



Spatial iterative coordination for parallel simulation-based optimization of large-scale traffic signal control

Wen Jun Tan¹ , Philipp Andelfinger² , Wentong Cai¹,
David Eckhoff^{3,4} and Alois Knoll⁴

Abstract

Applying simulation-based optimization to city-scale traffic signal optimization can be challenging due to the large search space resulting in high computational complexity. A divide-and-conquer approach can be used to partition the problem and optimized separately, which leads to faster convergence. However, the lack of coordination among the partial solutions may yield a poor-quality global solution. In this paper, we propose a new method for simulation-based optimization of traffic signal control, called spatially iterative coordination for parallel optimization (SICPO), to improve coordination among the partial solutions and reduce synchronization between the partitioned regions. The traffic scenario is simulated to obtain the interactions, which is used to spatially decompose the scenario into regions and identify interdependencies between the regions. Based on the regions, the problem is divided into subproblems which are optimized separately. To coordinate between the subproblems, the interactions between partial solutions are synchronized in two ways. First, multiple iterations of the optimization process can be executed to coordinate the partial solutions at the end of each optimization process. Second, the partial solutions can also be coordinated among the regions by synchronizing the trips across the regions. To reduce computational complexity, parallelism can be applied on two levels: each region is optimized concurrently, and each solution for a region is evaluated in parallel. We demonstrate our method on a real-world road network of Singapore, where SICPO converges to an average travel time 21.6% faster than global optimization at $62.8 \times$ shorter wall-clock time.

Keywords

Traffic signal optimization, parallel simulation-based optimization, spatial coordination

1. Introduction

Rising population densities in cities around the world lead to an increase in traffic congestion.¹ As traffic signal timings have a strong influence on traffic flow,² their optimization is, therefore, essential to alleviate congestion and improve the efficiency of the transport system. However, real-world experiments to evaluate the objective of traffic signal control problems are often prohibitively expensive to set up, especially if the projects did not yield expected results.³ Simulation-based optimization is one of the more cost-effective methods that can address the problem of city-scale traffic signal control without real-world experimentation. A traffic system can be represented by microscopic agent-based traffic simulation, where each agent is usually modeled as a driver-vehicle unit that makes autonomous decisions based on behavioral models and its environment. The emerging behavior resulting from the

interactions of single agents enables the study of system parameters such as traffic flow.

Urbanization also leads to an increase in road network size, which in turn increases the complexity of solving traffic signal control problems. For example, the city state

¹School of Computer Science and Technology, Nanyang Technological University (NTU), Singapore

²Modeling and Simulation Group, Institute for Visual and Analytic Computing, Universität Rostock, Germany

³MoVES (Mobility in Virtual Environments at Scale) laboratory, TUMCREATE Limited, Singapore

⁴School of Computation, Information and Technology, Technical University of Munich, Germany

Corresponding author:

Wen Jun Tan, School of Computer Science and Technology, Nanyang Technological University (NTU), 50 Nanyang Avenue, 639798 Singapore.
Email: wjtan@ntu.edu.sg

of Singapore has a road network comprising more than 3,440 roads and 164 km of expressways,⁴ including 2449 traffic signal-controlled intersections.⁵ A city-wide traffic signal control problem can be considered as a *large-scale spatial optimization*. Traffic signal timing optimization has been solved using methods based on metaheuristic methods such as genetic algorithm (GA),^{6–8} particle swarm optimization,^{9,10} or meta-models.^{11,12} We find, however, that these methods have mainly been employed to optimize only a single or a few intersections.¹³ The literature on the optimization of traffic signals on city-scale road networks is still scarce.

The computational complexity of large-scale traffic signal control problems is high since the constraints and the objective functions associated with the problems are resource intensive and the size of the decision or objective space is large.¹⁴ Hence, the application of metaheuristic algorithms to the traffic signal optimization of large-scale road networks can be computationally expensive.¹⁵ The large search space of the problem (i.e., traffic signals) makes naive sequential optimization methods infeasible due to their slow convergence.¹⁶ In addition, traffic signal coordination is an EXP-complete problem,¹⁷ which renders the optimization of large traffic networks in cities intractable in practice using naive optimization methods. Moreover, when applied at a large-scale, microscopic traffic simulation still faces computational challenges.¹⁸

One way to solve the large-scale optimization problems is to split them into smaller subproblems.¹⁹ First, the original problem is decomposed spatially into multiple subproblems. Then, the subproblems are optimized in parallel via an individual evolutionary algorithm (EA) to obtain the partial solutions. The global solution is obtained by combining the partial solutions. When evaluating the partial solutions using simulation, parallel simulation can also be used.

There are several advantages in the divide-and-conquer strategy for optimization. First, decomposition of the problem allows the subproblems to be solved in parallel, which speeds up the optimization process. Second, through the proper decomposition of the problem, the “curse of dimensionality,” i.e., the rapid deterioration in performance with an increase in the number of decision variables, can be alleviated to some extent.²⁰ Finally, solving each subproblem separately maintains good solution diversity²¹ and increases the robustness of the whole optimization process against the errors and failures in dynamic environments.²²

However, there can be coordination issues using a naive divide-and-conquer strategy. Due to the complex interdependencies in most practical optimization problems, combining partial solutions may result in a poor-quality global solution.²³ Without considering interaction among regions, although optimal partial solutions could be obtained for each region, they would not form an optimal global solution when being simply put together. The synchronization between partitions is also another issue as it becomes a

bottleneck when the scale of the network is large or when there is a strong coupling between the traffic signals.²⁴

To resolve the coordination and synchronization issues, we proposed a novel parallel simulation-based optimization method called *spatially iterative coordination for parallel optimization* (SICPO) for traffic signal optimization. Through an initial simulation, the optimization scenario is partitioned spatially into regions according to the interactions sampled from the simulation. The interactions are also used to identify the interdependencies between the regions. Based on the regions, the problem is divided into subproblems, where each subproblem optimizes a region separately using an EA. To coordinate among the subproblems, the interactions between partial solutions are synchronized in two ways. First, multiple iterations of the optimization process are executed so that new interactions can be sampled using the improved global solution from each iteration. Second, the partial solutions can also synchronize the interactions during the EA based on the interdependencies between the regions. Hence, multiple iterations of optimization are executed until the global solution converges.

Our contributions can be summarized as follows:

- **Problem decomposition:** Spatial decomposition is used to partition the optimization scenario, i.e., road network and traffic simulation, into regions which is used to decompose the problem into subproblems. The spatial decomposition also determines the grouping of the decision variables based on their spatial positions and the interdependencies of the subproblems based on the interactions between regions.
- **Coordination among partial solutions:** To reduce synchronization between the regions and coordinate among the partial solutions, the interactions are synchronized after each iteration and during subproblem optimizations.
- **Reducing computational complexity:** To reduce the computational complexity, parallelism for the optimization method can be applied on two levels: (1) each subproblem is optimized concurrently, and (2) each partial solution for a region is evaluated in parallel. During the evaluation of the global solution, parallel simulation can also be used.
- **Applicability to real-life problems:** A real-world road network of Singapore is used to demonstrate the applicability of our approach.

The remainder of the paper is structured as follows: We introduce the traffic signal control problem in Section 2. We propose our novel parallel method for large-scale traffic signal optimization in Section 3. Section 4 evaluates the convergence and runtime of the new method. Section 5 discusses the trade-off and limitations in our method. We

discuss related works in Section 6. Section 7 concludes the paper and discusses the avenues for future work.

2. Traffic signal control problem

In this section, we formulate the traffic signal control problem based on the road network model used as described in Viswanathan et al.²⁵ A traffic signal phase is defined as “the right-of-way, yellow change, and red clearance intervals in a cycle that is assigned to an independent traffic movement or combination of traffic movements.”²⁶ A common cross-intersection with two-directional traffic is shown in Figure 1, where there are four traffic signal phases. The directions of traffic allowed in each phase are shown in the figure. In phase 1, only the traffic indicated by the red arrows is allowed. All other traffic directions are shown a red light. Switching from phase 1 to phase 2 means switching the previously green traffic directions to red and showing a green light to the traffic directions indicated by green arrows in the illustration. Hence, the traffic controller will rotate the green phase among the phases depending on the time allocated for each phase. The *cycle time* is the time required to complete the sequence of phases, i.e., time to switch from phase 1 to phase 4, and back to phase 1.

We consider the problem of generating traffic signal patterns that minimize the average travel time of vehicles by controlling the timings of green phases at a set of intersections. Assuming the overall cycle time is fixed, the green time ratio is defined as the ratio of effective green time of a phase to the cycle time. To formulate the problem of traffic signal control, X is defined as a vector of real values, representing the green time ratio for each of the phases in the traffic signal intersections in a road network. \mathbf{I} is the set of all traffic signal intersections in a road network. For each intersection $\mathbf{i} \in \mathbf{I}$, \mathbf{p} refers to the phase number of the traffic signal, where \mathbf{P}_i is set of phases for intersection \mathbf{i} . $X[\mathbf{i}, \mathbf{p}]$ refers to the green time ratio of intersection \mathbf{i} for phase \mathbf{p} .

The traffic signal optimization problem can be formulated as:

$$\begin{aligned} & \arg \min f(X) \\ & \text{s.t. } X[\mathbf{i}, \mathbf{p}] \in \mathbb{R}^+, \forall \mathbf{i} \in \mathbf{I}, \forall \mathbf{p} \in \mathbf{P}_i \\ & \sum_{\mathbf{p} \in \mathbf{P}_i} X[\mathbf{i}, \mathbf{p}] = 1, \forall \mathbf{i} \in \mathbf{I} \\ & X[\mathbf{i}, \mathbf{p}] \geq t_{min}, \forall \mathbf{i} \in \mathbf{I}, \forall \mathbf{p} \in \mathbf{P}_i. \end{aligned} \quad (1)$$

where the objective function $f(X)$ represents the result of a simulation run that returns the average travel time under the green time ratios X , and t_{min} is the minimum ratio for a phase. For example, given a cycle time of 60 s, an intersection has four phases, where phases 1, 2, 3, and 4 have 20 s, 10 s, 20 s, and 10 s green time, respectively. The green time ratios for the intersection 1 are $[\frac{1}{3}, \frac{1}{6}, \frac{1}{3}, \frac{1}{6}]$. To reduce

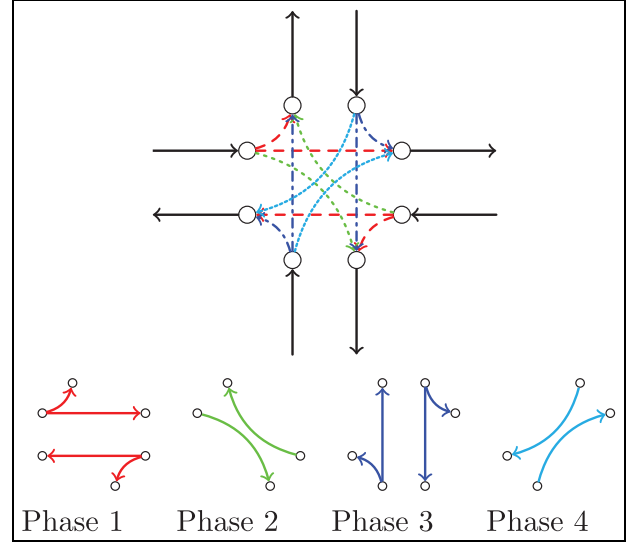


Figure 1. Cross-intersection with four traffic signal phases.

the dimensionality of the problem, the traffic signal offsets are not considered in this problem.²⁷

3. Spatially iterative coordination for parallel optimization

First, we described the decomposition of the traffic signal control problem. Then, we explain our proposed method for improving coordination and reducing synchronization for city-scale traffic signal optimization in detail.

3.1. Decomposition of traffic signal optimization

The objective function $f(X)$ returns the average travel time given the green time ratios X as the objective function parameter, which is obtained through traffic simulation. The data for the traffic simulation also include input parameters such as the directed graph G representing the road network and the travel demand given by a set of trips \mathbb{T} . The trips represent the interactions in the simulation. A trip is represented by a triple (t_d, v_o, v_d) consisting of the departure time t_d , origin node v_o , and destination node v_d . Each trip is handled by an agent in the simulation, where the agent represents a vehicle that starts a trip according to the departure time from the origin node to the destination node. The traffic simulation can be formulated as:

$$\mathbb{R} = \text{sim}(G, \mathbb{T}, X) \quad (2)$$

where the set of trajectories for each agent, \mathbb{R} , are returned as the simulation output. A trajectory is a sequence of tuples (t, v) , indicating the agent reaching node v at time t . Given a trajectory R is a sequence of tuples $(t_0, v_0), \dots, (t_{n-1}, v_{n-1})$, the total travel time of a

trajectory \mathcal{T} can be calculated as $t_{n-1} - t_0$. Hence, the objective function $f(X)$, which determines the fitness Y given a solution X , returns the average travel time \bar{T} for all agents in the simulation.

Spatial decomposition is used to divide the road network into regions and the objective function into many partial functions, such that each partial simulation is represented by:

$$\mathbb{R}_j = \text{sim}(G_j, \mathbb{T}_j, X_j) \quad (3)$$

The road network G is decomposed spatially into a set of regions \mathbb{G} , where $G_j \in \mathbb{G}$ is a subgraph of G . X_j is the set of green time ratios for the traffic signals located in the region G_j . Similarly, we can calculate the average travel time for all agents in a region G_j using the trajectories \mathbb{R}_j . Then, the optimization problem for each partial function, f_j , can be formulated as:

$$\begin{aligned} & \text{argmin} f_j(X_j) \\ & \text{s.t. } X_j[\mathbf{i}, \mathbf{p}] \in \mathbb{R}^+, \forall \mathbf{i} \in \mathbf{I}_j, \mathbf{p} \in \mathbf{P}_i \\ & \sum_{\mathbf{p} \in \mathbf{P}_i} X_j[\mathbf{i}, \mathbf{p}] = 1, \forall \mathbf{i} \in \mathbf{I}_j \\ & X_j[\mathbf{i}, \mathbf{p}] \geq t_{\min}, \forall \mathbf{i} \in \mathbf{I}_j, \forall \mathbf{p} \in \mathbf{P}_i. \end{aligned} \quad (4)$$

where \mathbf{I}_j is the set of intersections in region G_j . After partitioning the interactions among the regions, each partial function can be optimized separately to return the optimized green time ratio X_j .

To partition the interactions among the regions, the set of trips \mathbb{T} is divided spatially, where the time and node where each agent crosses a region are required. First, the whole network is simulated to obtain a set of trajectories \mathbb{R} taken by each agent based on the trips \mathbb{T} . Based on the partition boundaries, the trajectories taken by each agent are cut into segments. When an agent crosses the partition boundary, the time and node of the crossings are used to generate the set of subtrips \mathbb{T}_j . For example, an agent crosses from regions G_1 to G_2 based on a trip (t_a, v_a, v_c) , where $v_a \in G_1$ and $v_c \in G_2$. After executing a simulation on the whole road network G , the agent crosses from regions G_1 to G_2 at node v_b at time t_b based on the trajectory of the agent. Hence, the trip is divided into two segments (t_a, v_a, v_b) and (t_b, v_b, v_c) , and returned as a subtrip for the respective regions. The interdependencies between the regions can also be identified based on the outgoing trips from a region. By recording the outgoing trips from a region, we can also identify the neighboring regions that are dependent on the region.

The departure time of a subtrip depends on the sampling of the trip that is done on the whole network. After performing the partial function optimization, the optimized green time ratios for each partition are combined. The next iteration of the sampling process utilizes the optimized green time ratios in the simulation. Hence, the sampling

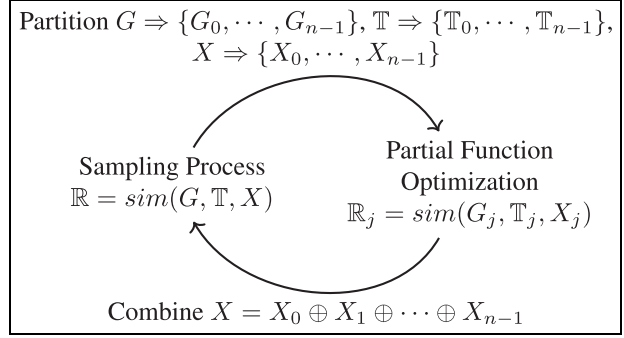


Figure 2. Cyclic dependency between the sampling process and partial function optimization. Initial sampling will obtain a set of trajectories \mathbb{R} that will be used to generate the subtrips \mathbb{T}_j . Partial function optimization will generate new green time ratios X_j , which are combined and fed back to the sampling process.

process is also dependent on the optimized green time ratios obtained in partial function optimization. This creates a cyclic dependency between the sampling process and partial function optimization, as shown in Figure 2. Therefore, an iterative approach is required.

3.2. Method overview

An overview of the proposed method is shown in Figure 3, where simulation is used for the evaluation of the objective function. First, an *iterative search* is performed on the objective function to improve the solution across iterations until reaching convergence. Second, a *parallel search* decomposes the problem spatially into subproblems, i.e., dividing the objective function into partial functions, that can be optimized in parallel. Third, *partial function optimization* optimizes the partial function to search for better partial solutions using an EA, where each solution is evaluated in parallel. After the partial function optimization, partial solutions are combined to obtain a new solution for the objective function. The partial solutions can be coordinated at the end of each parallel search (see Figure 3(a)) and cooperatively co-evolve among partial function optimizations (see Figure 3(b)). The new solution is evaluated and iteratively improved until convergence is reached. The following subsections describe each process in detail.

3.3. Iterative search

At each iteration, optimization will be applied to the current solution to find a new solution, and the fitness of the new solution is evaluated. The new solution is accepted if there is a significant improvement in the fitness of the solution. Otherwise, the optimization process terminates.

Iterative Search is the first process flow shown in Figure 3. Initially, the initial solution X^0 is randomly

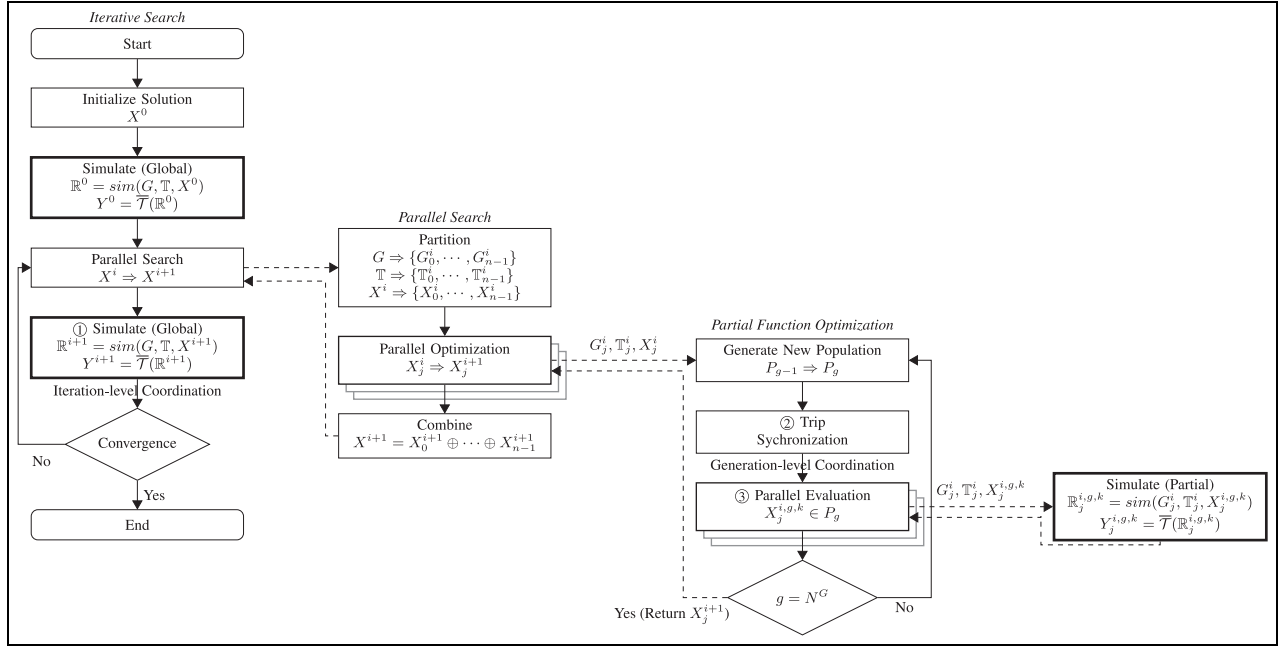


Figure 3. Our proposed method can be broken down into three process flows: (a) iterative search, (b) parallel search, and (c) partial function optimization. The multiple stacked rectangles are parallel processes. Bold rectangles depict the simulation processes.

generated. The initial solution X^0 is evaluated by simulating the whole scenario to obtain the initial fitness, Y^0 . Parallel search is performed on the current solution X^i to find a new solution, X^{i+1} . The new solution is evaluated using the whole scenario to obtain the new fitness, Y^{i+1} . This evaluation also represents *iteration-level coordination* across regions (Figure 3(a)). The coordination is done by using the trajectories from the simulation output to construct the subtrips for the next iteration, as described in Section 3.1. Hence, the subtrips are updated with better arrival times from neighboring regions.

The convergence of the algorithm is determined by the relative improvement, ΔY^{i+1} , which is measured by:

$$\Delta Y^{i+1} = \frac{Y^i}{Y^{i+1}} \quad (5)$$

If the improvement is not better by a threshold of Θ^I , the algorithm will terminate and return the new solution, X^{i+1} .

3.4. Parallel search

Parallel Search is the second process flow shown in Figure 3, which returns improved green time ratios, X^{i+1} , based on the current solution X^i . Spatial decomposition is achieved by partitioning the whole road network G along with input X^i and trips \mathbb{T} based on the intersection flows and the number of traffic signals. The traffic flow at each node is the number of agents crossing the node per hour.

Based on the trajectories, \mathbb{R} , obtained from the simulation of the entire road network (Figure 3(a)), the traffic flows can be calculated based on the trajectories taken by all agents. Hence, the intersection flow is the sum of the traffic flows at an intersection.

An edge cut is determined on the road network using METIS²⁸ given the maximum number of regions (N^R), with the intersection flows and the number of traffic signals as the vertex weights. This aims to balance the number of decision variables (i.e., number of traffic signals) and workload per region. If each region only consisted of a single traffic signal, many trips would need to be synchronized among the regions. Moreover, there will be large discrepancies between the partial solutions and global solution due to the lack of cooperation between the partial function optimizations. On the contrary, if the number of regions is small, the evaluation time per region will be high due to the large regions and more iterations will be required for convergence. If there is an imbalance in the weights, METIS can return fewer than the maximum number of regions.

The subtrips are generated based on the partition boundaries, as described in Section 3.1. After spatial decomposition, each region is optimized in parallel using partial function optimization. As the traffic flows improve over the iteration, the road network needs to be re-partitioned for every iteration. Due to the decomposition, the dimension of the parameter space is divided by the number of regions. The reduction in the dimensions of the parameter

space can enable the search to converge faster. The level of parallelism is equal to the number of partial functions.

Usually, a reduction operation is performed on the results returned by the optimization of partial functions, such as summation of the results. With SICPO, the set of partial solutions is combined using a concatenation operation (\oplus) to obtain the new solution X^{i+1} . The coordination between the partial solutions is performed at the end of the partial function optimization when the new solution is evaluated (Figure 3(a)).

3.5. Partial function optimization

Each partial function is optimized separately using an EA. In this paper, we used the genetic algorithm (GA) as the EA. A population of candidate solutions representing the partial problem is evolved toward better solutions. Each solution has a chromosome containing a set of values representing a vector of green time ratios in each partitioned region.

The generation index is defined as g . First, the initial population P_0 of N^P solutions is randomly initialized. Each solution k in the population P_g represents the green time ratios $X_j^{i,g,k}$ in iteration i and region j , $P_g = [X_j^{i,g,0}, X_j^{i,g,1}, \dots, X_j^{i,g,N^P-1}]$. Next, the fitness of the population P_g is evaluated in parallel. The fitness function is a partial function $f_j(X_j^{i,g,k})$ that evaluates the average travel time for all agents traveling through region G_j based on the green time ratios $X_j^{i,g,k}$ and subtrips \mathbb{T}_j . Since each solution can be evaluated independently, the fitness evaluation can be carried out in parallel to reduce the running time per generation without the need for synchronization.

After the fitness evaluation, genetic operations are applied to generate the next population P_{g+1} . First, Hall-of-Fame (HoF) is applied to retain the top N^H individuals for the next generation. Tournament selection is then used to select the remaining individuals for evolution, where N^T individuals are drawn from the current population and the best individual is selected. Next, one point crossover of probability ρ^c is applied to the selected individuals, where a single point is chosen and genes from the two parents on the right of the point are swapped to generate the new offspring. Then, uniform mutation of probability ρ^m is performed where each gene of the selected individual has a probability of 0.1 to select a new green time ratio between 0 and 1. After mutation, the green time ratios are normalized to ensure the sum of the green time ratio is one. The newly generated offspring are combined with the HoF individuals to form the next generation. This is repeated until the maximum number of generations N^G is reached. The solution with the best fitness value over the entire GA run is returned.

3.6. Trip synchronization

As the regions are not synchronized during the solution evaluations, there is no coordination among the partial function optimizations. For example, assume an agent crosses from regions G_1 to G_2 . The departure time of the agent at G_2 depends on the arrival time of the agent at the partition boundary in G_1 . When the optimization improves the traffic flows in G_1 , the agent can arrive at the boundary earlier (or later). However, since the coordination between the regions is only done at the iteration level at $\textcircled{1}$ in Figure 3, this will result in the departure time of the agent at G_2 to be out of date. To address this issue, additional *generation-level coordination* is introduced during the partial function optimizations to synchronize between the regions. The coordination is done by updating the trips based on the interdependencies between the regions, i.e., the newest arrival time from the neighboring regions. There are two modes of generation-level coordination: *synchronous* and *asynchronous*. However, generation-level coordination cannot fully replace iteration-level coordination. The best arrival times are only achieved when the partial solution converges to their local minimum. Hence, these timings can no longer influence the convergence of the partial function optimization. Although the generation-level coordination can reduce the discrepancies between the partial solutions, the global solution still requires an iterative approach to coordinate between the regions.

In *synchronous* generation-level coordination, all partial function optimizations are executed synchronously and synchronized in every generation. The arrival time of all trips of the best solution is recorded during the fitness evaluation. At the end of each generation, all the partial function optimizations are synchronized at a barrier. Then, the departure times of the subtrips are updated based on the arrival time from the neighboring regions. For example in Figure 4(a), the arrival times are updated synchronously between region G_1 and region G_2 . Since the workload in G_2 is higher than G_1 , G_1 needs to wait for synchronization before exchanging the updated arrival times and only continues the next generation after synchronization. There will be an overhead due to the barrier synchronization.

In *asynchronous* generation-level coordination, all partial function optimizations are executed asynchronously and synchronized independently of other regions. As the partial optimization improves the travel time of the subtrips, the partial results are fed into the neighboring regions and the neighboring regions only update the departure times of the subtrips in the next generation. Each region does not need to wait for partial results from each generation of their neighboring regions and continues the next generation based on the last updated results. In Figure 4(a), the arrival times are updated asynchronously between

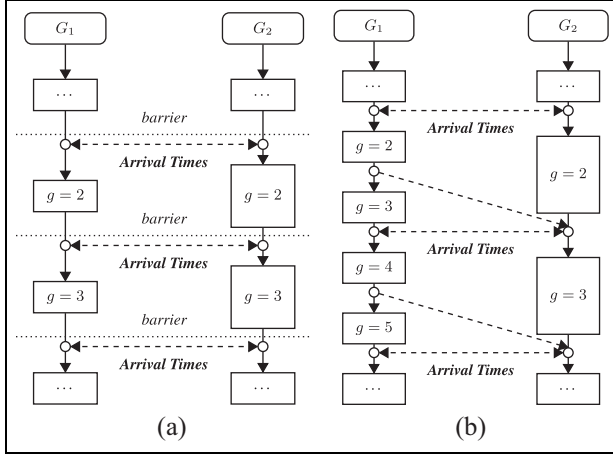


Figure 4. Trip synchronization between region G_1 and region G_2 . Each rectangle represents the evaluation workload in a generation, where the rectangle height represents the evaluation time: (a) synchronous coordination occurs at every generation and (b) asynchronous coordination occurs independently of other regions.

region G_1 and region G_2 , where the evaluation workload for G_1 is smaller compared with G_2 . When generation 2 in G_1 completed execution, the arrival times at the boundaries are sent to G_2 . However, since G_2 is still executing generation 1, G_1 continues to execute generation 3 based on the last updated results from G_2 (generation 1). On the contrary, G_1 only receives the new timings from G_2 at generation 4. As there is no barrier synchronization, this reduces the synchronization overhead compared with synchronous coordination. In addition, asynchronous coordination allows regions with a smaller workload to complete earlier. However, due to the reduced synchronization, more generations may be needed comparing to synchronous coordination.

4. Experiments

We conduct experiments to evaluate the effectiveness of the coordination methods on the convergence and optimization time for a city-scale scenario. In these experiments, a parallel microscopic traffic simulator CityMoS¹⁸ is used to simulate vehicular traffic. The agent-based model and traffic assignment model are described in Viswanathan et al.²⁵ Two road networks are considered: a synthetic grid-shaped road network and the large-scale road network of Singapore. We apply three variations of coordination methods for optimization (see Table 1): (1) SICPO with only iteration-level coordination, (2) SICPO with iteration-level coordination and synchronous generation-level coordination (SICPO-S), and (3) SICPO with iteration-level coordination and asynchronous generation-level coordination (SICPO-A).

Table 1. Three variations of coordination.

Methods	Coordination	
	Iteration level	Generation level
SICPO	✓	—
SICPO-S	✓	Synchronous
SICPO-A	✓	Asynchronous

SICPO: spatially iterative coordination for parallel optimization.

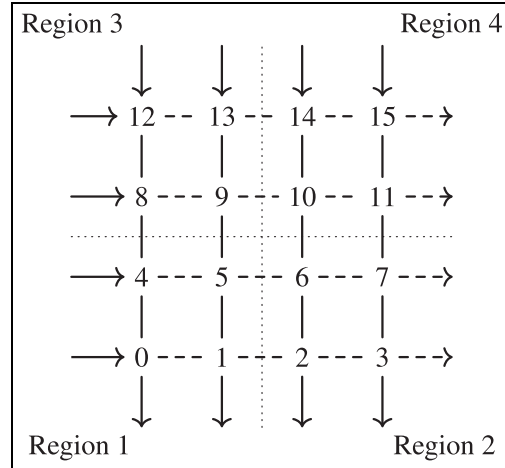


Figure 5. A 4×4 synthetic grid-shaped road network with 16 traffic signal intersections. Directional arrows indicate the flow of traffic, where solid lines are high traffic flows, and dashed lines are low traffic flows.

4.1. Grid network

To investigate the effect of different coordination methods on the optimization convergence, we evaluate a small-scale scenario using a synthetic grid-shaped road network. This experiment was executed on a workstation equipped with an E5-1650 v2 CPU running at 3.5 GHz and 16 GB of random access memory (RAM). Figure 5 shows a synthetic grid-shaped road network, which comprises a 4×4 regions with each edge being 100 m in length. The grid network contains 16 traffic signal intersections. Each intersection has four traffic signal phases, as shown in Figure 1. The traffic flows in the grid network are generated based on predefined origin-destination pairs and the interarrival time for the agents. The scenario is executed for a total of 6 h of simulated time. The optimizations are executed 30 times for each scenario to obtain confidence intervals.

The grid network is partitioned into four 2×2 grid networks. Figure 5 shows the partition boundary dividing the grid network into four regions, represented by the dotted lines. Region 1 comprises $\{0, 1, 4, 5\}$ intersections, region 2 with $\{2, 3, 6, 7\}$ intersections, region 3 with $\{8, 9, 12, 13\}$ intersections, and region 4 with $\{10, 11, 14, 15\}$ intersections.

Table 2. Scenario shows average wall-clock time, CPU time, and convergence of different methods across iterations.

Method	Iter.	Wall-clock time [s]	CPU time [s]	Best objective value
SICPO	1	200	6800	941 ± 342
	2	214	6886	72 ± 1
SICPO-A	1	238	6872	576 ± 302
	2	236	6920	69 ± 2
SICPO-S	1	246	7058	376 ± 240
	2	233	7451	69 ± 2

SICPO: spatially iterative coordination for parallel optimization.

The routes for scenarios are also shown in Figure 5. There are two directions of traffic flows: vertically from top to bottom and horizontally from left to right. The inter-arrival time for the agents is 20 s on each traffic flow. Initially, the green time ratios for intersections $\{0, 4, 8, 12\}$ are not optimized for the horizontal traffic flow from left to right. Hence, the heaviest congestion occurs in front of these intersections. The solid lines in Figure 5 indicate large amounts of traffic flow (approximately three cars per minute), and dashed lines indicate small amounts of traffic flow (approximately one car per minute).

The grid network scenario is used to support the following statements: (1) the iterative approach to optimization is necessary to coordinate across regions to achieve a higher quality global solution, (2) generation-level coordination during the partial function optimization can improve the convergence to a better objective value. Table 2 shows the average wall-clock time, total CPU time, and convergence of each method across iterations.

4.1.1. Iteration-level coordination. We examine the convergence of SICPO across iterations in the *Iterative Search* (see Figure 3). First in Iteration 0, a random solution is generated initially. The initial solution is evaluated on the whole grid network with the nonoptimal green time ratios for intersections. At the same time, the simulation output from the evaluation will be used in the optimization in the next iteration. The objective value obtained is 22,430.

Next in Iteration 1, the parallel search is performed on the grid network. The trajectories from the simulation output in Iteration 0 will be used to construct the subtrips according to the region boundaries. Partial function optimization is applied to each of the regions. The optimized green time ratios from each region are combined to obtain the overall solution. The new solution is evaluated on the whole grid network to determine the quality of the solution. This is repeated for Iteration 2.

Since the traffic flows change after each iteration of optimization, additional iteration is required to synchronize the trips across the regions. After optimization, the green time ratios at intersections $\{0, 4, 8, 12\}$ improve, resulting in higher horizontal traffic flow. After Iteration 1 of

SICPO, the objective value decreases to 941. The quality of the solution is limited by the low traffic flow sampled in Iteration 0. The green time ratios from Iteration 1 are evaluated on the whole scenario and generate the subtrips for Iteration 2. After Iteration 2, the objective value further reduces to 72.

The results show that a single iteration of the optimization is only able to optimize for trips sampled from the previous iteration. By iteratively optimizing using the trips sampled from the previous iteration, the solutions are able to converge to higher quality ones.

4.1.2. Generation-level coordination. Generation-level coordination is used in SICPO-A and SICPO-S, where they show to have better objective values compared with SICPO. After Iteration 1 of optimization, SICPO-A and SICPO-S have better objective values of 576 and 376, respectively. After Iteration 2, both SICPO-A and SICPO-S converged to an objective value of 69. However, the additional coordination at the generation level does introduce an overhead due to the synchronization to update the neighboring regions regarding the improved trip timings crossing the partition boundary. In Iteration 1, SICPO-S has a slightly higher wall-clock time and CPU time compared with SICPO-A.

To examine the impact of generation-level coordination, the number of trips with departure time updated at every generation is shown in Figure 6. If the trip from the neighboring region has improved arrival time, this improved time is updated across the regions to generate new departure time.

Figure 6(a) shows the number of trips synchronized in Iteration 1 of SICPO-S. Region 2 shows the highest number of trips synchronized as the traffic flows are incoming from regions 1 and 4. Region 3 does not have any trip synchronized as there are no incoming traffic flows. Region 4 shows to have more trips synchronized than region 1. As the optimization progresses, the improvement of the green time ratios at intersections $\{0, 4, 8, 12\}$ improves the horizontal traffic flows. Hence, more agents cross the boundary from region 3 to region 4, thereby increasing the number of coordination. Iteration 2 of SICPO-S is shown

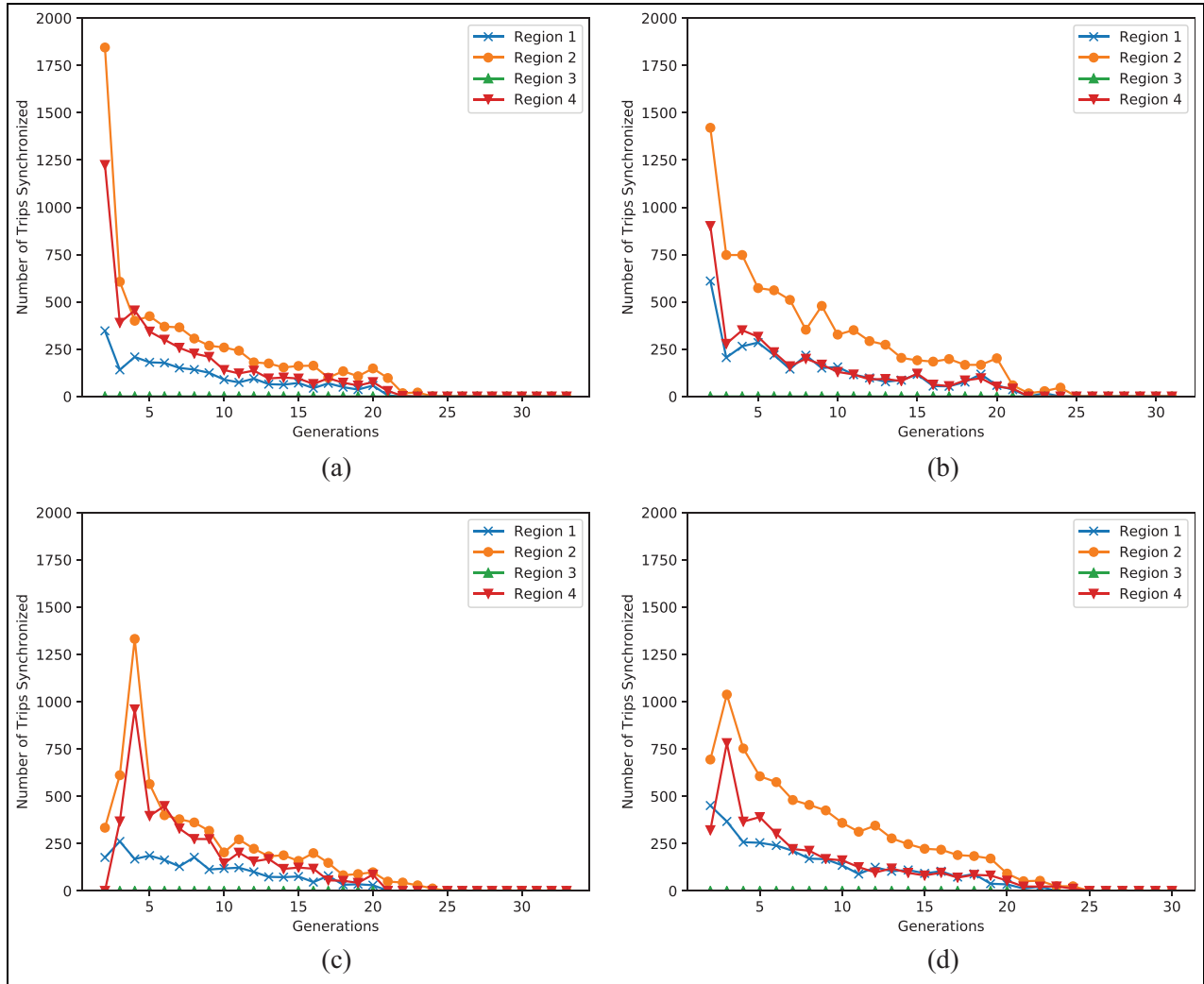


Figure 6. Number of trips with departure time updated across generations: (a) SICPO-S—Iteration 1, (b) SICPO-S—Iteration 2, (c) SICPO-A—Iteration 1, and (d) SICPO-A—Iteration 2.

in Figure 6(b). Region 2 has an increased number of trips synchronized due to more trips being sampled after the improvement in the green time ratios after optimization in Iteration 1.

In Iteration 1, SICPO-A has a lower quality solution (576) compared with SICPO-S (376). Initially, regions 2 and 4 have lower traffic flows (as shown in Figure 5). The evaluation of partial solutions for regions 2 and 4 has a shorter wall-clock time due to low traffic flows. Therefore, the partial function optimization for regions 2 and 4 progresses with a few generations before they coordinate with regions 1 and 3, and the number of trip synchronizations initially is low. After a few generations of optimization in regions 1 and 3, the improved travel time is reflected in the increase in the number of trips synchronized in regions 2 and 4. This explains the peak of the number of trip synchronizations for regions 2 and 4 as observed in Figure 6(c) and (d).

These results show that generation-level coordination can improve the convergence of the optimization, where SICPO-A and SICPO-S generate better objective values compared with SICPO as shown in Table 2. The improvements are due to the additional synchronization of trip timings between the regions, as shown in Figure 6. SICPO-A achieves a solution with similar quality as SICPO-S but using slightly lesser resources (i.e., CPU time). The overheads of the synchronization are reflected by the increased wall-clock time and CPU time.

4.2. City-scale network

After the experiments on the grid network, we evaluate SICPO on a city-scale road network to demonstrate convergence and optimization time on a large-scale problem. The road network and origin-destination matrix used in the city-scale scenario is based on real-world data. We rely on

Table 3. Algorithm parameters.

Parameter	Symbol	Value
Iteration threshold	Θ^I	3%
Maximum number of regions	N^R	128
Population size	N^P	100
Hall of fame size	N^H	10
Tournament size	N^T	3
Crossover probability	ρ^c	30%
Mutation probability	ρ^m	10%
Maximum generation	N^G	50

a representation of the road network of Singapore city from Viswanathan et al.,²⁵ which comprised of 152,032 nodes and 224,312 edges. The road network also contains a total of 6557 signal phases of 1115 traffic signal-controlled intersections. Singapore's Land Transport Authority²⁹ conducts a travel diary survey, Household Interview Transport Survey (HITS), once every 4 years. Origin-destination pairs are derived from the results of the 2008 HITS data set. We simulate 1 day's afternoon and evening traffic of

6 h from 3 p.m. to 9 p.m. In our scenario, the maximum number of agents during the peak traffic hours of the day is approximately 50,000.

The experiment is executed on the ASPIRE-1 cluster of the National Supercomputing Centre (NSCC), Singapore, which comprises 1288 nodes, each equipped with two 12-core E5-2690 v3 CPUs running at 2.6 GHz and 96 GB of RAM. The algorithm parameters used in SICPO are shown in Table 3, based on suggested values from Hassanat et al.³⁰ For a lower population size, a higher mutation rate is used to increase the random search. The crossover rate is defined to be higher than the mutation rate. The maximum number of regions, N^R , is currently limited by the scheduler constraint on the number of concurrent jobs. Figure 7 shows the city-scale road network partitioned into regions (different colored edges) with the traffic signal intersections (as black nodes).

Each generation of the partial optimization is allocated on two compute nodes (four CPUs with 48 cores in total), where a total of 48 evaluations can be executed concurrently. Each partial solution is evaluated by traffic simulator using only a single thread. For maximum parallelism

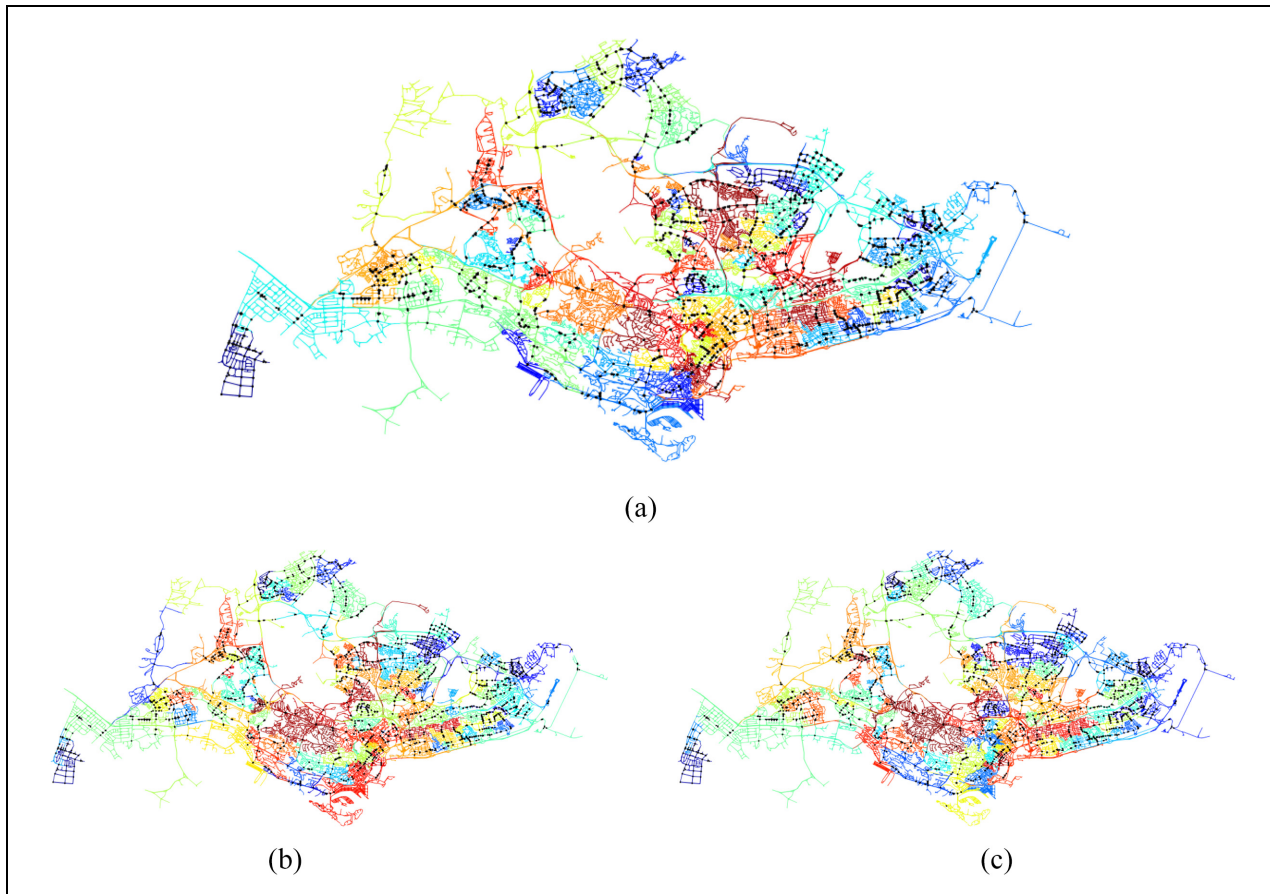


Figure 7. Road network regions with traffic signal intersections (nodes in black). The network is partitioned into different regions across iterations due to the changes in the traffic flows: (a) Iteration 1, (b) Iteration 2, and (c) Iteration 3.

Table 4. Overall improvements of different methods across iterations.

Method	Iteration	Optimization wall-clock time [s]	Global evaluation time [s]	Av. travel time [s]	Overall improvement	Relative improvement
SICPO	0	—	480	2188	—	—
	1	984	385	1450	1.50	1.50
	2	1395	407	1413	1.54	1.03
SICPO-A	3	1320	359	1411	1.55	1.00
	1	2970	391	1416	1.54	1.54
	2	3875	356	1371	1.60	1.03
SICPO-S	3	2606	378	1332	1.64	1.02
	1	2852	397	1382	1.58	1.58
	2	3511	388	1293	1.69	1.06
	3	3195	347	1289	1.70	1.00

SICPO: spatially iterative coordination for parallel optimization.

for 128 regions, a total of 256 compute nodes will be used. The execution time of the parallel search is limited by the slowest partial function optimization. To evaluate the global solution, the parallel traffic simulator is used, which utilize all the cores in the same compute node, i.e., 24 cores.

4.2.1. Optimization convergence. The convergence of SICPO is examined by evaluating the relative improvement in the partial results with each new solution found by the parallel search. First, the relative improvement of the solutions in each region is evaluated to analyze the convergence of optimization with respect to the partial objective functions. Based on the new solution found in each iteration, the relative improvement of the whole scenario is evaluated.

The *overall improvement* of the global solution for Iteration i compared with the initial solution X^0 can be computed as:

$$\Delta Y^i = \frac{f(X^0)}{f(X^i)}$$

We also calculate the *relative improvement* of the global solution and partial solution in the region j between Iteration $i - 1$ and Iteration i as follows:

$$\Delta Y^i = \frac{f(X^{i-1})}{f(X^i)} \quad \Delta Y_j^i = \frac{f_j(X_j^{i-1})}{f_j(X_j^i)}$$

In Iteration 0, an initial solution is generated and used for all methods. A total of three iterations of SICPO, SICPO-S, and SICPO-A are executed before reaching the termination criterion where the objective improvement across iterations is less than the iteration threshold (Θ^I) of 3%. The partitions in each iteration are shown in Figure 7, where the regions change due to the re-partitioning in each

iteration. The results shown are obtained from a single optimization run.

The summary of the results over the iterations is shown in Table 4, together with the wall-clock time for the optimization and evaluation. Optimization wall-clock time is the wall-clock time spent in executing the parallel search in an iteration; global evaluation time is the wall-clock time in simulation at \oplus in Figure 3.

For each iteration, the relative improvements of the partial function optimization are shown in Figure 8. Global convergence can be observed by the decrease in relative improvements across iterations. A large change in local improvements, i.e., there are many regions with an increase in improvements or a decrease in improvements, indicates that partial function optimizations only converged locally but combined partial solutions lead to a worse global solution. Hence, trips between the regions need to be synchronized to coordinate among the partial solutions.

4.2.1.1. Iteration 1. The whole road network is partitioned into 117 regions using METIS, as shown in Figure 7(a). Comparing the improvements of each region, all methods exhibit improvements for many regions, as shown in Figure 8(a), (d), and (g) where the improvements are greater than 1.

However, SICPO-A and SICPO-S have lower average relative improvements ($3.65 \times$ and $3.66 \times$) compared with SICPO ($4.06 \times$). As SICPO-A and SICPO-S introduce additional coordination, the optimization of each sub-problem also considers the traffic flows from the neighboring regions. On the contrary, the partial solutions in SICPO tend to converge locally as it does not consider the traffic flows from the neighboring regions.

As shown in Table 4, Iteration 1 shows high relative improvements compared with the initial solution. SICPO-S shows the best overall improvement ($1.58 \times$), followed by SICPO-A ($1.54 \times$) and SICPO ($1.5 \times$). SICPO-S ensures

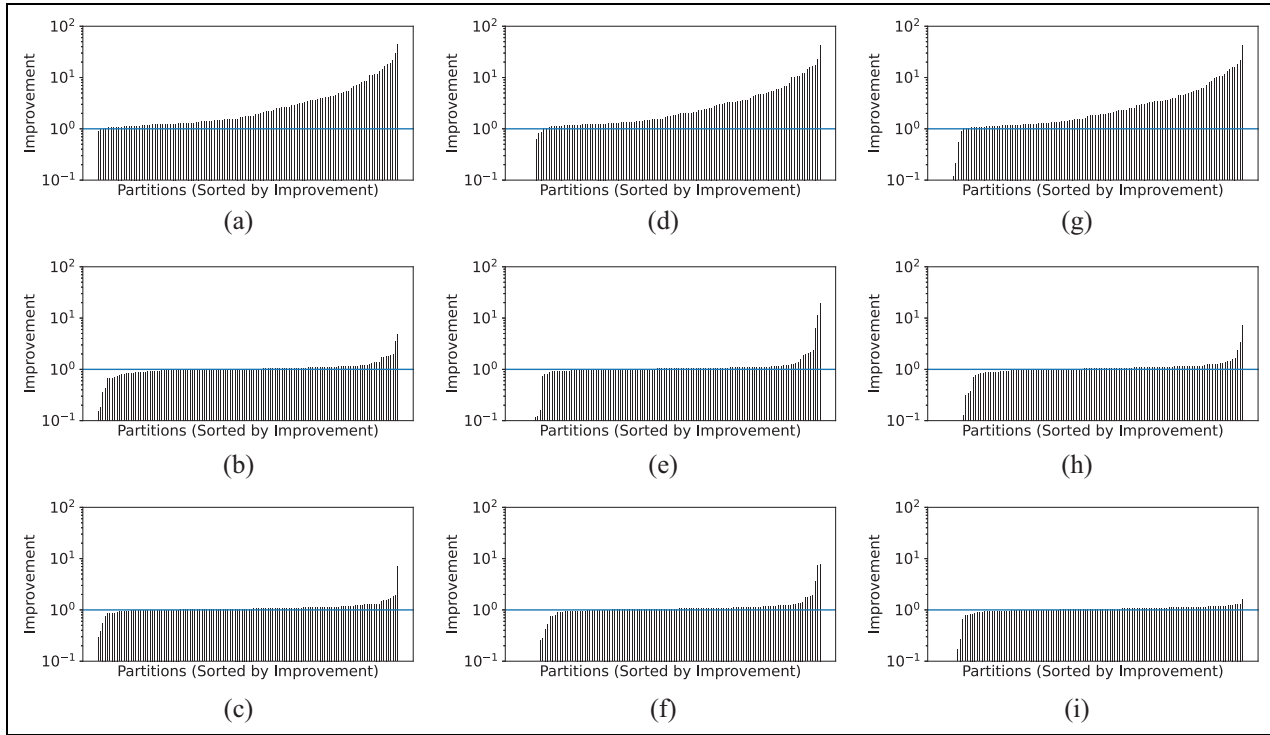


Figure 8. Improvement of solution X_j^{i-1} compared with solution X_j^i across the regions. (a)–(c) shows the improvements for SICPO, (d)–(f) for SICPO-A, and (g)–(i) for SICPO-S. The average relative improvements across regions are also shown in the subcaption.

that all trips are synchronized across regions in each generation, whereas SICPO-A only synchronizes some trips. This shows that better global convergence can be obtained by more frequent synchronization of the subtrips for each region.

4.2.1.2. Iteration 2. The whole scenario is decomposed spatially into 128 regions, as shown in Figure 7(b). As the traffic flows across the road network changes, it affects the partitioning, resulting in different regions compared with Iteration 1.

Compared with the previous iteration, the improvement per region decreases substantially, indicating that the solutions are approaching the global convergence. The average relative improvements are significantly lower compared with Iteration 1. Figure 8(b), (e), and (h) show fewer regions with improvements while some regions show regressions (if the improvement is less than 1). SICPO has more regions with regressions (46 regions) compared with SICPO-A (36 regions) and SICPO-S (41 regions). This indicates that generation-level coordination reduces discrepancies of locally optimal partial solutions by more frequent synchronization of the subtrips for each region.

Table 4 shows overall improvement over the previous iteration, which also indicates that executing only a single iteration is insufficient to achieve convergence of the overall optimization process. In this iteration, SICPO-S still

shows the best overall improvement ($1.69 \times$), followed by SICPO-A ($1.6 \times$) and SICPO ($1.54 \times$).

4.2.1.3. Iteration 3. In Iteration 3, the whole scenario is again decomposed into 128 regions, as shown in Figure 7(c). Figure 8(c), (f), and (i) shows more regions with no significant improvement or regression compared with Iteration 2, indicating global convergence. As shown in Table 4, the relative improvements for Iteration 3 are small compared with the previous iterations. Since the overall relative improvements are less than the threshold (Θ^I) of 3%, the iterative search stops for all methods, with the overall improvement of SICPO-S at $1.70 \times$, followed by SICPO-A ($1.64 \times$) and SICPO ($1.55 \times$).

4.2.1.4. Global convergence. This large-scale scenario shows that an iterative approach for coordination can converge to higher quality solutions. As shown in Figure 8, the improvements over the iterations for the regions reduce as more partial solutions converge and do not improve further. Hence, the relative improvements of the global solution over the iterations also reduce.

Although SICPO can achieve global convergence, there can be large variations in the local convergence across iterations. To resolve this issue, generation-level coordination can reduce the discrepancies of locally optimal partial solutions by synchronizing the trips during partial function

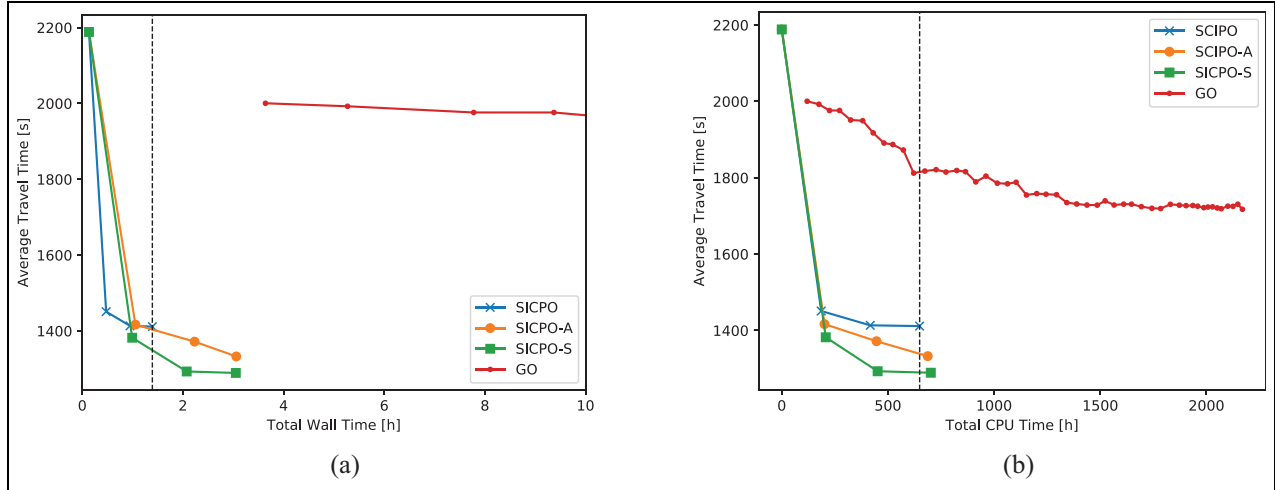


Figure 9. Comparison of convergence of SICPO against GO. Dashed line indicates the time when SICPO terminates: (a) total wall-clock time and (b) total CPU time.

Table 5. Relative wall-clock time, CPU time, and objective value compared with SICPO.

	Relative compared with SICPO		
	Wall-clock time	CPU time	Objective value
GO	62.8	97.6	1.22
Hand-tuned	—	—	1.19
SICPO-A	2.20	1.06	0.93
SICPO-S	2.19	1.08	0.91

SICPO: spatially iterative coordination for parallel optimization; GO: Global Optimization.

optimization, leading to better objective values compared with only iteration-level coordination.

The global evaluation times of the methods are similar, while there is a difference in the optimization wall-clock time due to the generation-level coordination. SICPO-A and SICPO-S show significantly higher optimization time compared with SICPO, showing the trade-off between the overhead of synchronization and computing resource usage for convergence. SICPO-S introduces more overhead in synchronization but may reach a converged global solution faster. On the contrary, SICPO-A introduces less synchronization overhead but may require more time to reach convergence. We conclude that the key to generation-level coordination is to balance this trade-off.

4.2.2. Comparison with global optimization. For comparison, we conducted a global optimization (GO) using GA on the whole road network. GO utilizes generation-level parallelism to evaluate individual solutions concurrently, using the GA parameters from Table 3. Each evaluation of the

individual solution is executed by the traffic simulator using a single thread. The convergence of different methods across total wall-clock time and total CPU time is shown in Figure 9(a) and (b), respectively. GO terminates after 50 generations (~ 3 days 15 h) once no significant improvement is observed across three generations. The results show that due to the large parameter space, the convergence of GO is slow. Figure 9(a) shows when SICPO completes the optimization, GO still executing the first generation of optimization.

4.2.3. Limited budget. Since generation-level coordination has a huge impact on the optimization time, we consider the implications of having a limited budget and determine which methods return a better quality solution. Both the relative wall-clock time and CPU time compared with SICPO are tabulated in Table 5. Total wall-clock time measures the difference between the start and end time of the whole iterative search, which is the sum of the optimization wall-clock time and global evaluation time across iterations in Table 4. Total CPU time measures the sum of the wall-clock time spent on the optimization across all cores. These timings are normalized according to the respective time obtained by SICPO. The last column of Table 5 lists the relative objective value compared with SICPO obtained by the different methods. Furthermore, we compare the results generated by SICPO to hand-tuned green time ratios that were determined iteratively based on a lengthy visual inspection of the traffic simulation. Comparing to SICPO, SICPO-S exhibits the best performance, followed by SICPO-A. SICPO, SICPO-A, and SICPO-S achieve better results compared with the hand-tuned values, which has normalized objective value > 1 .

The multiple levels of parallelism enable SICPO to achieve a high level of concurrency to reduce the overall wall-clock time. Due to the additional generation-level coordination, SICPO-A and SICPO-S have higher total wall-clock time compared with SICPO. This is a trade-off to achieve a slightly better global convergence.

Given a limited time budget where SICPO terminates (shown as the dashed line in Figure 9(a)), SICPO-A and SICPO-S manage to converge to better objective values ($0.99\times$ and $0.96\times$ compared with SICPO). The GO approach is still evaluating the first generation, which has the worst objective value. Comparing to a limited CPU budget when SICPO terminates (shown as the dashed line in Figure 9(b)), SICPO-A and SICPO-S converge to better objective values ($0.95\times$ and $0.91\times$ compared with SICPO). At the same time, the GO method is evaluating the 11th generation, which has the worst objective value ($1.29\times$ compared with SICPO).

In summary, generation-level coordination enables convergence to a higher quality global solution compared with both GO and SICPO given the same CPU time or total wall-clock time. The usage of SICPO-A or SICPO-S depends on the constraints on the computing budget, which is discussed in the next section.

5. Discussion

In this paper, we introduced new coordination mechanism for parallel optimization of traffic signal control. On the small-scale case study of a grid network, convergence to a higher quality global solution can be observed using an iterative approach in SICPO. Similar convergence is also observed on the city-scale network. By coordinating the traffic flows between iterations, it enables the improved traffic flows to be used in the optimization for the next iteration. As the grid scenario exhibits strong coupling between the regions, additional coordination at the generation level helps in converging to a better global solution, especially for synchronous coordination. For the city-scale scenario, SICPO-A or SICPO-S can converge to better objective values compared with SICPO given a limited time or CPU budget. However, there are still some limitations to our proposed approach.

5.1. Partitioning

The optimization time required by the methods is limited by the partitioning approach. Since the regions are optimized in parallel, the overall execution time in each iteration is limited by the slowest region. Figure 10 shows sorted per-region optimization times for Iteration 1 of SICPO. Currently, the intersection flows and the number of traffic signals are used as the vertex weights. This attempts to balance the agent count and the number of nodes in each region, which can affect both the

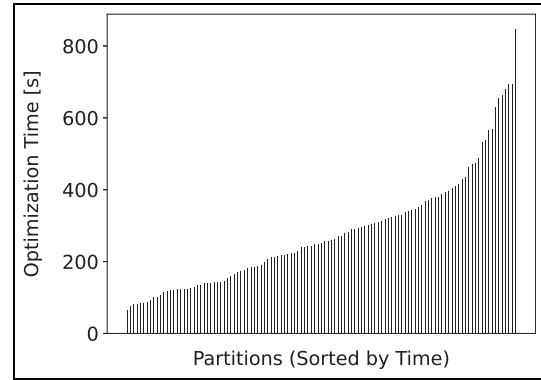


Figure 10. Optimization wall-clock time for each region shows a skewed degree distribution.

convergence and optimization time. However, METIS is unable to return a well-balanced partition, resulting in a skewed degree distribution in the optimization time as shown in Figure 10. The results from Iterations 2 and 3 show similar skewed distribution. Better partitioning may be achieved by assigning vertex weights to represent traffic density. In a heavily congested region, there may be many vehicles making little progress, but the simulator will still invoke their car-following and lane-changing models. A suitable measure of the traffic density in a region can better represent the workload of the agents within a region.

5.2. Workload scheduling

If there is a limit on the available parallel processors such that we cannot achieve full parallelism for the partial function optimizations, there needs to be an execution mechanism to efficiently schedule the tasks (i.e., partial function optimization). A simple master-worker paradigm can be used to execute the tasks using a first-in-first-out (FIFO) strategy. Using a master-work scheduler, SICPO can efficiently execute the partial function optimizations even if the workload between the regions is imbalanced. On the contrary, SICPO-S requires a synchronization barrier at every generation. There can be idle processors waiting at the barrier for the slower regions to complete. Meanwhile, SICPO-A allows regions with a lighter workload to complete first, freeing up the processors for other workloads. Hence, in terms of the overall computing budget, SICPO-A requires a lower computing budget compared with SICPO-S (see Table 5).

6. Related works

GA has been frequently used in the traffic signal optimization. Roupail et al.³¹ optimized the traffic signal of a traffic network with nine intersections using a GA to reduce

the link delay and total network queue time. Sánchez-Medina et al.³² optimized traffic signal using GA with cellular automata-based microscopic simulator on a traffic network of seven intersections. They attempted to optimize several separate optimization objectives such as reducing the travel time and gas emission.

The above approaches have demonstrated good performance in small traffic networks. However, signal optimization of large-scale traffic networks is still very challenging, and only a few related works were presented. García-Nieto et al.³³ attempted to solve large-scale traffic signal optimization problem using a modified Particle Swarm Optimization (PSO) algorithm for traffic networks involving up to 40 junctions. Gao et al.³⁴ evaluated different traffic optimization algorithms on a set of traffic networks involving 9 to 400 intersections to reduce the total delay time of vehicles.

These approaches directly optimize the whole traffic network with an optimization algorithm. As the number of traffic signals increases, this also increases the solution space of the problem, resulting in difficulty in finding satisfying results by tackling the entire traffic network directly. We proposed a spatial decomposition of the traffic signal problem so that parallel EA can be used to reduce the computational complexity. Our method is demonstrated on a road network of more than 1000 intersections, which is significantly larger compared with other works.

6.1. Spatial optimization

Spatial decomposition has also been applied to traffic signal optimization problems. There are existing works that divide the road network into regions containing an intersection and optimize signal timing parameters of each intersection simultaneously.

Timotheou et al.³⁵ modeled the problem as a mixed-integer linear program (MILP) using the cell transmission model (CTM). The problem is decomposed temporally into temporal subproblems with a smaller time window and then further decomposed spatially into spatial subproblems for each intersection. First, the decomposed subproblems are solved in a distributed fashion while exchanging traffic flow information with neighboring intersections. Then, the relaxed partial solutions are combined through distributed rounding schemes, where noninteger decision variables are rounded to Boolean variables for traffic signal control. Mehrabipour and Hajbabaie³⁶ also utilized a MILP and CTM to optimize traffic signal timings, considering oversaturated traffic conditions. After decomposition, each intersection is optimized separately for every time step, where capacity in the downstream intersections and arrival from the upstream intersections are exchanged. A rolling time horizon solution technique is used for optimizing the traffic signal for a time step based on the exchanged traffic flows from the previous time steps.

Other approaches decompose the road network into regions containing multiple intersections. Adacher and colleagues^{37,38} proposed to use a surrogate method based on a platoon model for traffic signal optimization. The road network is decomposed using clustering algorithms, e.g., k-means or Newman clustering algorithm. The nodes in the network are ranked according to their node degrees, betweenness, and flows. Subnetworks are coordinated by connecting to the highest-ranking nodes from the neighboring subnetworks, such that the traffic flows from these neighboring nodes are fed into the subnetworks. Then, the subnetworks are optimized in order of their node ranks, while using the partial solutions from subnetworks of higher rank. Villagra et al.³⁹ utilized cellular genetic algorithms (cGAs) for optimizing traffic lights by structuring the population in a two-dimensional toroidal grid, such that individuals may only interact with their neighbors. The exploration of the search space is smoother due to the slow diffusion of solutions through the population.

Existing works have proposed decomposition approaches to solve traffic signal optimization, such as partitioning into individual intersections,³⁵ clustering approaches,^{37,38,40} or using cellular grids.³⁹ However, the synchronization between the partitions becomes the key bottleneck for large road networks. Frequent communication, such as every time window of 10 min³⁵ or every time step,³⁶ can substantially increase the synchronization overhead in large road networks. Compared with existing works, our proposed method reduces synchronization and improves coordination among the partial solutions by only synchronizing at each iteration and during partial optimization.

Liang et al.⁴⁰ proposed an iterative approach for cooperative signal optimization that is similar to our work. The road network is decomposed into subnetworks and optimized using a surrogate-assisted optimizer. The partial solutions are coordinated by combining them and using the complete solution in the next iteration. Their traffic scenario utilizes predefined turning rates, which allows a more accurate surrogate model. However, changing traffic signals may affect the traffic pattern, limiting the accuracy of the surrogate model. Our approach utilizes a global simulation to evaluate the whole signal plan to handle the changes in the traffic pattern.

6.2. Parallel evolutionary algorithm

An extensive review of parallel EAs is carried out by Talbi,¹⁴ categorizing different levels of parallelism for EAs based on their parallelism granularity. The granularity reduces (i.e., levels of parallelism increase) from algorithmic level, generation level, to solution level. In algorithmic-level parallelism, the optimization algorithm composes of different metaheuristics that can be executed in parallel. In generation-level parallelism, each generation

of a metaheuristic is parallelized by evaluating individual solutions in the population in parallel. In solution-level parallelism, each solution is decomposed and evaluated in parallel.

Based on the classification by Talbi, our method proposes a decomposition at the problem level, which decomposes at both algorithm level and solution level. Then, we introduce new coordination mechanisms for synchronizing interactions among the subproblems, including synchronous or asynchronous coordination among the subproblems. At the iteration level, we also divide the population and evaluate the subpopulations in parallel.

7. Conclusion

We proposed a parallel simulation-based optimization approach called SICPO to solve large-scale traffic signal optimization by introducing new coordination mechanisms to improve convergence and reduce synchronization overheads. A traffic simulator is used to generate the interactions for the spatial decomposition of the problem into subproblems. This decomposition reduces the dimensionality of the problem and enables faster convergence in a smaller number of generations. The optimization time is greatly reduced by applying parallelism at two levels: (1) each subproblem is optimized concurrently, and (2) each solution in the subproblem is evaluated in parallel. The coordination among the subproblems is achieved in two ways: (1) coordinating at the end of the optimization process in every iteration through simulating the whole scenario, and (2) coordinating at generation level during the partial function optimization by synchronizing the interactions between the regions.

We demonstrated the applicability of our method on a city-scale traffic signal optimization problem based on a real-world road network of Singapore. Our performance evaluation shows that in a given time or CPU budget, SICPO converges faster than GO and is thus more applicable to city-scale traffic networks. In addition, we showed that generation-level coordination (i.e., SICPO-A and SICPO-S) can improve convergence at the cost of higher computational time. We found that asynchronous generation-level coordination (SICPO-A) has a lower computing budget compared with synchronous generation-level coordination (SICPO-S) at the cost of slightly worse convergence.

The simulation-based evaluation of the solutions is the most time-consuming portion of the optimization process. In future work, workload imbalance in the evaluations can be addressed by using a better partitioning strategy to return a more balanced partition in terms of the simulation workload, or a better scheduling strategy to execute the

imbalanced tasks. Budget allocation can further reduce the number of evaluations in the optimization process.⁴¹


Other than that, the proposed approach only generates a static traffic signal timings targetting a specific time window. Temporal partitioning will be investigated to handle the dynamic changes in the traffic flows throughout the day.


The proposed method may be applied to other spatial optimization problems, as long as the interactions between the regions exhibit some degree of spatial locality. A potential example is the urban network design problem, which consists of optimizing the layout of an urban road network by deciding the driving directions of existing roads and the signal settings at intersections.⁴² However, determining a suitable decomposition of the problem remains a problem-dependent challenge.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was financially supported by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) program and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) (grant no. 497901036).

ORCID iDs

Wen Jun Tan  <https://orcid.org/0000-0003-2204-0639>

Philipp Andelfinger  <https://orcid.org/0000-0002-0211-7136>

References

1. Zhao F, Sun H, Wu J, et al. Analysis of road network pattern considering population distribution and central business district. *PLoS One* 2016; 11: e0151676.
2. Brockfeld E, Barlovic R, Schadschneider A, et al. Optimizing traffic lights in a cellular automaton model for city traffic. *Phys Rev E* 2001; 64: 056132.
3. Kamrani M, Hashemi Esmail Abadi SM and Rahimpour Golroudbary S. Traffic simulation of two adjacent unsignalized T-junctions during rush hours using Arena software. *Simul Model Pract Th* 2014; 49: 167–179.
4. Land Transport Authority (LTA), Singapore. Length of roads maintained by LTA (Singapore), 2018, <https://data.gov.sg/dataset/length-of-road-maintained-by-lta>
5. Land Transport Authority (LTA), Singapore. Number of traffic lights (Singapore), 2018, <https://data.gov.sg/dataset/traffic-lights>
6. Teklu F, Sumalee A and Watling D. A genetic algorithm approach for optimizing traffic control signals considering routing. *Comput-Aided Civ Inf* 2007; 22: 31–43.
7. Ceylan H and Bell MGH. Traffic signal timing optimisation based on genetic algorithm approach, including drivers' routing. *Transport Res B: Meth* 2004; 38: 329–342.

8. Sanchez JJ, Galan M and Rubio E. Genetic algorithms and cellular automata: a new architecture for traffic light cycles optimization. In: *Proceedings of the 2004 congress on evolutionary computation (IEEE cat. no. 04th8753)*, Portland, OR, 19–23 June 2004, vol. 2, pp. 1668–1674. New York: IEEE.
9. Garcia-Nieto J, Ferrer J and Alba E. Optimising traffic lights with metaheuristics: reduction of car emissions and consumption. In: *Proceedings of the 2014 international joint conference on neural networks (IJCNN)*, Beijing, China, 6–11 July 2014, pp. 48–54. New York: IEEE.
10. Gökçe MA, Öner E and Işık G. Traffic signal optimization with particle swarm optimization for signalized roundabouts. *Simulation* 2015; 91: 456–466.
11. Vlahogianni EI. Optimization of traffic forecasting: intelligent surrogate modeling. *Transport Res C: Emer* 2015; 55: 14–23.
12. Andelfinger P, Udayakumar S and Cai W. Model preemption based on dynamic analysis of simulation data to accelerate traffic light timing optimisation. In: *Proceedings of the 2018 winter simulation conference (WSC'18)*, Gothenburg, 9–12 December 2018, pp. 652–663. New York: IEEE.
13. Ezzat AA, Farouk HA, El-Kilany KS, et al. Optimization using simulation of traffic light signal timings. In: *Proceedings of the 2014 international conference on industrial engineering and operations management*, IEOM Society International, Bali, Indonesia, 7–9 January 2014, p. 11.
14. Talbi EG. A unified view of parallel multi-objective evolutionary algorithms. *J Parallel Distr Com* 2019; 133: 349–358.
15. Jalili S, Nallaperuma S, Keedwell E, et al. Application of metaheuristics for signal optimisation in transportation networks: a comprehensive survey. *Swarm Evol Comput* 2021; 63: 100865.
16. Van den Bergh F and Engelbrecht AP. A cooperative approach to particle swarm optimization. *IEEE T Evolut Comput* 2004; 8: 225–239.
17. Papadimitriou CH and Tsitsiklis JN. The complexity of optimal queuing network control. *Math Oper Res* 1999; 24: 293–305.
18. Xu Y, Cai W, Eckhoff D, et al. A graph partitioning algorithm for parallel agent-based road traffic simulation. In: *Proceedings of the 2017 ACM SIGSIM conference on principles of advanced discrete simulation*, Singapore, 24–26 May 2017, pp. 209–219. New York: ACM Press.
19. Law AM and McComas MG. Simulation-based optimization. In: *Proceedings of the winter simulation conference*, San Diego, CA, 8–11 December 2002, vol. 1, pp. 41–44. New York: IEEE.
20. Ma X, Li X, Zhang Q, et al. A survey on cooperative co-evolutionary algorithms. *IEEE T Evolut Comput* 2019; 23: 421–441.
21. Chandra R, Frean M and Zhang M. A memetic framework for cooperative coevolution of recurrent neural networks. In: *Proceedings of the 2011 international joint conference on neural networks*, San Jose, CA, 31 July–5 August 2011, pp. 673–680. New York: IEEE.
22. Nguyen MH, Abbass HA and McKay RI. Analysis of CCME: coevolutionary dynamics, automatic problem decomposition, and regularization. *IEEE T Syst Man Cy C* 2008; 38: 100–109.
23. Gong YJ, Chen WN, Zhan ZH, et al. Distributed evolutionary algorithms and their models: a survey of the state-of-the-art. *Appl Soft Comput* 2015; 34: 286–300.
24. Rinaldi M, Himpe W and Tampère CMJ. A sensitivity-based approach for adaptive decomposition of anticipatory network traffic control. *Transport Res C: Emer* 2016; 66: 150–175.
25. Viswanathan V, Zehe D, Ivanchev J, et al. Simulation-assisted exploration of charging infrastructure requirements for electric vehicles in urban environments. *J Comput Sci* 2016; 12: 1–10.
26. Federal Highway Administration (FHWA). *Manual on uniform traffic control devices (MUTCD)*. FHWA, https://mutcd.fhwa.dot.gov/pdfs/2009r1r2/pdf_index.htm
27. Mafecki K, Pietruszka P and Iwan S. Comparative analysis of selected algorithms in the process of optimization of traffic lights. In: Nguyen NT, Tojo S, Nguyen LM, et al. (eds) *Intelligent information and database systems (Lecture notes in computer science)*. Cham: Springer International Publishing, 2017, pp. 497–506.
28. Karypis G and Kumar V. Multilevel k -way partitioning scheme for irregular graphs. *J Parallel Distr Com* 1998; 48: 96–129.
29. Cheong CC and Toh R. *Household interview surveys from 1997 to 2008 — a decade of changing travel behaviours*. Singapore: LTA Academy, Land Transport Authority, 2010, pp. 52–61.
30. Hassanat A, Almohammadi K, Alkafaween E, et al. Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach. *Information* 2019; 10: 390.
31. Roupail NM, Park BB and Sacks J. Direct signal timing optimization: strategy development and results (technical report). In: *Proceedings of the XI Pan American conference in traffic and transportation engineering*, Gramado, Brazil, January 2000.
32. Sánchez-Medina JJ, Galan-Moreno MJ and Rubio-Royo E. Traffic signal optimization in La Almozara district in Saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing. *IEEE T Intell Transp* 2010; 11: 132–141.
33. García-Nieto J, Alba E and Olivera AC. Enhancing the urban road traffic with Swarm Intelligence: a case study of Córdoba city downtown. In: *Proceedings of the 2011 11th international conference on intelligent systems design and applications*, Cordoba, 22–24 November 2011, pp. 368–373. New York: IEEE.
34. Gao K, Zhang Y, Su R, et al. Solving traffic signal scheduling problems in heterogeneous traffic network by using meta-heuristics. *IEEE T Intell Transp* 2019; 20: 3272–3282.
35. Timotheou S, Panayiotou CG and Polycarpou MM. Distributed traffic signal control using the cell transmission model via the alternating direction method of multipliers. *IEEE T Intell Transp* 2015; 16: 919–933.
36. Mehrbipour M and Hajbabaie A. A cell-based distributed-coordinated approach for network-level signal timing optimization. *Comput-Aided Civ Inf* 2017; 32: 599–616.
37. Adacher L and Tiriolo M. A distributed approach for traffic signal synchronization problem. In: *Proceedings of the 2016 3rd international conference on mathematics and computers in sciences and in industry (MCSI)*, Chania, 27–29 August 2016, pp. 191–196. New York: IEEE.

38. Adacher L. A global optimization approach to solve the traffic signal synchronization problem. *Procd Soc Behv* 2012; 54: 1270–1277.
39. Villagra A, Alba E and Luque G. A better understanding on traffic light scheduling: new cellular GAs and new in-depth analysis of solutions. *J Comput Sci* 2020; 41: 101085.
40. Liang Y, Ren Z, Wang L, et al. Surrogate-assisted cooperative signal optimization for large-scale traffic networks. *Knowl-Based Syst* 2021; 234: 107542.
41. Al-Salem M, Almomani M, Alrefaei M, et al. On the optimal computing budget allocation problem for large scale simulation optimization. *Simul Model Pract Th* 2017; 71: 149–159.
42. Gallo M, D’Acerno L and Montella B. A meta-heuristic approach for solving the urban network design problem. *Eur J Oper Res* 2010; 201: 144–157.

Author biographies

Wen Jun Tan is a postdoctoral research fellow at Nanyang Technological University (NTU), Singapore. He received his Ph.D. in Computer Science in 2020 from NTU. His research interests include cyber-physical systems, large-scale parallel and distributed simulations for supply chain, traffic and manufacturing systems.

Philipp Andelfinger is a postdoctoral researcher at the Institute for Visual and Analytic Computing of the University of Rostock. He received his PhD in Computer Science from the Karlsruhe Institute of Technology in 2016.

Wentong Cai is a professor in the School of Computer Science and Engineering at NTU. He received his PhD in Computer Science from University of Exeter (UK) in

1991. His expertise is mainly in the areas of Modeling and Simulation and Parallel and Distributed Computing. He has published extensively in these areas and has received a number of the best paper awards at the international conferences for his research in parallel and distributed simulation. He is an associate editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS) and an editor of the Future Generation Computer Systems (FGCS).

David Eckhoff is a principal scientist and the director of the MoVES (Mobility in Virtual Environments at Scale) laboratory at TUMCREATE, a research institute by Technical University of Munich in Singapore. He received his PhD degree in engineering and his MSc degree in computer science from the University of Erlangen in 2016 and 2009, respectively. His research interests include privacy protection, smart cities, vehicular networks, and intelligent transportation systems with a particular focus on modeling and simulation.

Alois Knoll is a professor of Computer Science at the Department of Informatics of the Technical University of Munich. He received the Dipl.-Ing. (MSc) degree in Electrical/Communications Engineering from the University of Stuttgart, Germany, in 1985 and his PhD in Computer Science from the Technical University of Berlin, Germany, in 1988. His research interests include cognitive, medical, and sensor-based robotics, multi-agent systems, data fusion, adaptive systems, multimedia information retrieval, model-driven development of embedded systems with applications to automotive software and electric transportation, as well as simulation systems for robotics and traffic.