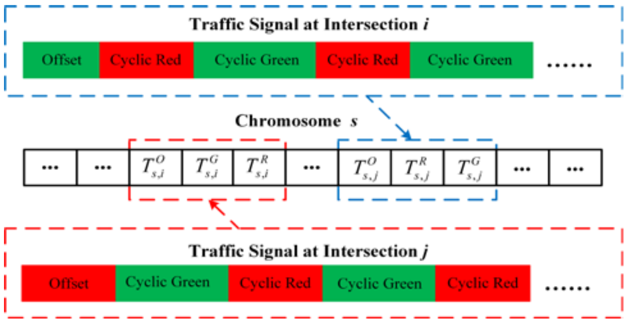


Fig. 5 Chromosome representation

The subgroups are consecutively placed in the chromosome for mapping the complete traffic signal settings. Such chromosome representation takes into account the interdependency of decision variables among adjacent traffic signals and phase durations of individual traffic signals [9].

Step 2: *Fitness evaluation.* This is the most distinctive step in the proposed algorithm, where an individual's fitness value in the current population is evaluated. The individual's fitness is the objective value of the upper-level problem, which is computed after solving the lower-level problem once the traffic signal settings (represented as individual chromosomes) are given. In other words, individuals correspond to a feasible solution of the original bi-level problem which is calculated sequentially by performing a global search for best individual(s) at the upper-level problem and finding the corresponding solution in the lower-level problem. Specifically, traffic signals are set according to (8)-(11) for individual s defining $T_{s,n}^{O}$, $T_{s,n}^{G}$ and $T_{s,n}^{R}$ ($\forall n$), and the corresponding fitness value is calculated by solving the following problem,

$$f_s = \sum_{m}^{M} T_m^{Travel}\left(r_{s,m}^{*}\right), \forall s \quad (14)$$

where

$$r_{s,m}^{*} = \arg \min_{r_{s,m} \in R_{s,m}} T_m^{Trip}\left(T_m^{Depart}, I_{s,1,1}, \ldots, I_{s,N,NT}, r_{s,1}^{*}, \ldots, r_{s,m-1}^{*}, r_{s,m+1}^{*}, \ldots, r_{s,M}^{*}\right), \forall m, \forall s \quad (15)$$

Intuitively, the fitness value is obtained by simultaneously locating the optimal route for each driver (i.e., network equilibrium), which is decoupled from the upper-level problem.

Step 3: *Evolution.* The population is updated by replacing all individuals at the current generation with new potential solutions based on fitness values. This step is implemented using the following four genetic operators:

Elitism: A subset of individuals with the best fitness value(s) is selected and passed on to the next generation, which guarantees that the solution quality at the next generation will not be any worse than that of the current generation, thereby avoiding any solution oscillations. Note that these elitist individuals will not go through the following three operators.

Selection: The remaining individuals of the current generation are randomly chosen as the parents of the next generation using the tournament selection mechanism (representing the best fitness value of each tournament) until the number of parents is equal to the size of the population excluding the elitist individuals.

Mating and Crossover: The selected parents are randomly arranged in pairs to produce two solutions for the next generation. For simplicity, a single-point crossover is implemented by randomly selecting a point in each pair and swapping the right-side substrings of the parent strings in the pair.

Mutation: Individuals are randomly self-mutated to achieve a local search around the current solutions, thereby avoiding the premature convergence and maintaining the population diversity. We utilize an individually adaptive mechanism which adapts the individual's mutation probability according to its fitness value so as to probably maintain the individuals with above-average fitness values and disrupt those with below-average fitness values. Particularly, an individual's mutation probability $p_s^{mu} (\forall s)$ is determined by the following equation [11], [12]:

$$p_s^{mu} = \begin{cases} \left(p_{\max}^{mu} - p_{\min}^{mu}\right) \cdot \dfrac{f^{\max} - f_s}{f^{\max} - f^{avg}} + p_{\min}^{mu}, \text{if } f_s \ge f^{avg} \\ p_{\max}^{mu}, \text{if } f_s < f^{avg} \end{cases}, \forall s \quad (16)$$

where $p_{\max}^{mu}$ and $p_{\min}^{mu}$ denote the pre-specified maximum and minimum mutation probability, respectively; f_s , $f^{\max}$ and f^{avg} represent the fitness value of individual *s*, the global best fitness value (which is equal to the best fitness value of the current generation), and the average fitness value of the current generation, respectively. Then a random number is generated for each gene of individual *s* and compared with the determined mutation probability p_s^{mu} ; if the random number is larger than p_s^{mu} , this gene is replaced by a new random value within its range.

Step 4: *Termination criterion.* If either the generation number is at the threshold or the fitness level is viewed satisfactory (e.g., the global best fitness value has not changed for 50 consecutive generations), the algorithm will be terminated with the global best value as the final solution; otherwise, the next generation starts with the new population and go through the process starting from Step 2.

C. *Revised Dijkstra's Algorithm*

Here we determine the expected fastest route for each driver at its departure point. First, traffic network is abstracted to node-edge graph by representing intersections as nodes and lanes as directed edges. If the cruise time for all edges does not vary through the trip, this fastest-route problem can then be solved by revising the Dijkstra's algorithm [13]. *Algorithm* 1 ($\forall m$) calculates the fastest route as shown below.

Algorithm 1: Revised Dijkstra's Algorithm

Input: Driver m's origin and destination nodes s_m and d_m, and its departure time T_m^{Depart}

Output: Driver m's fastest route r_m^* and travel time $T_{r_m^*}$

```
1: $\mathbf{U} \leftarrow \{all\ the\ nodes\}$ , $\mathbf{V} \leftarrow \varnothing$
2: $T_{m,o_m}^{Arrive} \leftarrow T_m^{Depart}$
3: for each $i \in \mathbf{U} \setminus \{o_m\}$ do
4:     $T_{m,i}^{Arrive} \leftarrow \infty$
5:     $Prev[i] \leftarrow \varnothing$
6: while $d_m \in \mathbf{U}$ do
7:     $i^* \leftarrow \arg\min_{i\in\mathbf{U}} \{T_{m,i}^{Arrive}\}$
8:     $\mathbf{U} \leftarrow \mathbf{U} \setminus \{i^*\}$ , $\mathbf{V} \leftarrow \mathbf{V} \cup \{i^*\}$
9:     for each adjacent node $j$ of $i^*$ in U do
10:        if $T_{m,j}^{Arrive} > T_{m,i^*}^{Arrive} + T_{m,i^*-j}^{Cruise} + T_{m,i^*-j}^{Wait}$ then
11:            $T_{m,j}^{Arrive} \leftarrow T_{m,i^*}^{Arrive} + T_{m,i^*-j}^{Cruise} + T_{m,i^*-j}^{Wait}$
12:            $Prev[j] \leftarrow i^*$
13: $r_m^* \leftarrow \varnothing$ , $i \leftarrow d_m$
14: while $Prev[i] \neq \varnothing$ do
15:     $r_m^* \leftarrow r_m^* \cup \{i\}$
16:     $i \leftarrow Prev[i]$
17: $T_{r_m^*} = T_{m,d_m}^{Arrive} - T_{m,o_m}^{Arrive}$
```

The travel time consists of the cruise time along traversed edges and the waiting time at traversed nodes. Neglecting the accelerating and decelerating processes, the cruise time $T_{m,i-j}^{Cruise}$ of driver m along the directed edge *i-j* is determined by

$$T_{m,i-j}^{Cruise} = L_{i-j} / V_{m,i-j} \tag{17}$$

where L_{i-j} is the length of edge *i-j*; $V_{m,i-j}$ is the expected speed on edge *i-j* for driver *m*. In addition, when drivers arrive at the end node of the traversed edge, they will enter the downstream edge immediately if there is no traffic signal at this node or the traffic signal is green; otherwise, they will join the queue at the end of the edge and wait until the traffic signal turns green at a later time. To be more realistic, the effect of yellow signals is also considered here, which depend on the driving behavior (aggressive or mild). In other words, when the signal is yellow, an aggressive driver tends to pass the intersection quickly but a mild driver tends to stop. Accordingly, the waiting time for driver *m* at node i^* who is traveling to node *j* is calculated by,

$$T_{m,i^*-j}^{Wait} = \begin{cases} 0, \text{if } green \\ T_{i^*}^{NextG} - T_{m,i^*}^{Arrive}, \text{if } red \\ \delta_m \cdot \left(T_{i^*}^{NextR} - T_{m,i^*}^{Arrive} + T_{i^*}^{R}\right), \text{if } yellow \end{cases} \tag{18}$$

where $T_{i^*}^{NextR}$ and $T_{i^*}^{NextG}$ are the time instants when the traffic signal at node i^* turns green and red, respectively; δ_m is a binary indictor representing driving behavior of driver *m* (if driver *m* is aggressive, δ_m equals 1, otherwise, 0). For simplicity, the aggressiveness of driver *m* is sampled by

$$\delta_m = \begin{cases} 0, \text{if } rand(0,1) < 0.5 \\ 1, \text{if } rand(0,1) \geq 0.5 \end{cases}, \forall m \tag{19}$$

in which $rand(0,1)$ is a uniform distributed single random number which is between 0 and 1.

D. *Dynamic Traffic Assignment*

Each driver's choice for the fastest route depends on not only traffic signal sequences but also the level of traffic congestion. The congestion level, in turn, depends on the routes selected by other drivers. Considering that such interdependencies make it difficult to predict the actual edge-cruise time, the fastest route cannot be computed solely by the revised Dijkstra's algorithm. Furthermore, the optimal solution of the lower-level problem corresponds to the network equilibrium for all drivers. It is also difficult to solve the lower-level problem analytically mainly due to the mutual interactions among drivers. Instead, we resort to the microscopic simulation-based DTA, which is executed by invoking the revised Dijkstra's algorithm and microscopic traffic simulation iteratively.

At each iteration, DTA finds the expected fastest route for each driver using the revised Dijkstra's algorithm. As this iterative process continues, the fastest routes and the resulting traffic assignment tend to be brought closer to the network equilibrium. Note that if all drivers choose the expected fastest routes at each step, these routes might become congested and would no longer be the fastest. Thus, alternative routes should be also considered by travelers at each iteration.

Once all the drivers have chosen their routes at departure, the road traffic dynamics are obtained by microscopic traffic simulation, where drivers follow their routes without further updates along their trips. In this paper, microscopic traffic simulation is conducted using the Simulation of Urban Mobility (SUMO) [14] which is an open-source agent-based microscopic road traffic simulator. In SUMO, the movement of individual vehicles is simulated based upon car-following [15] and lane-changing [16] theories, which render the simulation to be more consistent with the real-world scenarios. Thus, the travel times calculated by the microscopic traffic simulation are reliable. Furthermore, SUMO provides trajectories for each vehicle, which are vitally helpful in extracting temporal and spatial dynamics of drivers' behaviors. For instance, it is convenient to obtain the average cruise time, speed, or density on individual edges during any specified time intervals.

The procedure for performing the microscopic-simulation based DTA is shown in Fig. 6 and illustrated as follows.

Step 1: *Initialization.* Initialize the traffic network and determine each edge's initial expected cruise time for drivers departing at different time.

Step 2: *Finding expected fastest routes.* Determine the expected fastest route for each driver in the traffic condition with time-dependent expected edge-cruise time. In such condition, edge-cruise time is approximated by the average speed on this edge from driver m's departure time T_m^{Depart} to the end of simulation. The speed is obtained based on the results of microscopic traffic simulation discussed in the next subsection. Note that if there is no driver on this edge during the relevant simulation periods, the edge's speed limit is used as its expected speed.

Step 3: *Forming or updating route choice set.* Drivers are assumed to opt for other feasible routes besides the expected fastest routes at each iteration. That is to say, each driver randomly chooses his routes from the candidate route set and the probabilities of these candidate routes are determined by the revised method of successive averages [17],[18], as shown below.

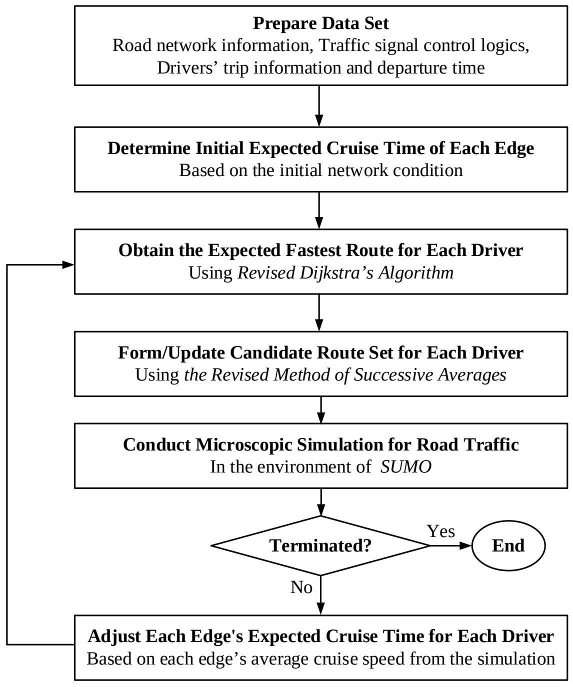

Prepare Data Set
Road network information, Traffic signal control logics, Drivers' trip information and departure time
Determine Initial Expected Cruise Time of Each Edge
Based on the initial network condition
Obtain the Expected Fastest Route for Each Driver
Using Revised Dijkstra's Algorithm
Form/Update Candidate Route Set for Each Driver
Using the Revised Method of Successive Averages
Conduct Microscopic Simulation for Road Traffic
In the environment of SUMO
Terminated?
Yes
End
No
Adjust Each Edge's Expected Cruise Time for Each Driver
Based on each edge's average cruise speed from the simulation

Fig. 6 Dynamic Traffic Assignment

- First iteration (when *k*=1)

Here, superscript *k* refers to the *k*-th iteration. The initial candidate route set is formed for each driver, which includes its initial expected fastest route and some other feasible routes. Note that there is no need to exhaustively search all feasible routes. Instead, at most five representative routes for each driver are included in the initial set in order to accelerate the convergence speed. Accordingly, the candidate route set is defined as

$$R_m^{(k)} \leftarrow \{r_m^{(k)*}\} \cup \{initial\ assigned\ routes\} \tag{20}$$

and the probability for each route is determined by,

$$p_{m,r_m}^{(k)} = \begin{cases} \dfrac{\eta}{k+1}, \text{ if } r_m = r_m^{(k)*} \\ \left(1-\dfrac{\eta}{k+1}\right)\cdot \alpha_{m,r_m}^{(k)}, \text{ if } r_m \neq r_m^{(k)*} \end{cases}, \forall m \tag{21}$$

where

$$\alpha_{m,r_m}^{(k)} = \frac{e^{-T_{r_m}^{(k)}}}{\sum\limits_{r_m \neq r_m^{(k)*}} e^{-T_{r_m}^{(k)}}}, \forall m, \forall r_m \neq r_m^{(k)*} \tag{22}$$

In the above equation, η is a user-defined constant within the range [0, 2]; $p_{m,r_m}^{(k)}$ is the probability to choose the route r_m at the *k*-th iteration for driver *m*; $T_{r_m}^{(k)}$ is the expected travel time on route r_m as a result of the expected traffic condition at the *k*-th iteration. Note that $T_{r_m}^{(k)}$ can be multiplied by a user-defined parameter for representing driver *m*'s aversion to the longer travel time.

- Subsequent iterations (when *k*>1)

If the expected fastest route $r_m^{(k)*}$ is included in the previous candidate route set $R_m^{(k-1)}$ (i.e. $r_m^{(k)*} \in R_m^{(k-1)}$), the probabilities are determined by

$$p_{m,r_m}^{(k)} = \begin{cases} \left(1-\dfrac{\eta}{k+1}\right)\cdot p_{m,r_m}^{(k-1)} + \left(\dfrac{\eta}{k+1}\right)\cdot \alpha_{m,r_m}^{(k)}, \text{ if } r_m = r_m^{(k)*} \\ \left(1-\dfrac{\eta}{k+1}\right)\cdot p_{m,r_m}^{(k-1)}, \text{ if } r_m \neq r_m^{(k)*} \text{ and } r_m \in R_m^{(k-1)} \\ \left(\dfrac{\eta}{k+1}\right)\cdot \alpha_{m,r_m}^{(k)}, \text{ if } r_m \notin R_m^{(k-1)} \end{cases}, \forall m \tag{23}$$

where,

$$\alpha_{m,r_m}^{(k)} = \frac{e^{-T_{r_m}^{(k)}}}{e^{-T_{r_m^{(k)*}}^{(k)}} + \sum\limits_{r_m \notin R_m^{(k-1)}} e^{-T_{r_m}^{(k)}}}, \forall m, \forall r_m \notin R_m^{(k-1)} \text{ or } r_m = r_m^{(k)*} \tag{24}$$

Then the candidate route set remains unchanged (i.e., $R_m^{(k)} \leftarrow R_m^{(k-1)}$). If the expected fastest route $r_m^{(k)*}$ is newly generated, (i.e., $r_m^{(k)*} \notin R_m^{(k-1)}$, excluded in the previous candidate route set $R_m^{(k-1)}$), the probabilities are determined by

$$p_{m,r_m}^{(k)} = \begin{cases} \left(\dfrac{\eta}{k+1}\right)\cdot \alpha_{m,r_m}^{(k)}, \text{ if } r_m \notin R_m^{(k-1)} \\ \left(1-\dfrac{\eta}{k+1}\right)\cdot p_{m,r_m}^{(k-1)}, \text{ if } r_m \in R_m^{(k-1)} \end{cases}, \forall m \tag{25}$$

where

$$\alpha_{m,r_m}^{(k)} = \frac{e^{-T_{r_m}^{(k)}}}{\sum\limits_{r_m \notin R_m^{(k-1)}} e^{-T_{r_m}^{(k)}}}, \forall m, \forall r_m \notin R_m^{(k-1)} \tag{26}$$

Then the candidate route set is updated as $R_m^{(k)} \leftarrow R_m^{(k-1)} \cup \{r_m^{(k)*}\}$. Clearly, the routes with longer travel times are assigned lower probabilities. Besides, the candidate route set contains the expected fastest route from all previous iterations. In fact, those retained from very early iterations are probably very remote from the more recently identified fastest routes. Meanwhile, the newly-generated expected fastest routes identified at later iterations are presumably closer to the actual fastest routes, but receive lower probabilities. Thus, a restarting mechanism is applied to reset the iteration counter to 1 at designated iterations (e.g., every 10 iterations in this paper) in order to minimize the direct influences of earlier iterations and adjust the probabilities for routes identified at later iterations. Accordingly, this algorithm tends to offer a better convergence performance.

Step 4: *Microscopic traffic simulation.* Perform the microscopic traffic simulation in SUMO using the chosen routes so as to obtain the actual travel time $T_{r_m^{(k)}}$ along driver m's chosen route $r_m^{(k)}$ ($\forall m$).

Step 5: *Termination Check.* Approximated error for the network equilibrium solution is measured by the following relative gap at each iteration,

$$Gap^{(k)} = \frac{\sum_m T_{r_m^{(k)}} - \sum_m T_{r_m^{(k)*}}^{(k)}}{\sum_m T_{r_m^{(k)*}}^{(k)}} \times 100\%, \forall k \quad (27)$$

If either a pre-specified tolerance level for the approximated error (e.g., 5% [19]) or maximum number of iterations is reached, this algorithm will be terminated and the chosen routes at the last iteration will be assumed to represent the network equilibrium; otherwise, the next iteration will start by adjusting each edge's expected cruise time according to the microscopic simulation results and reinitiating the process from Step 2.

IV. Case Studies

A. *Synthetic Traffic Network*

To compare the performance of the revised Dijkstra's algorithm with the original one, we create a synthetic traffic network (see Fig. 7) with the parameters in Table 1. Without the loss of generality, we assume each intersection is controlled by traffic signals with the same settings for traffic flow regulations in two perpendicular directions (see Fig. 8).

Assume that an aggressive driver enters at the bottom-left intersection and finishes its trip at the top-right intersection in the free-flow condition. Without considering the waiting time in calculating the total travel time, there will be multiple and equally designated fastest routes that are computed by the original Dijkstra's algorithm. Fig. 9 lists three of such routes as an example. However, it is clear that these routes all have the same total cruise time of 800 *s*, with a totally different waiting time (0 s, 60 s, 80 s, respectively).

Table 1 Parameters for case studies

Network Size	8 km ×8 km
Equally-divided Block Length	2 km
Vehicle Speed Limit	20 m/s
Traffic signals Location	At each intersection

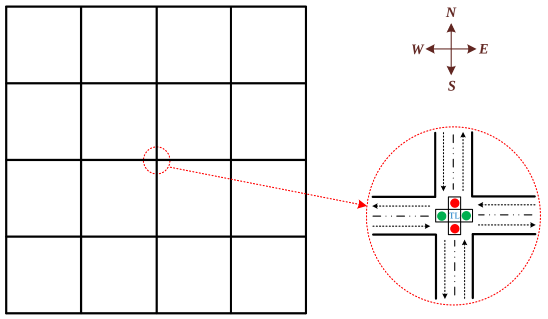


Fig. 7 A synthetic signal network

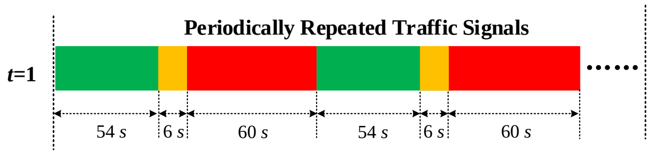


Fig. 8 Traffic signal settings

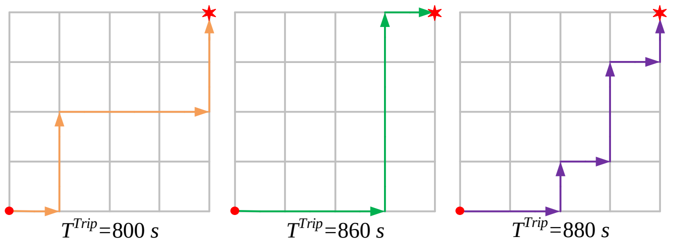


Fig. 9 Travel time comparisons

Then the selection of routes will lead to distinct total travel time, which cannot be distinguished by the original Dijkstra's algorithm. Obviously, the route with a total travel time of 800 *s* is the fastest route, which is also the optimal solution according to the revised Dijkstra's algorithm. The proposed algorithm offers a more suitable solution for the optimal traffic signal setting problem.

B. *Urban Traffic Subnetwork*

To further examine the effectiveness of the proposed model and the corresponding solution method, an urban traffic subnetwork (i.e., vicinity of Illinois Institute of Technology and the Bronzeville community in Chicago) shown in Fig. 10 and obtained from OpenStreetMap [20] is used in our experiments.

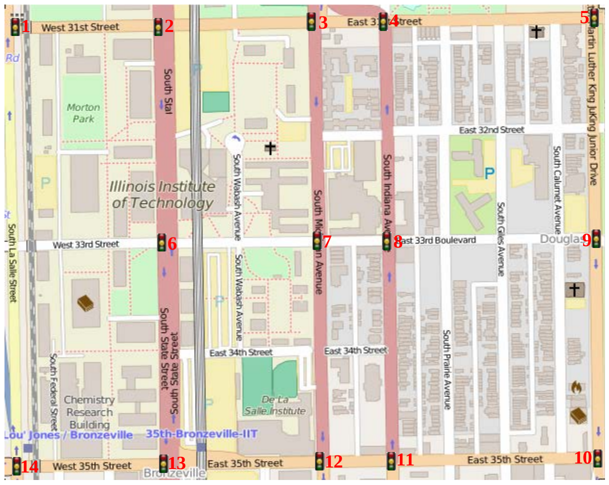


Fig. 10 Urban traffic subnetwork in Chicago

The intersections in our study are all labeled in which the traffic signals are controlled by logics. The drivers' trip information is estimated by the average daily traffic counts [21]. The estimated information is close to the actual number of vehicles crossing the points with installed sensors in certain streets on an average weekday, which is denoted as the base case. The studied time is 1000 *s* and all the drivers are assumed to uniformly insert into the subnetwork in the first 500 *s*. The traffic signal requirements are listed in Table 2 and the HGA parameters are listed in Table 3. The traffic subnetwork is built in SUMO as shown in Fig. 11 with the relevant vehicle settings given in Table 4.

Table 2 Traffic signal requirements

#	Initial State		Green Time		Red Time	
	State	Duration/s	Min/s	Max/s	Min/s	Max/s
1	Red	20	30	60	20	40
2	Red	10	30	60	40	80
3	Green	10	30	60	30	60
4	Green	20	30	60	30	60

5	Red	20	30	60	40	80
6	Green	10	20	40	40	80
7	Green	10	20	40	30	60
8	Red	10	20	40	30	60
9	Green	10	20	40	40	80
10	Green	20	30	60	40	80
11	Green	20	30	60	30	60
12	Green	10	30	60	30	60
13	Red	30	30	60	40	80
14	Red	10	30	60	20	40

Table 3 Parameters of HGA

Population Size	20
Max. Generation	50
Elite Number	1
Tournament Pool Size	4
Min. and Max. Mutation Probability	0.05, 0.5

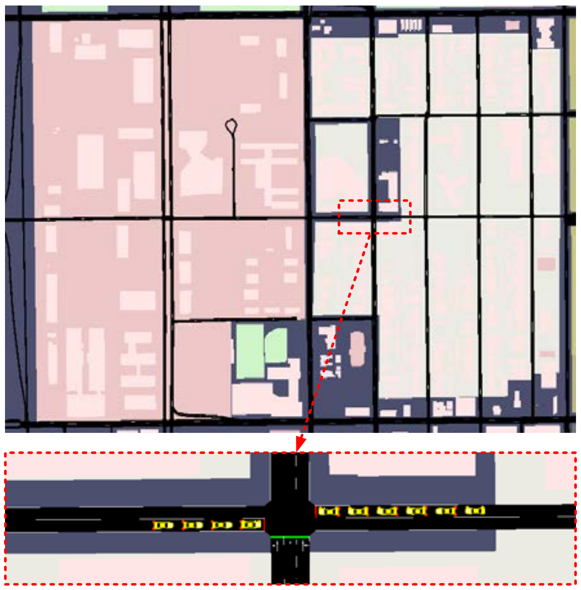


Fig. 11 Test traffic subnetwork in SUMO

Table 4 Vehicle settings in SUMO

Acceleration Ability	2.6 m/s^2
Deceleration Ability	4.5 m/s^2
Driver's Imperfection	0.5
Driver's Reaction Time	1.0 s
Vehicle Length	4 m
Minimum Gap between Vehicles	2 m

In addition, vehicles' movements along every lane-to-lane link *(lane-to-lane link means turning left or right, or going straight)* at any intersection are controlled by the relevant traffic signals (see Fig. 12). The calculated optimal phases of integrated traffic signals are thus divided into a set of coordinated link-based signals at each intersection in SUMO.

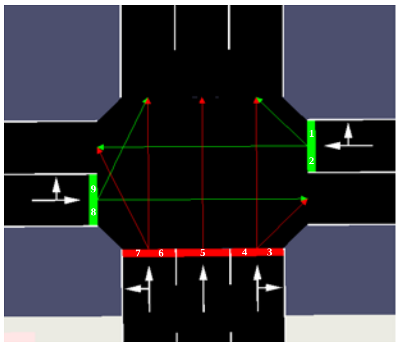


Fig. 12 Traffic signals at intersection 8

For instance, Table 5 shows the coordinated settings in the base case for link-based signals in Fig. 12. Here, default settings are generated by SUMO to adhere to traffic rules. To consider the suitability of the proposed model and the corresponding solution methods, we double the number of drivers in the new case to form a heavy-loaded case. The convergence process of HGA in the base and heavy-loaded cases is shown in Figs. 13 and 14, respectively. Obviously, the final settings optimized by HGA after 50 generations are better than the default settings in both cases. Specifically, in the base case, the average travel time is reduced from 465.2 *s* with default settings to 429.1 *s* with optimal settings after 50 generations; in the heavy-loaded case, the average travel time is reduced from 604.7 *s* with default settings to 510.8 *s* with optimal settings after 50 generations. It is clear that HGA performs better in handling heavy-loaded cases when there might be more potential cases of congestion.

Table 5 Link-based signal coordination at intersection 8

Intersection #	**Phase**	**Intersection Signal**		**Duration**	
		1, 2, 8, 9	3, 4, 5, 6, 7	Default	Optimal
8	A	Red	Green	31	42
	B	Red	Yellow	6	6
	C	Green	Red	31	26
	D	Yellow	Red	6	6

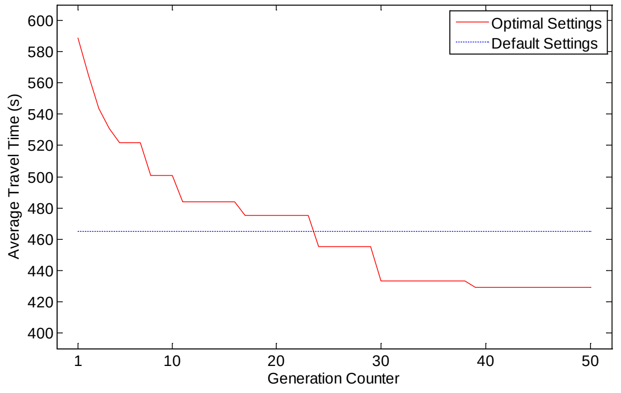


Fig. 13 HGA convergence process in the base case

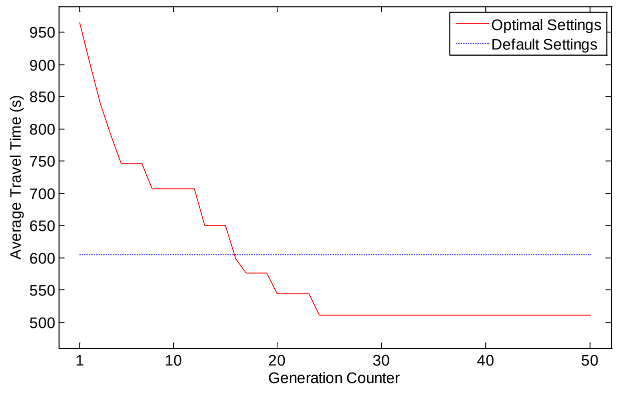


Fig. 14 HGA convergence process in the heavy-loaded case

The representative results for the DTA convergence in both cases are shown in Fig. 15. Here, DTA converges to the network equilibrium in both cases. Especially for the base case, DTA would only require three iterations to converge mainly because there are very few cases of congestion during the simulation process and the expected fastest routes are very close to the actual fastest ones.

The representative convergence process of DTA for identifying the expected fastest routes with and without waiting time are shown in Figs. 16 and 17, respectively. Intuitively, the calculation of the expected fastest routes with waiting time renders the gap to converge more rapidly in both cases. Hence, DTA with the consideration of waiting time is a more efficient approach to achieving the network equilibrium.

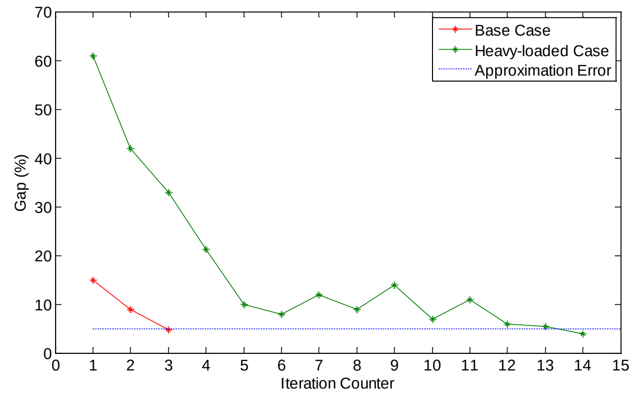


Fig. 15 DTA's convergence process

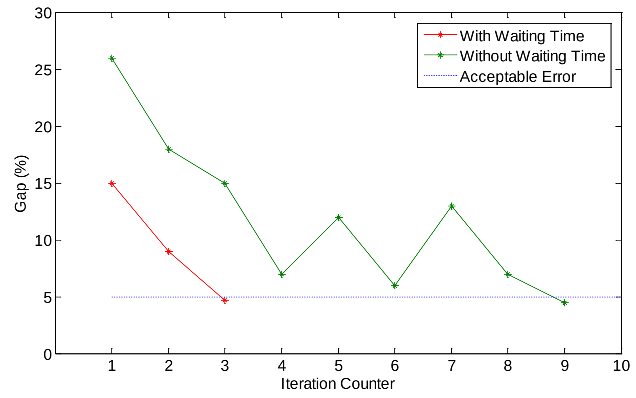


Fig. 16 DTA convergence process in the base case

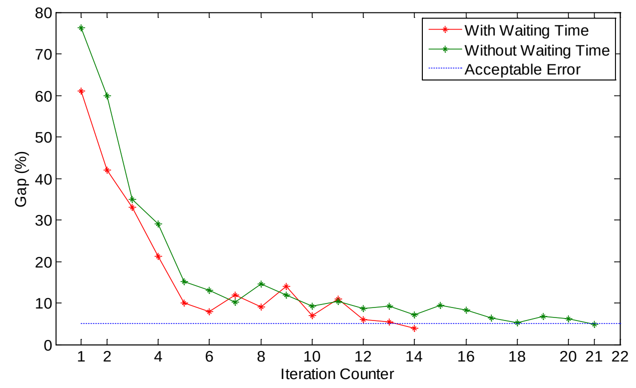


Fig. 17 DTA convergence process in the heavy-loaded case

V. Model Extension and Solution Acceleration for Real-world Large-area Applications

A. Model Extension for Traffic-Adaptive Control Logics

Although the traditional fixed-time traffic signal control mechanism is used most widely in practical cases, the fixed-time tends to cause traffic congestion, especially when the traffic density is high in urban areas. In this context, adaptive traffic control mechanisms become increasingly popular. The adaptive approach can overcome the disadvantages of the fixed-time control by detecting the traffic flow in real time and efficiently responding to any changes in the traffic situation.

The proposed bi-level optimization model can be easily extended to consider the traffic-adaptive signals. Initially, the open-source interface TraCI4Matlab [22] is employed to enable Matlab to interact with SUMO for the implementation of agent-based algorithms in a server-client mechanism as shown in Fig. 18.

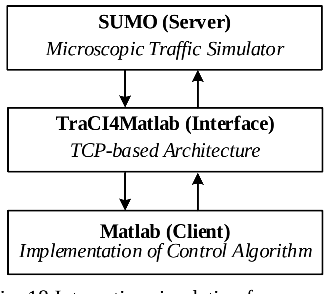


Fig. 18 Interactive simulation framework

In addition, green (red) phase durations of traffic signals controlled by the traffic adaptive mechanism can be adjusted in real time within allowable ranges,

$$\begin{aligned} &T^G_{\min,n} \le T^G_n \le T^G_{\max,n}, \forall n \\ &T^R_{\min,n} \le T^R_n \le T^R_{\max,n}, \forall n \end{aligned} \tag{28}$$

where $T^G_{\min,n}$ ($T^R_{\min,n}$) and $T^G_{\max,n}$ ($T^R_{\max,n}$), rather than T^G_n (T^R_n), are to be optimized ($\forall n$) and T^G_n (T^R_n) is configured in real time based on the agent-based algorithms. Thus, (3) and (4) for traffic-adaptive signals are replaced by the following set in the bi-level optimization model,

$$\begin{aligned} &T^{G,\min}_{\min,n} \le T^G_{\min,n} \le T^{G,\max}_{\min,n}, \forall n \\ &T^{G,\min}_{\max,n} \le T^G_{\max,n} \le T^{G,\max}_{\max,n}, \forall n \\ &T^{R,\min}_{\min,n} \le T^R_{\min,n} \le T^{R,\max}_{\min,n}, \forall n \\ &T^{R,\min}_{\max,n} \le T^R_{\max,n} \le T^{R,\max}_{\max,n}, \forall n \end{aligned} \tag{29}$$

B. Acceleration of Solution Process at Upper-level Problem

1) Parallel Processing for HGA

The proposed HGA can be parallelized in terms of the master-slave implementation. Accordingly, a master is the main processor which generates and evolves the full population of chromosomes and assigns a certain fraction of the individuals to slave processors. The slaves evaluate fitness values by conducting simulation separately for the assigned fraction and return their values (see Fig. 19). Accordingly, parallel processing boosts the speed of global search and offers a full advantage for searching the entire feasible region.

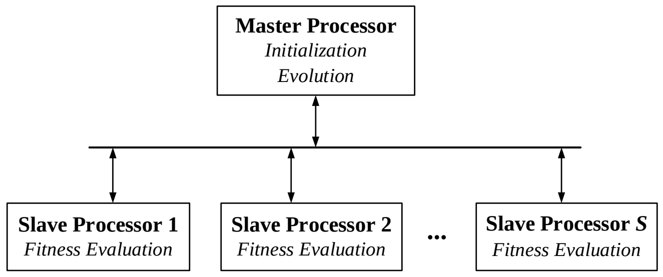


Fig. 19 Implementation of parallel fitness evaluation

2) Multi-layer Partition and Relaxation for Traffic Signals

In a large traffic network, the number of traffic signals is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSG.2016.2526032, IEEE Transactions on Smart Grid

commonly very large and the simultaneous optimization of all traffic signal settings would be unrealistic due to its prohibitive computational cost. Generally, traffic signals only affect local traffic flows in a geographical area. That is, traffic signal settings in a certain area would have negligible effects on drivers travelling in other areas. Accordingly, it is reasonable to optimize traffic signal settings locally.

Accordingly, we partition the entire network into several subareas and allocate layers to these subareas. Each layer is composed of several subareas with uncorrelated traffic signal settings among the subareas. Figure 20 shows a three-layer partition. Here the outer and the inner layers are numbered the lowest and the highest, respectively. The optimization starts from the lowest-numbered layer and proceeds upward through the layers until all traffic signal settings in the entire network are optimized. During the process of finding optimal traffic signal settings in a certain layer, the optimization algorithm is executed in parallel for all subareas while traffic signals in the higher-numbered (lower-numbered) layers are controlled by their default (optimal) settings.

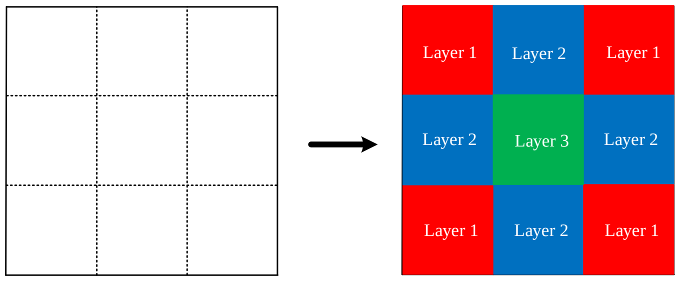


Fig. 20 Multi-layer partition

3) Pre-defined Settings for Certain Traffic Signals

According to vehicle counts, the settings of certain traffic signals can be pre-defined to reduce congestion. For instance, when the traffic flow in one direction is much lighter than that in the perpendicular direction, we may fix the green (red) time in this direction to be set at its minimum (maximum) value. This approach could help reduce the search space and thereby accelerate the solution process.

C. Acceleration of Solution Process at Lower-level Problem

1) Network Pruning for Finding the Expected Fastest Routes

Similar to the original Dijkstra's algorithm, the revised Dijkstra's algorithm might become quite costly for finding the fastest routes in large-area traffic networks [23]. Intuitively, this algorithm will be expedited if the network size is reduced. Here we propose a heuristic approach to speed up the route-finding process by pruning the network for each driver.

The heuristic approach works as follows: First, we classify all the trips into two categories of short and long distance by comparing the Euclidean distances between origins and destinations with a pre-specified value. Second, we form the reduced network for each driver with reference to its trip. Specifically, for a short distance trip, rectangle boundaries are formed by *the four connected* strictly horizontal and vertical paths between its origin and destination. Only the nodes and edges inside or on the boundaries are retained to form the required reduced network. Meanwhile, for a long distance trip, we keep the closest nodes on both sides along the direction of the straight line (corresponding to the Euclidean distance between its origin and destination) as well as the edges connecting these nodes. In case there exists no feasible route for some trip within its reduced network, that network is then expanded by including all the neighboring nodes and in-between edges of the previous network. This expansion process continues until there is at least one feasible route in the newly-formed network.

Fig. 21 provides a simple example in the grid network for forming two types of pruned networks for two drivers with different trip distances. In accordance with drivers' intuition for choosing routes in real life, this heuristics approach reduces the entire network to driver-dependent local areas which are around the diagonals regarding the straight-line distances of their trips, thereby leading to considerable increase in the subsequent fastest-route algorithm with reasonable results.

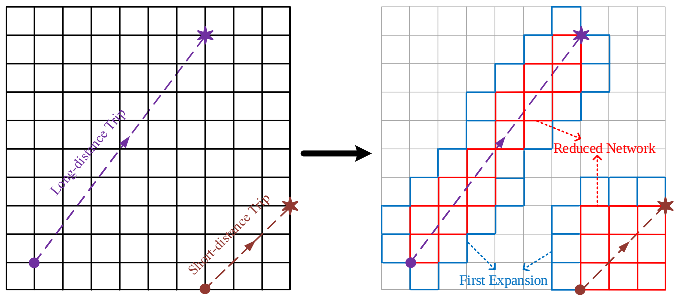


Fig. 21 Network pruning heuristic method

2) Parallel Multi-subarea Microscopic Traffic Simulation

When the simulated traffic network is large, we can perform a spatial decomposition by dividing the entire network into several contiguous subareas and then run microscopic traffic simulation for each subarea in parallel. Such multi-subarea simulation can also be implemented using the master-slave mechanism (see Fig. 22). Accordingly, vehicle movements within each subarea are simulated locally by microscopic simulation. A master controller is responsible for synchronizing and coordinating the simulation process of each subarea when vehicles are crossing the borders between any two adjacent subareas. Since the simulation speed depends largely on the problem size, parallelized microscopic traffic simulation for each subarea runs faster than that for the entire area, which leads to an increase in the entire simulation process.

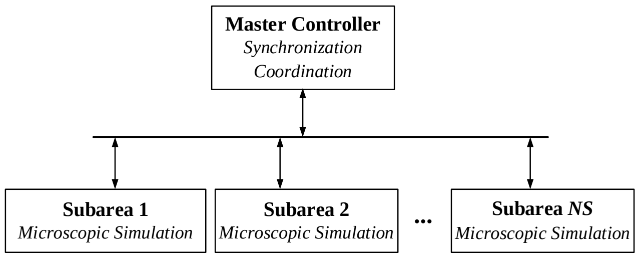


Fig. 22 Implementation of parallel microscopic traffic simulation

3) Sequential Multi-subinterval Dynamic Traffic Assignment

When the simulation interval is long, we can perform temporal decomposition by cutting the whole interval into several successive subintervals and then assigning numbers in sequence to drivers departing each subinterval. In each subinterval, a complete DTA process is implemented and network conditions in the subsequent subintervals are updated by converting the DTA results into a time-dependent background traffic. This process is shown in Fig. 23. With the implementation of this heuristic method, the network equilibrium is approximated in each subinterval instead of the entire interval, thereby resulting in near-optimal solutions with fewer computational efforts.

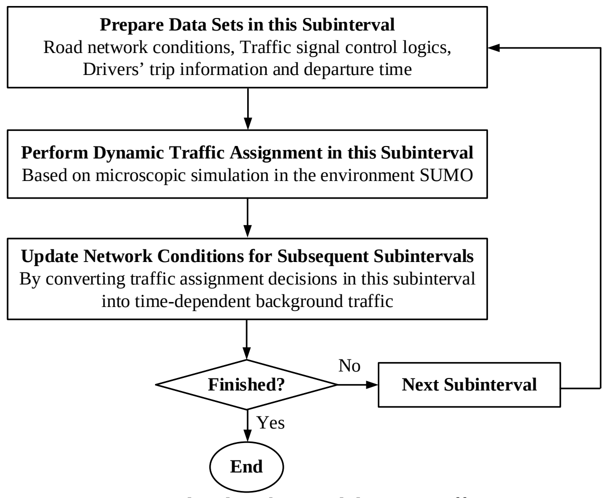

Prepare Data Sets in this Subinterval
Road network conditions, Traffic signal control logics, Drivers' trip information and departure time
Perform Dynamic Traffic Assignment in this Subinterval
Based on microscopic simulation in the environment SUMO
Update Network Conditions for Subsequent Subintervals
By converting traffic assignment decisions in this subinterval into time-dependent background traffic
Finished?
No
Next Subinterval
Yes
End

Fig. 23 Sequential multi-subinterval dynamic traffic assignment

VI. Conclusions and Future Work

This paper proposes a bi-level optimization framework for optimally setting the regional traffic signals. The frequency for optimizing traffic signals mainly depends on traffic patterns. Specifically, when the traffic pattern is dynamically changing (e.g. peak hours), traffic signal settings should be optimized more frequently to accommodate the complex traffic demands; when the traffic pattern remains steady (e.g. late night hours), traffic signal settings should be optimized less frequently. Note that this optimization framework can be easily extended to consider acyclic signal patterns which are more suitable for cases when traffic conditions are dynamically changing within a short time period. In addition, the proposed optimization framework allows smooth transitions between successive traffic signal settings by taking offsets into account.

To solve the proposed optimization problem in a satisfactory manner, the paper applies HGA to the corresponding solution framework. Since HGA decouples the upper and the lower level problems, more sophisticated models can be plugged into upper-level and lower-level frameworks for achieving more reasonable results. Note that other efficient artificial intelligent methods (e.g., Particle Swarm Optimization [24], Harmony Search Algorithm [25]) can be used in place of GA to perform the global search for the optimal setting. In addition, we are validating the proposed acceleration strategies for much large-area traffic network applications.

References

[1] H. Chourabi, T. Nam, S. Walker, J. R. Gil-Garcia, S. Mellouli, K. Nahon, T. a. Pardo, and H. J. Scholl, "Understanding smart cities: An integrative framework," *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, pp. 2289–2297, 2011.

[2] S. An, B.-H. Lee, and D.-R. Shin, "A survey of intelligent transportation systems," *2011 Third Int. Conf. Comput. Intell. Commun. Syst. Networks*, pp. 332–337, 2011.

[3] V. Milanés, J. Villagrá, J. Godoy, J. Simó, J. Pérez, and E. Onieva, "An intelligent V2I-based traffic management system," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 49–58, Mar. 2012.

[4] H. Y. Cheng, V. Gau, C. W. Huang, and J. N. Hwang, "Advanced formation and delivery of traffic information in intelligent transportation systems," *Expert Syst. Appl.*, vol. 39, no. 9, pp. 8356–8368, 2012.

[5] J. Kim, H. S. Mahmassani, T. Hou, and R. M. Alfelor, "Development of real-time simulation-based decision support system for weather responsive traffic signal operations," *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pp. 810–815, 2014.

[6] J. Garcia-Nieto, A. C. Olivera, and E. Alba, "Optimal cycle program of traffic lights with particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 6, pp. 823–839, Dec. 2013.

[7] W. H. Lin and C. Wang, "An enhanced 0-1 mixed-integer LP formulation for traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 238–245, Dec. 2004.

[8] L. Davis, *Handbook of Genetic Algorithms*, New York: Van Nostrand Reinhold, 1991.

[9] J. García-Nieto, E. Alba, and A. C. Olivera, "Swarm intelligence for traffic light scheduling: Application to real urban areas," *Eng. Appl. Artif. Intell.*, vol. 25, no. 2, pp. 274–283, 2012.

[10] Y.-C. Chiu, J. Bottom, M. Mahut, A. Paz, R. Balakrishna, T. Waller, and J. Hicks, "Dynamic traffic assignment: A primer," *Transportation Research E-Circular*, E-C153, 2011.

[11] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in Genetic Algorithms," *IEEE Trans. Syst. Man Cybern.*, vol. 24, no. 4, pp. 656–667, Apr. 1994.

[12] X. Ma, J. Jin, and W. Lei, "Multi-criteria analysis of optimal signal plans using microscopic traffic models," *Transp. Res. Part D Transp. Environ.*, vol. 32, pp. 1–14, Oct. 2014.

[13] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, 1959.

[14] D. Krajzewicz, "Traffic simulation with SUMO – Simulation of urban mobility," in *Fundamentals of Traffic Simulation*, Springer New York, 2010, pp. 269–293.

[15] S. Krauß, "Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics," Universitat zu Koln, 1998.

[16] J. Erdmann, "SUMO's Lane-Changing Model," in *Modeling Mobility with Open Data*, Springer International Publishing, 2015, pp. 105–123.

[17] H. R. Varia and S. L. Dhingra, "Dynamic user equilibrium traffic assignment on congested multidestination network," *J. Transp. Eng.*, vol. 130, no. 2, pp. 211–221, 2004.

[18] H. R. Varia, P. J. Gundaliya, and S. L. Dhingra, "Application of Genetic Algorithms for joint optimization of signal setting parameters and dynamic traffic assignment for the real network data," *Res. Transp. Econ.*, vol. 38, no. 1, pp. 35–44, 2013.

[19] C. O. Tong and S. C. Wong, "A predictive dynamic traffic assignment model in congested capacity-constrained road networks," *Transp. Res. Part B Methodol.*, vol. 34, no. 8, pp. 625–644, 2000.

[20] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *Pervasive Comput. IEEE*, vol. 7, no. 4, pp. 12–18, Oct.-Dec. 2008.

[21] City of Chicago. "Average daily traffic counts," [Online Available]: https://data.cityofchicago.org/Transportation/Average-Daily-Traffic-Counts/pfsx-4n4m. Mar. 2015.

[22] A. F. Acosta, J. J. Espinosa, and J. Espinosa, "TraCI4Matlab: Re-engineering the Python implementation of the TraCI interface," in *SUMO2014-Modeling Mobility With Open Data*, 2014.

[23] P. Sanders and D. Schultes, "Engineering fast route planning algorithms," *Proc. 6th Int. Conf. Exp. Algorithms*, pp. 23–36, 2007.

[24] J. Kennedy, "Particle swarm optimization," *Encyclopedia of Machine Learning*, Springer US, pp. 760-766, 2010.

[25] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol.76, no.2, pp. 60-68, 2001.

Biographies

Zhiyi Li (GSM'14) received the B.S. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 2011 and the M.S. degree in electrical engineering from Zhejiang University, China, in 2014. He is currently pursuing the Ph.D. degree in the Electrical and Computer Engineering Department, Illinois Institute of Technology. His research interests include cyber-physical systems and power system optimization.

Mohammad Shahidehpour (F'01) received the Honorary Doctorate degree from the Polytechnic University of Bucharest, Bucharest, Romania. He is the Bodine Chair Professor and a Director of the Robert W. Galvin Center for Electricity Innovation, Illinois Institute of Technology, Chicago, IL, USA, and also a Research Professor at the King Abdulaziz University in Saudi Arabia.

Shay Bahramirad (SM'14) is Director of Smart Grid and Technology and Innovation Ambassador at ComEd. Her responsibilities include leading, developing, and implementing Microgrid and Smart City initiatives in ComEd's service territory. Dr. Bahramirad is also an Adjunct Professor at the Illinois Institute of Technology. She is the Chair of the IEEE Power & Energy Society (PES) Women in Power, Technical Chair of the 2016 IEEE PES T&D Conference, and Vice Chair of the IEEE PES Distribution Subcommittee. She holds a Ph.D. degree in Electrical Engineering from the Illinois Institute of Technology.

Amin Khodaei (SM'14) received his Ph.D. degree in electrical engineering from the Illinois Institute of Technology, in 2010. He was a visiting faculty (2010–2012) in the Robert W. Galvin Center for Electricity Innovation at IIT. He joined the University of Denver, Denver, CO, USA, in 2013 as an Assistant Professor. His research interests include power system operation, planning, computational economics, and smart electricity grids.