Improved Quantum-Inspired Evolutionary Algorithm for Large-Size Lane Reservation

Ada Che, Peng Wu, Feng Chu, Mengchu Zhou

Improved Quantum-Inspired Evolutionary Algorithm for Large-Size Lane Reservation

Ada Che, Peng Wu, Feng Chu, and MengChu Zhou

***Abstract*—This paper studies a lane reservation problem for large sport events in big cities. Such events require organizers to deliver certain people and materials from athlete villages to geographically dispersed venues within a given travel duration. A lane reservation strategy is usually adopted in this circumstance to ensure that time-critical transportation tasks can be completed despite heavy urban traffic congestion. However, it causes negative impact on normal traffic. The problem aims to optimally select and reserve some lanes in a transportation network for the exclusive use of the tasks such that the total traffic impact is minimized. To solve the problem, we first develop an improved integer linear program. Then, its properties are analyzed and used to reduce the search space for its optimal solutions. Finally, we develop a fast and effective quantum-inspired evolutionary algorithm for large-size problems. Computational results on instances with up to 500 nodes in the network and 50 tasks show that the proposed algorithm is efficient in yielding high-quality solutions within a relatively short time.**

***Index Terms*—Integer linear program (ILP), lane reservation, large-size problem, optimization, transportation planning.**

I. Introduction

With the rapid development of economies, high urbanization has become a reality in many countries. However, heavy traffic congestion arises due to the rapid increases of vehicles. Traffic situations in many large cities have been worse than ever before. Transportation planning and management play a critical role in the sustainable development of economies and the improvement to human's daily travel environment. Many problems, such as vehicle routing [1], location routing [2], and facility location [3], have drawn much attention over the past few decades. People are facing more and more new and special transport needs. To meet these needs in saturated urban transportation networks and in an increasingly congested traffic situation, it is necessary to develop novel and efficient technologies and methods for transportation planning and management. The lane reservation problem (LRP) is such a new transportation planning problem to deal with special transport needs that have arisen from large sport events, emergencies, and so on.

Large sport events require organizers to transport certain people and materials from athlete villages to geographically dispersed stadiums within given deadlines. For example, the organizers of the Guangzhou Asian Games in 2010 were committed to deliver athletes to any stadium within 30 min [4]. However, it is not so easy to meet such transport needs due to the host city's congested traffic situation. A lane reservation strategy in an existing transportation network may solve this problem in a flexible and efficient way. With this strategy, a lane on some road segments is temporarily reserved for these special transportation tasks such that they can be completed within the given travel duration. In fact, the lane reservation strategy has been successfully applied to some large sport events. For example, the strategy of reserving lanes for buses was adopted during the Sydney Olympic Games in 2000 [5]. Zagorianakos [6] analyzed the importance and advantage of the lane reservation strategy for the Athens Olympic Games in 2004. It was also successfully applied to the Beijing Olympic Games in 2008 where Olympic exclusive lanes were created and implemented for Olympic buses to deliver athletes to stadiums in a fast, safe, and reliable way. In addition, the lane reservation strategy was applied to evacuating people during large-scale emergencies [7].

However, reserving lanes in an existing transportation network has negative impact on normal traffic since only the vehicles for special deliveries can use the reserved lanes while the general-purpose vehicles are not allowed. This may further worsen the host city's already congested traffic. Hence, how to reserve lanes is critical to minimizing the impact caused by lane reservation. Such a transportation planning problem is called as an LRP. The problem lies in how to reserve lanes in an existing transportation network for some special tasks such that they can be completed within their given travel duration while the total traffic impact of reserved lanes is minimized.

This work was supported in part by the National Natural Science Foundation of China under Grants 71071129 and 71471145, in part by the National Science Foundation under Grant CMMI-1162482, in part by the Cai Yuanpei Program between the French Ministries of Foreign and European Affairs and the Higher Education and Research and the Chinese Ministry of Education under Grant 27927VE, in part by the Program of 100 Foreign Experts in Anhui Province and the Program of Chair Professor of Huangshan Scholars at the Hefei University of Technology, and in part by the Humanities, Social Sciences and Management Innovation Foundation of Northwestern Polytechnical University under Grant RW201301. This paper was recommended by Associate Editor M. P. Fanti. *(Corresponding author: Peng Wu.)*

A. Che is with the School of Management, Northwestern Polytechnical University, Xi'an 710072, China (email: ache@nwpu.edu.cn).

P. Wu is with the School of Management, Northwestern Polytechnical University, Xi'an 710072, China, and also with the Laboratory IBISC, University of Évry-Val d'Essonne, Évry 91020, France (e-mail: peng.wu@ibisc.univ-evry.fr).

F. Chu is with the Laboratory IBISC, University of Évry-Val d'Essonne, Évry 91020, France, and also with the School of Transportation Engineering, Hefei University of Technology, Hefei 230009, China (email: feng.chu@ibisc.univ-evry.fr).

M. Zhou is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA and also with the Institute of Systems Engineering, Macau University of Science and Technology, Macau 999078, China (e-mail: zhou@njit.edu).

There are only limited studies on the LRP in the literature. Wu *et al.* [4] first developed an integer linear program (ILP) and solved small-size problems via a simple heuristic algorithm. Their heuristic can only solve the problems with up to 22 nodes in the network and 22 tasks. Fang *et al.* [8] investigated a capacitated LRP where each lane in the network was supposed to have limited residual capacity for the special transportation tasks. If the residual capacity of a lane is not large enough for the special transportation tasks, then this lane should be reserved. Later, Fang *et al.* [9] studied an LRP for automated truck freight transportation, and designed entirely reserved paths for time-guaranteed and safety-critical freight transportation. Recently, Fang *et al.* [10] studied the LRP with dynamic link travel time, and divided the whole time period into several time intervals. The link travel times in different intervals are different. In the above three studies, the authors proposed exact cut-and-solve-based methods to solve their problems. In addition, Zhou *et al.* [11] examined a multiobjective hazardous material transportation problem via a lane reservation strategy and proposed an ε-constraint and fuzzy logic-based method. Exact methods like cut-and-solve algorithms [8]–[10] can obtain the optimal solutions while their solution time is found to exponentially increase with the size of problems. The results on randomly generated instances with up to 120 nodes in the network and 30 tasks were reported. Unfortunately, they usually fail to solve large-size problems within a reasonable time. Therefore, efficient heuristics or metaheuristics are required to solve large-size problems, which is the focus of this paper.

To solve various large-size transportation planning problems, heuristics or metaheuristics are proposed, such as Lagrangian relaxation-based algorithm [12]–[14], genetic algorithm [15], [16], tabu search [17], [18], and scatter search [19]. Quantum-inspired evolutionary algorithm (QEA) [20] is an algorithm based on the concept and principles of quantum computing. It was reported that QEA outperformed genetic algorithms for knapsack problems [20], [21]. Due to its excellent performance, it was used to solve many NP-hard problems, such as machine scheduling problem [22], traveling salesman problem [23], and vehicle routing problem [24]. Its successful applications have motivated us to apply it to the LRP.

The work in this paper is an extension and completion of our prior work [25]. In [25], we proposed a QEA to solve the LRP. Its main features include: 1) each arc in a network corresponds to a gene in the encoded chromosome to represent a solution for the LRP; 2) probability amplitudes are initialized with the same value to generate an initial quantum population; 3) a simple reparation strategy is developed; and 4) computational results on a limited number of instances (75 instances in total) with up to 150 nodes in the network and 30 tasks are reported. Note that crossover, mutation, and catastrophe operators as well as a dereserving operation are not described in detail in [25]. This paper differs from [25] as follows.

1) It develops an improved mathematical model for the LRP and proposes a new method to assess the traffic impact of a reserved lane.
2) We perform property analysis for the model and derive some analytical properties that can be used to reduce the search space for its optimal solutions.
3) Based on the derived properties we develop an improved QEA (IQEA). In IQEA, we only need to consider a partial set of arcs in the network for the solution representation according to the derived properties.
4) We propose a new method based on the lower and upper bounds on the number of reserved lanes to initialize quantum population.
5) Crossover, mutation, and catastrophe operators as well as a dereserving strategy are explained in detail.
6) To repair infeasible solutions, we develop two procedures. The first one is to directly repair the solutions based on the derived lower and upper bounds on the number of reserved lanes. The second one is similar to that in [25], but used only after the first one fails. Their combination may lead to a rapid and efficient reparation of infeasible solutions.
7) Computational results on 485 randomly generated instances show that the proposed algorithm is able to yield high-quality solutions for large-size problems with up to 500 nodes in the network and 50 tasks within a relatively short time. Therefore, IQEA significantly improves QEA in [25].

The remainder of this paper is organized as follows. Section II describes the problem and formulates an improved ILP. Its analytical properties are analyzed in Section III. Section IV presents an IQEA. Section V reports its computational and comparison results. Section VI concludes this paper and discusses future work.

II. Problem Formulation

Let a directed graph $G = (N, A)$ be a connected transportation network, where N is the set of nodes and A is the set of directed arcs that connect the nodes. Arcs and nodes can be viewed as road segments and road intersections in the transportation network, respectively. Multiple special transportation tasks need to be performed from origins $O \subseteq N$ to their corresponding destinations $D \subseteq N$, and each task corresponds to an origin–destination (OD) pair.

For the investigated LRP, we need to convert some lanes into reserved lanes for exclusive use of the special transportation tasks, and design a time-guaranteed path for each task so as to ensure that it can be completed within its given travel duration. By doing so, vehicles of these tasks can travel at a relatively high speed on the reserved lanes and their travel time on these reserved lanes is reduced. General-purpose vehicles are not allowed to use the reserved lanes. Thus, they use the non-reserved lanes that may be more congested. Hence, reserved lanes cause negative impact on normal traffic. The objective is to minimize the traffic impact caused by reserved lanes.

To well formulate the problem, the following assumptions are made. First, there are at least two lanes on each road segment and the lanes on the same road segment are assumed to be identical. Second, at most one lane on any road segment is allowed to be reserved for the special tasks, and each

TABLE I
NOTATIONS AND VARIABLES

Notations	
N:	set of nodes in the network
A:	set of arcs in the network
B_i:	set of arcs outgoing from node $i \in N$, $B_i \subseteq A$
F_i:	set of arcs entering node $i \in N$, $F_i \subseteq A$
K:	set of tasks, $k \in K$
O:	set of origin nodes, $o_k \in O \subseteq N$
D:	set of destination nodes, $d_k \in D \subseteq N$
T_k:	prescribed travel duration for task k, $k \in K$
T_a:	travel time on arc a if no lane of arc a is reserved, $a \in A$
T'_a:	travel time on a reserved lane of arc a, $a \in A$.
M_a:	number of lanes on arc a, $a \in A$
C_a:	traffic impact if a lane is reserved on arc a, $a \in A$
Decision variables	
z_a:	z_a =1, if there is a reserved lane on arc $a \in A$; otherwise z_a =0
x_{ak}:	x_{ak}=1, if there is a reserved lane on arc $a \in A$, and the path of task k passes the arc; otherwise x_{ak} =0
y_{ak}:	y_{ak}=1, if there is no reserved lane on arc $a \in A$, and the path of task k passes the arc; otherwise y_{ak} =0

reserved lane can be shared by all special tasks. Third, the path of each task can be composed of reserved and nonreserved lanes. That is to say, the task paths can be partially reserved. Finally, the travel time on a lane can be decreased if it is reserved. Some notations are given in Table I. Note that the definitions of the decision variables are similar to those in [4] and [8].

The investigated LRP is formulated as the following ILP:

$$P_0: \ \min \sum_{a \in A} C_a z_a \tag{1}$$

$$\text{s.t.} \sum_{a \in B_i} (x_{ak} + y_{ak}) = 1, i = o_k, \forall k \in K \tag{2}$$

$$\sum_{a \in F_i} (x_{ak} + y_{ak}) = 1, i = d_k, \forall k \in K \tag{3}$$

$$\sum_{a \in F_i} (x_{ak} + y_{ak}) = 0, i = o_k, \forall k \in K \tag{4}$$

$$\sum_{a \in B_i} (x_{ak} + y_{ak}) = 0, i = d_k, \forall k \in K \tag{5}$$

$$\sum_{a \in F_i} (x_{ak} + y_{ak}) = \sum_{a \in B_i} (x_{ak} + y_{ak}) \quad \forall i \in N/\{o_k, d_k\}, \forall k \in K \tag{6}$$

$$\sum_{a \in F_i} (x_{ak} + y_{ak}) \le 1, \forall i \in N/\{o_k\}, \forall k \in K \tag{7}$$

$$\sum_{a \in B_i} (x_{ak} + y_{ak}) \le 1, \forall i \in N/\{d_k\}, \forall k \in K \tag{8}$$

$$\sum_{a \in A} (T_a y_{ak} + T'_a x_{ak}) \le T_k, \forall k \in K \tag{9}$$

$$x_{ak} \le z_a, \forall a \in A, \forall k \in K \tag{10}$$

$$y_{ak} \le 1 - z_a, \forall a \in A, \forall k \in K \tag{11}$$

$$x_{ak}, y_{ak}, z_a \in \{0, 1\}, \forall a \in A, \forall k \in K. \tag{12}$$

Objective function (1) is to minimize the total negative impact caused by reserved lanes. Constraints (2)–(8) guarantee that a feasible path from origin o_k to destination d_k, $k \in K$, exists. To be more specific, (2) and (3) ensure that there is one and only one arc outgoing from o_k and one arc coming into d_k on the travel path of task k, respectively. Constraints (4) and (5) guarantee that there are no arcs coming into o_k and no arcs outgoing from d_k on the travel path of task k, respectively. Constraint (6) ensures the flow balance. Constraints (7) and (8) mean that each node in the network is visited by task k, $k \in K$, at most once. Constraint (9) ensures that the total travel time of task k, $k \in K$, does not exceed its given travel duration T_k. Constraint (10) guarantees that the path of task k cannot pass a reserved lane on arc a if no lane of arc a is reserved. Constraint (11) guarantees that the path of task k cannot pass a general-purpose lane on arc a if this arc is reserved. Constraint (12) is binary constraint on the decision variables.

The following theorem gives the complexity of the LRP.

Theorem 1: The LRP is an NP-hard problem.

Proof: If the given travel duration of each task is so large that (9) can be relaxed and removed from the model, the reduced LRP corresponds to the Steiner tree problem in a directed graph. The latter one is known to be an NP-hard problem [26]. Hence, the LRP is also NP-hard. ■

Note that, we improve the models [4], [8] after relaxing the residual capacity constraint from the following aspects. First, (4) and (5) are added in our model to prevent from generating a loop that starts from and comes back to origin o_k and destination d_k, respectively. Second, (7) and (8) are added to the model to provide a tighter formulation. Third, we denote an arc by a single-tuple a, instead of using a two-tuple (i, j), where i and j are the head and tail nodes of the arc, respectively. Such representation simplifies the formulation. Computational results in Section V show that the improved model can save 79% and 35% average computational time compared with the models in [4] and [8] after the relaxation of the residual capacity constraint, respectively.

In addition, we propose a new method to assess the traffic impact of reserved lanes. As previously described, the goal of the LRP is to minimize the traffic impact of reserved lanes. Therefore, a reasonable assessment of the traffic impact of reserved lanes is critical to selecting which lanes to be reserved. Up to now, researchers have not come to a consensus on how to evaluate their traffic impact. According to [8], it is not so easy to quantitatively assess it. Wu *et al.* [4] pioneered in an assessment formula, which is followed by [8]–[11] and [25], and is shown as follows:

$$C_a = T_a/(M_a - 1) \tag{13}$$

where C_a, T_a, and M_a are the traffic impact of a reserved lane on arc a, the travel time on arc a if no lanes are reserved, and the number of lanes on arc a, respectively, $a \in A$. Equation (13) shows that the longer the travel time before reservation of a lane on arc a, the greater the traffic impact. The more the lanes, the less the traffic impact. Below, we give an illustration to show the limitation of (13).

Fig. 1 describes two arcs a and b in a transportation network. Their lengths are 0.1 and 0.5 km, respectively. There are two lanes on each arc, and the travel time of vehicles on arcs a and b before reservation of a lane are both 1 min. Assume that one of the lanes on both arcs is reserved. Thus, the traffic impact of the two reserved lanes is exactly the same

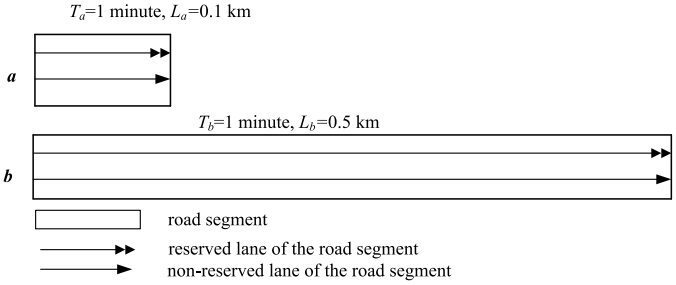

Fig. 1. Two road segments in a transportation network.

according to (13). We now demonstrate that this does not match what happens in reality. Note that the average travel speeds on arcs a and b before reservation of the lanes are 6 and 30 km/h, respectively. It is not difficult to find that vehicles travel much slowly on arc a than on arc b before reserving a lane. The generalized exponential speed-concentration relationship model [27] can be formulated as

$$V = \alpha e^{-\mu q^{\beta}} \tag{14}$$

where V denotes the average travel speed (km/h), q represents the concentration (vehicles/km), and μ, α, and β are user-specified parameters. By (14), we can find that the smaller the speed, the higher the concentration, which implies that the corresponding road segment is more congested. Therefore, a is much more congested than b before reservation of the lane. It is understandable that reserving a lane on a would have a much greater traffic impact than that on b. Hence, reserving one of the two lanes in two scenarios in Fig. 1 has the same impact is in contradiction with the above analysis. Hence, formula (13) is not appropriate in measuring the impact of reserved lanes under some traffic situations.

A reserved lane on a road segment reduces the number of available lanes for general-purpose vehicles, and consequently, causes negative traffic impact. The traffic impact is related to the road segment condition before reservation of lanes such as traffic flow, type of road segment, and capacity. Note that the average travel speed of vehicles on a road segment before reservation of a lane is a consequence of all the above factors. For this reason, the average travel speed of vehicles on a road segment before reservation of a lane is used to estimate such an impact, which is expressed by

$$C_a = R_a / V_a \tag{15}$$

where V_a denotes the average travel speed on arc a before reservation of a lane and R_a denotes the impact coefficient. By (15), we can see the impact is in inverse proportion to V_a. The greater the travel speed on arc a before reservation of a lane, the smaller the impact of the reserved lane. With (15), reserving a lane on a has a greater traffic impact than that on b.

Note that in this paper, the traffic impact caused by a reserved lane on each segment is separately estimated, as is the case in the previous studies [4], [8]–[11]. It might not be able to accurately reflect the actual traffic impact of a reserved lane due to its complexity. This paper only makes an attempt to give a relatively more reasonable one than (13), which was the only one used in the previous studies. This issue demands more studies in the future. Besides, the impact parameters of reserved lanes can be viewed as input data and their sensitive analysis is conducted in this paper.

III. Analysis of LRP Properties

In this section, we first conduct new preprocessing to reduce the search space for optimal solutions. Then, we derive lower and upper bounds on the number of reserved lanes for an LRP by solving two linear programs to further reduce it.

A. Preprocessing

Define $l(i, j)$ as the shortest travel time from node i to $j, i, j \in N$, when all the arcs in a transportation network have a reserved lane, which can be easily obtained by the Floyd–Warshall algorithm with its complexity $O(|N|^3)$. A set A_k is defined as follows:

$$A_k = \left\{ a \middle| \begin{array}{l} l(o_k, \text{head}(a)) + T'_a \\ \quad + l(\text{tail}(a), d_k) > T_k, a \in A \end{array} \right\}, \forall k \in K \tag{16}$$

where head(a) and tail(a) denote the head and tail nodes of arc a, respectively. According to (16), it is not difficult to see that any reserved arc a in A_k would not be used by the vehicles of task k due to the violation of its given travel duration. Consequently, the corresponding decision variables can be set as zero. We thus have

$$x_{ak} + y_{ak} = 0, a \in A_k, k \in K. \tag{17}$$

We note that, Fang *et al.* [8] also performed a similar preprocessing. However, they checked only the arcs outgoing from origin nodes and the arcs coming into destination nodes whereas we preprocess all the arcs. Hence, this paper is a generalization of the work in [8].

According to (16), we have:

$$\Omega = \left\{ a | a \in A_1 \cap A_2 \cdots \cap A_{|K|} \right\}. \tag{18}$$

According to (18), any reserved arc a in set Ω would not be passed by any task, i.e., $x_{ak} = 0, \forall k \in K$. Hence, it is not required to reserve any lanes on arc a. We thus have

$$z_a = 0, a \in \Omega. \tag{19}$$

With (19), the arcs in set Ω do not need to be reserved. The search space is thus reduced.

B. Lower Bound on the Number of Reserved Lanes

We obtain a lower bound on the number of reserved lanes by solving a linear relaxation problem of an ILP (i.e., all the integer variables in it are relaxed to real variables), denoted by LP_1. It is formulated as follows:

$$\text{LP}_1 : \min \sum_{a \in A} z_a$$
$$\text{s.t. Constraints (2)–(11), (17), (19)}$$

and

$$0 \le x_{ak} \le 1, \forall a \in A, \forall k \in K \tag{20}$$
$$0 \le y_{ak} \le 1, \forall a \in A, \forall k \in K \tag{21}$$
$$0 \le z_a \le 1, \forall a \in A. \tag{22}$$

The optimal value of LP_1 provides a lower bound on the number of reserved lanes, denoted by L^*. By definition, we have $L^* = \lceil l^* \rceil$, where l^* is the optimal value of problem LP_1. Note that x denotes the smallest integer greater than or equal to x.

C. Upper Bound on the Number of Reserved Lanes

As previously defined, $l(o_k, d_k)$ represents the shortest travel time from origin o_k to destination $d_k, k \in K$, when each arc in a transportation network has a reserved lane. The following theorem is thus straightforward.

Theorem 2: The optimal solution of model P_0 must satisfy the following relation:

$$\sum_{a \in A} \left(T_a y_{ak} + T'_a x_{ak} \right) \geq l(o_k, d_k). \tag{23}$$

We derive an upper bound on the optimal number of reserved lanes by solving the following linear relaxation problem (LP_2).

$$\begin{aligned} &LP_2 : \max \sum_{a \in A} z_a \\ &\text{s.t. Constraints (2)–(11), (17), (19)–(23)} \end{aligned}$$

and

$$z_a \leq \sum_{k \in K} (x_{ak} + y_{ak}), \forall a \in A. \tag{24}$$

Constraint (24) means that one of the lanes on arc a is required to be reserved if at least one task's path passes the arc.

Note that, LP_2 is a linear relaxation problem. The optimal value of LP_2 provides an upper bound of its corresponding integer program. Let U^* denote an upper bound on the number of reserved lanes. We obtain $U^* = \lfloor u^* \rfloor$, where u^* is the optimal value of problem LP_2. Note that $\lfloor x \rfloor$ is the largest integer smaller than or equal to x.

IV. Proposed Approach

In this section, an IQEA is proposed to find near-optimal solutions for large-size LRP. To improve its performance, some techniques are developed. First, preprocessing is conducted to reduce the search space. Second, a new quantum population initialization method is proposed based on the lower and upper bounds on the number of reserved lanes. Third, a dereserving procedure and two reparation procedures, called direct and greedy ones, are developed to repair the solutions generated in the search process.

A. Solution Representation

A Q-bit representation is adopted in QEA instead of binary, numeric, and symbolic representations. The smallest unit of information stored in a two-state quantum computer is called a Q-bit whose state is expressed as $|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers that specify the probability amplitudes of their corresponding states with $|\alpha|^2 + |\beta|^2 = 1$ [20]. $|\alpha|^2$ and $|\beta|^2$ give the probabilities that the state of a Q-bit will be found in the "0" and "1", respectively. A Q-bit may be in 0 and 1, or in any linear superposition of both states.

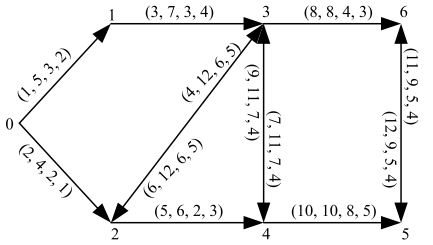

Fig. 2. Network of an LRP.

A Q-bit chromosome can be defined by a string of n Q-bits as follows:

$$\left[\begin{array}{c|c|c|c|c|c|c} \alpha_1 & \alpha_2 & \ldots & \alpha_i & \ldots & \alpha_{n-1} & \alpha_n \\ \beta_1 & \beta_2 & \ldots & \beta_i & \ldots & \beta_{n-1} & \beta_n \end{array} \right]$$

where $0 \leq \alpha_i \leq 1$, $0 \leq \beta_i \leq 1$, $|\alpha_i|^2 + |\beta_i|^2 = 1$, and $i = 1, 2, \ldots, n$. In this way, a Q-bit individual with n Q-bits can represent 2^n states at the same time. Q-bit representation has better population diversity than any other classical representation, since it can represent a linear superposition of solutions. The Q-bits in a Q-bit chromosome approach 1 or 0 once they are observed. Thus, a Q-bit chromosome collapses to a single state (i.e., a binary string) if it is observed. Readers can refer to [20] for more details.

Now, we show how to convert a Q-bit chromosome with n Q-bits into a binary string. For Q-bit i, if $|\alpha_i|^2 >$ random [0, 1], its state is observed as 0 and the gene of its corresponding binary string takes the value of 0; otherwise, its state is 1 and the gene of its corresponding binary string takes the value of 1. With a binary string, we can represent if each arc is reserved or not. To be more specific, each arc in a transportation network corresponds to a gene of a binary string. If a gene takes the value of 1, then there is a reserved lane on the corresponding arc; and otherwise, none.

Our prior work [25] directly adopted such a representation. It is obvious that the number of genes in a chromosome is equal to the total number of arcs in a transportation network (i.e., $|A|$). For a large-size problem with many arcs, the chromosome can be very long. Through the preprocessing presented in Section III, we know that the arcs in set Ω would not be reserved. Thus, we need consider only those arcs that do not belong to set Ω. Consequently, the number of genes in a chromosome is equal to the number of arcs in the network that do not belong to set Ω, i.e., $|A| - |\Omega|$. With the proposed preprocessing, the length of chromosome can be reduced as shown in the following example.

Example 1: Consider an LRP with only one task in a network with 7 nodes and 12 arcs, as shown in Fig. 2. Each arc is characterized by a four-tuple (Arc#, T, T', C), where Arc# represents the index of the arc, T is the travel time if no lanes are reserved on the arc, T' is the travel time if a lane is reserved on the arc, and C is the traffic impact generated by a reserved lane. For instance, (1, 5, 3, 2) represents that the travel time before and after reserving a lane on arc 1 is 5 and 3, respectively, and the impact due to a reserved lane on this arc is 2. Nodes 0 and 6 are the origin and destination

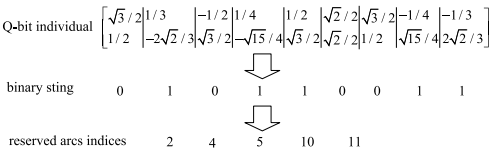

Fig. 3. Decoding of a Q-bit individual for Example 1.

nodes of the transportation task to be completed, respectively. Its given travel duration is 19.

With the coding way in [25], the chromosome length is 12 due to 12 arcs. The proposed preprocessing finds that $\Omega = \{6, 7, 12\}$. Thus, the chromosome length for this instance is reduced from 12 to 9. This reduces the search space of the proposed algorithm. Fig. 3 shows an example of decoding of a Q-bit individual. In Fig. 3, suppose that a binary string is [0, 1, 0, 1, 1, 0, 0, 1, 1], which is observed from a Q-bit chromosome. Thus, the lane reservation solution is [2, 4, 5, 10, 11], where the numbers represent the indices of the reserved arcs.

Given a lane reservation solution, the travel path for task $k, k \in K$, can be obtained by calculating the shortest travel time from its origin o_k to destination d_k. If the shortest travel time of any task is less than or equal to its given travel duration, then the solution is said to be feasible. Otherwise, it is infeasible.

B. Quantum Population Initialization

Let $L(p)$ denote the number of genes taking the value of 1 in an observed binary string p. The following theorem is straightforward.

Theorem 3: If $L(p) < L^*$, the observed binary string p must be an infeasible one, and if $L(p) > U^*, p$ must not be an optimal one.

The probability amplitudes (α, β) are usually initialized with the same probability of $1/\sqrt{2}$ [20]–[22]. Our prior work [25] applied this way to initialize quantum population. In this paper, we propose a new quantum population initialization method that initializes $Q(t)(t = 0)$ based on L^* and U^*. A Q-bit population $Q(t)$ can be expressed as: $Q(t) = \{q_1^t, q_2^t, \ldots, q_i^t, \ldots, q_{\mathrm{Ps}}^t\}$, where Ps is the population size and q_i^t is the ith Q-bit individual at the tth generation, $i = 1, 2, \ldots,$ Ps, represented by

$$q_i^t = \left[\begin{array}{c|c|c|c|c|c} \alpha_{i1}^t & \alpha_{i2}^t & \ldots & \alpha_{ij}^t & \ldots & \alpha_{in}^t \\ \beta_{i1}^t & \beta_{i2}^t & \ldots & \beta_{ij}^t & \ldots & \beta_{in}^t \end{array}\right].$$

For each Q-bit chromosome q_i^0, we randomly generate an integer number m from the interval $[L^*, U^*]$, and then generate m Q-bits satisfying $|a_{ij}^0|^2 < |\beta_{ij}^0|^2$, and the remaining $n-m$ Q-bits satisfying $|a_{ij}^0|^2 > |\beta_{ij}^0|^2$. Thus, when the Q-bits of the generated Q-bit individual are observed, the number of genes with the value of 1 in the binary string falls in the interval $[L^*, U^*]$ with a large probability. Fig. 4 compares the traditional and proposed ways via Example 1.

Suppose that $L^* = 1$ and $U^* = 3$, and $m = 3$, which is randomly selected from {1, 2, 3} for a Q-bit individual by the proposed method. Suppose that individuals q' and q in Fig. 4 are generated by the traditional method and the new one, respectively. When q' is observed, the number of genes

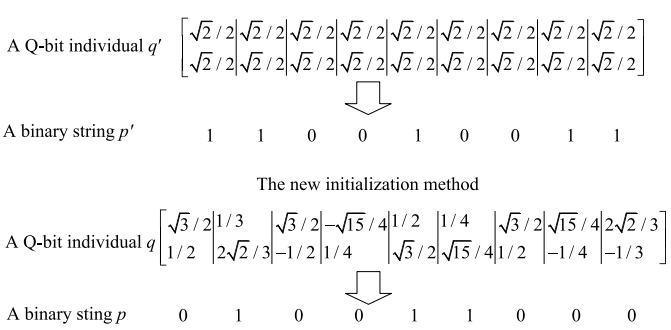

Fig. 4. Comparison of the two initialization methods.

TABLE II
LOOKUP TABLE OF ROTATION ANGLE [28]

p_i	b_i	$f(p) \geq f(b)$	$\Delta\theta_i$	$s(\alpha_i, \beta_i)$			
				$\alpha_i\beta_i>0$	$\alpha_i\beta_i<0$	$\alpha_i=0$	$\beta_i=0$
0	0	*false*	0	0	0	0	0
0	0	*true*	0	0	0	0	0
0	1	*false*	0	0	0	0	0
0	1	*true*	0.05π	−1	+1	±1	0
1	0	*false*	0.01π	−1	+1	±1	0
1	0	*true*	0.025π	+1	−1	0	±1
1	1	*false*	0.005π	+1	−1	0	±1
1	1	*true*	0.025π	+1	−1	0	±1

whose values are 1 in its observed binary string equals to 4 or 5 with a large probability. When q is observed, the number of genes whose values are 1 in its observed binary string falls in {1, 2, 3} with a large probability. Suppose that $p' = [1, 1, 0, 0, 1, 0, 0, 1, 1]$ and $p = [0, 1, 0, 0, 1, 1, 0, 0, 0]$, which are observed from q' and q, respectively. Obviously, $L(p') = 5$ and $L(p) = 3$. According to Theorem 3, p' must not be an optimal one, whereas p is likely to be so.

C. Quantum Rotation Gate

In this paper, the state transformation of Q-bits is executed by a quantum rotation gate, as done in [20]–[22] and [25]. The rotation gate is applied to update Q-bit individuals in IQEA in order to maintain the diversity of population, which is an important and widely used updating method.

Let $G(\theta)$ denote a quantum rotation gate. The ith Q-bit (α_i, β_i) is updated as follows:

$$\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = G(\theta_i)\begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix}\begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$$

where θ_i is a rotation angle. $\theta_i = s(\alpha_i, \beta_i)\Delta\theta_i$, where $s(\alpha_i, \beta_i)$ is its sign that determines the direction of rotation, and $\Delta\theta_i$ is its magnitude represented by a lookup table, as shown in Table II [28].

The quantum rotation gate focuses on a wise guidance by the current best individual in population at the current generation. In Table II, b_i and p_i are the ith binary bit of the best and current individuals at the current generation, respectively. $f(.)$ computes the fitness value. With the lookup table, the current individual can converge to a better solution. For example, if $p_i = 0$, $b_i = 1$, and $f(p) > f(b)$, then the current solution p_i

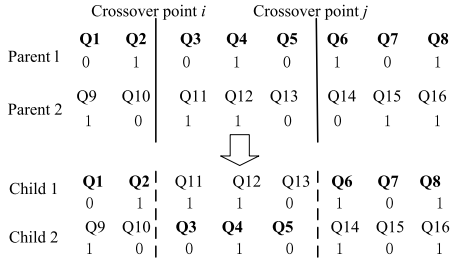

Fig. 5. Hybrid two-point crossover.

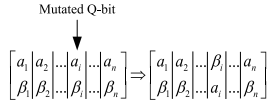

Fig. 6. Quantum mutation.

should have larger possibility to take the value of 0, i.e., $|\alpha_i|^2$ should be larger. Hence, if (α_i, β_i) in the first or third quadrant (i.e., $\alpha_i\beta_i > 0$), θ_i should rotate clockwise, and otherwise should rotate counter-clockwise.

D. Genetic Operators

1) Crossover: To improve the search efficiency, we propose a hybrid two-point crossover imposed on Q-bit chromosomes and their corresponding binary strings simultaneously. Fig. 5 illustrates this operator. Two paternal Q-bit chromosomes and their corresponding binary strings are selected from the population according to crossover probability Pc. For simplicity, let Q1–Q16 denote the Q-bits. Then, two positions are randomly generated (e.g., positions i and j). The genes of the paternal Q-bit chromosomes and their binary strings before position i and after position j are both reserved while the genes of the Q-bit chromosomes and their binary strings between position i and j are exchanged. The proposed hybrid crossover operator improves the diversity of the Q-bit chromosomes and their binary strings simultaneously.

2) Mutation: Mutation operator can effectively improve the local search ability and prevent premature convergence [21]. Therefore, a quantum mutation is employed in our IQEA. Its process can be described as follows. First, select paternal Q-bit individuals to be mutated according to probability Pm_1. Then, for each paternal Q-bit individual, determine whether its Q-bits are mutated according to probability Pm_2. For each Q-bit to be mutated, its probability amplitudes are exchanged. Fig. 6 shows such a quantum mutation, where we illustrate how Q-bit i is mutated.

3) Catastrophe: To avoid premature convergence, if the best solution does not change in the prescribed consecutive Sg iterations, the evolution search is considered to be trapped in the local optima. Then, we reserve the current best solution and the population is initialized again.

4) Selection: Selection is used to maintain the size of population, Ps. Roulette wheel selection is used in order to ensure some poor individuals carrying good genes have chance to be selected in the next generation. Besides, we also take the retention strategy of the global elite individual to prevent the best individual from losing in the evolution procedure.

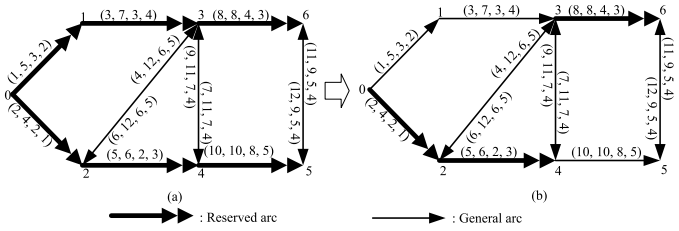

Fig. 7. Example for the dereserving strategy. (a) Before and (b) after dereserving strategies are implemented.

E. Dereserving and Reparation Strategies

1) Dereserving Strategy: As previously described, a Q-bit individual generated in a quantum evolution process is decoded to a lane reservation solution, i.e., which lanes in a network are reserved. The decision variables connected with travel paths are fixed by calculating the shortest path for each task. It is not difficult to see that some reserved lanes may be not included on the path of any task. That is to say, for a lane reservation solution generated in the above way, some of its reserved lanes may not contribute to the transportation tasks. These reserved but nonpassed lanes increase unnecessary traffic impact and worsen the fitness value of its corresponding binary solution. We develop a dereserving strategy to convert these reserved but nonpassed lanes into nonreserved ones.

To achieve this purpose, we specify all the lanes that are passed by the transportation tasks in a given binary solution. This can be accomplished by solving a shortest path problem for each task. Then, we determine all reserved lanes that are not passed by any task in the binary solution and dereserve them by converting the corresponding genes in the solution from 1 to 0. As these reserved but nonpassed lanes are not used by any task, such a procedure would not violate the feasibility of a solution.

We use a simple example to illustrate the proposed strategy. Assume that $p = [1, 1, 1, 0, 1, 1, 0, 1, 0]$ is a binary solution of the problem described in Example 1, whose genes correspond to arcs 1–5 and 8–11, respectively. This means that there is a reserved lane on arcs 1–3, 5, 8, and 10, as shown in Fig. 7(a). It can be observed that the shortest path from nodes 0 to 6 is $0 \to 2 \to 4 \to 3 \to 6$, which consists of arcs 2, 5, 9, and 8, and the total travel time of the task is 19. This implies that this solution is feasible. We see that the reserved arcs 1, 3, and 10 are not passed by the task. The dereserving operation is to reconvert them into general-purpose ones. After the operation, $p^* = [0, 1, 0, 0, 1, 1, 0, 0, 0]$, as shown in Fig. 7(b). Note that p^* is still feasible after conducting the dereserving procedure.

2) Reparation Procedures: As analyzed before, for a lane reservation solution generated in a quantum evolution process, if the calculated shortest travel duration for any task in the reserved network exceed its given travel duration, the obtained solution is infeasible due to violation of constraint (9). The generation of infeasible solutions may greatly influence the search efficiency of the algorithm. Here, we develop two methods to repair infeasible solutions generated in the evolution process according to probability Pr. They are called direct reparation and greedy algorithm-based reparation (greedy reparation in short), respectively.

1: For an infeasible solution p= [x_1, x_2, .., x_n], calculate $L(p)$.
2: **If** ($L(p)$<L^*) **do**
a. Generate IL = *rand* (L^*– $L(p)$, U^*– $L(p)$)
b. Randomly select IL genes with the value of 0 and convert their values into 1.
If ($L(p)$>U^*) **do**
a. Generate RL = *rand* ($L(p) - U^*$, $L(p) - L^*$)
b. Randomly select RL genes with the value of 1 and convert their values into 0.
where *rand* (x, y) denotes an integer randomly generated from [x, y].

Fig. 8. Flowchart of direct reparation.

1: Select an infeasible solution p= [x_1, x_2, .., x_n].
2: Obtain the reserved network according to solution p.
3:Initialize k = 1
while ($k \le |K|$) do
3.1 Find the shortest path from origin o_k to destination d_k in the obtained reserved network, and calculate the shortest travel duration $\tau(o_k, d_k)$.
3.2. **while** ($\tau(o_k, d_k) > T_k$) do
a. Calculate the reduced travel time per unit impact (Rcp) for all non-reserved arcs passed by task k.
b. Let arc e = arg max Rcp_a
c. Convert arc e into a reserved arc.
d. Compute $\tau(o_k, d_k)$, if $\tau(o_k, d_k) \le T_k$, break.
end while
3.3. $k = k+1$.
end while

Fig. 9. Flowchart of greedy reparation.

a) Direct reparation: By Theorem 3, we conduct the direct reparation for binary solution p satisfying $L(p) < L^*$ or $L(p) > U^*$, as shown in Fig. 8.

b) Greedy reparation: Direct reparation can repair some infeasible solutions. We propose greedy reparation to repair the rest. Infeasible solutions are generated due to violation of constraint (9). That is to say, there is at least one task that cannot be completed within the given duration T_k, i.e., the shortest travel time from origin o_k to destination d_k exceeds T_k. We call such a task as an unsatisfied task in the following discussion. The principal idea is described as follows. We first determine all unsatisfied tasks. Then, we repair them one by one by converting some nonreserved lanes on its shortest path into reserved ones. We introduce a greedy algorithm to select a nonreserved lane and convert it into reserved one.

Assume that arc a is a nonreserved arc on the shortest path of an unsatisfied task. Let $\text{Rt}_a = T_a - T'_a$ and $\text{Rcp}_a = \text{Rt}_a / C_a$ be the reduced travel time after this arc is reserved and the reduced travel time per unit impact, respectively. We choose the nonreserved arc with the largest Rcp and convert it into a reserved lane. Let $\tau(o_k, d_k), k \in K$, denote the shortest travel duration from origin o_k to destination d_k. The greedy reparation is shown in Fig. 9.

We use Example 1 to illustrate our procedure. For a binary solution $p = [0, 1, 0, 0, 1, 0, 0, 0, 0]$, its shortest path for the task is $0 \to 2 \to 4 \to 3 \to 6$. The shortest path for the task consists of arcs 2, 5, 9, and 8. Its shortest travel time is 23, which is greater than the given travel duration 19, as shown in Fig. 10(a). Therefore, this solution is infeasible. For this example, arcs 9 and 8 are two nonreserved arcs on its shortest path. Their ratios of reduced travel time/impact are 1 and 4/3, respectively. Thus, arc 8 is chosen and converted

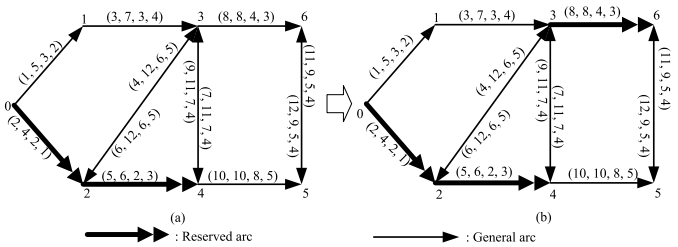

Fig. 10. Example of the greedy reparation. (a) Before and (b) after reparations.

into a reserved one, as shown in Fig. 10(b). After this conversion, the travel time of the task is 19. This means that the repaired solution [0, 1, 0, 0, 1, 1, 0, 0, 0] becomes feasible.

F. Fitness Computation

Infeasible solutions may exist in the evolution process. A fitness function combined with a penalty strategy for infeasible solutions is designed. As previously mentioned, infeasible solutions are generated due to violation of constraint (9). The penalty strategy is based on the concept of violation degree. We define the violation coefficient of task k, denoted by φ_k, as follows:

$$\varphi_k = \begin{cases} 0, & \tau(o_k, d_k) \le T_k \\ (\tau(o_k, d_k) - T_k)/T_k, & \tau(o_k, d_k) > T_k \end{cases}$$

where $\tau(o_k, d_k)$ denotes the shortest travel duration of task k, $k \in K$, in the reserved network for a given solution. φ_k reflects the gap between $\tau(o_k, d_k)$ and the given travel duration T_k, which measures the violation degree of task k regarding constraint (9). The average violation coefficient of all the tasks, denoted by φ, is defined as $\varphi_k/|K|$. Note that if a solution is feasible, $\varphi = 0$.

The fitness value f is computed as follows:

$$f = 1 \Big/ \left(1 + \sum_{a \in A \setminus \Omega_a} C_a z_a + \varphi w\right) \tag{25}$$

where $\sum_{a \in A \setminus \Omega_a} C_a z_a$ represents the total traffic impact, φ represents the penalty value, and w is a suitable positive number.

G. Termination Criterion

The algorithm stops when the number of iterations reaches the maximum iteration count (Maxg) or the number of catastrophes reaches the maximum value (Mct).

H. Framework of IQEA

We outline the framework of IQEA for the investigated LRP as follows.

Step 1: Implement the preprocessing and obtain the values of L^* and U^* by solving LP_1 and LP_2.

Step 2: Initialize the values of parameters Ps, Pc, Pm_1, Pm_2, Pr, Sg, Maxg, and Mct. Set iteration count $t_1 = 0$ and catastrophe count $t_2 = 0$.

Step 3: Initialize quantum population $Q(t_1)$ and observe $Q(t_1)$ to obtain the binary population $P(t_1)$, compute the fitness for each individual, and save the best individual $\boldsymbol{b}$.

Step 4: If the termination criterion is met, output the best LRP solution and stop.

Step 5: Perform crossover for both $Q(t_1)$ and $P(t_1)$ and compute the fitness for each child individual.

Step 6: Perform dereserving and reparation operations.

Step 7: Perform mutation and selection to generate $Q'(t_1)$ and $P'(t_1)$.

Step 8: If the catastrophe condition is met, perform catastrophe operation to $Q'(t_1)$ and let $t_2 = t_2 + 1$.

Step 9: Update $Q'(t_1)$ to $Q(t_1 + 1)$ by the quantum rotation gate.

Step 10: Observe $Q(t_1 + 1)$ to obtain the binary representation $P(t_1 + 1)$, and compute the fitness for each individual. Update the best solution $\boldsymbol{b}$ if current solution is better than the old one. Let $t_1 = t_1 + 1$ and go back to Step 4.

V. Computational Results

In this section, the performance of the proposed algorithm is evaluated on 15 problem sets in [25] and 82 new randomly generated problem sets with five instances for each set, thus leading to 485 instances in total. The algorithm is coded in C++ with Visual Studio 2008 embedded with optimization software CPLEX (version 12.4). We compare our algorithm with CPLEX IP solver in default setting in terms of solution quality and computational time (CPU seconds). All experiments are performed on a PC with 2.5 GHz processor and 2.95 GB RAM under the Windows 7 system.

The instances are generated in a similar way to that in [4] and [8]. The network is generated based on Waxman's method [29]. The nodes are randomly generated from 100×100 square. The existence probability of arc a between two nodes is determined by a probability function $\alpha \exp(-L_a/\beta L)$, where $0 < \alpha, \beta \leq 1$, L_a and L are the Euclidean distance of arc a and the maximum Euclidean distance between any pair of nodes in the network, respectively. The OD pairs of the tasks are uniformly and randomly generated from the nodes in the network. T_a is defined as L_a/V_a, where V_a is the average travel speed on arc a. T'_a is then defined as $b_a{*}T_a$. In order to make an instance feasible, T_k is randomly generated between the shortest travel time in a network with every arc being reserved and that in an entirely nonreserved path from origin o_k to destination d_k. As previously presented, C_a is defined as R_a/V_a in this paper. In the default setting, the parameters V_a, b_a, and R_a are randomly generated from the intervals [10, 80], [0.5, 0.8], and [11, 20], respectively.

Note that, IQEA is a class of iterative heuristic algorithms. Its parameters are set according to preliminary numerical tests. Via these tests, a relatively small population size, Ps, is set as 50 due to the good diversity of quantum population as previously described. Parameter Pc is set as a relatively large value of 0.7 to further improve the diversity of the population and make our algorithm search for the global best solution faster. Mutation is used to prevent premature convergence, but large Pm_1 or Pm_2 usually leads to random mutation. Thus, we set them both as 0.2. Larger Pr means that more infeasible solutions can be repaired in the evolution process, but costs more computational efforts. For this reason, Pr is set as 0.2. Finally, parameters Sg, Mct, and Maxg are set as 10, 100, and 500, respectively.

TABLE III
Performance of the Proposed Model

Set	$\|N\|$	$\|K\|$	T_C	T_W	T_F	T_C/T_W	T_C/T_F
1	50	20	3.61	6.11	4.14	0.59	0.87
2	60	20	11.5	22.21	15.85	0.52	0.73
3	70	20	19.11	52.68	35.69	0.36	0.54
4	80	20	33.64	158.28	56.79	0.21	0.59
5	90	20	56.94	351.75	79.91	0.16	0.71
Average			24.96	118.2	38.48	0.21	0.65

Let $|N|$ and $|K|$ denote the number of nodes in the network, and the number of tasks, respectively; U and Opt denote the average value of best upper bounds obtained by IQEA and the optimal objective value obtained by CPLEX IP solver if doable, respectively; and T and T_C denote the average value of CPU time spent by IQEA and CPLEX for solving the model P_0, respectively. The gap is computed as $(U - \mathrm{Opt}) \times 100/\mathrm{Opt}$, and if CPLEX cannot exactly solve an LRP, it is computed as $(U - U_C) \times 100/U_C$, where U_C is the best upper bound obtained by CPLEX within a given time. The gap reflects the solution quality of IQEA. Note that each value in Tables IV–X is its average value of the five instances. For each instance, we run both IQEA and QEA for five times and calculate their average values.

To validate the contributions of this paper, we conduct the experiments by comparing: 1) the proposed model P_0 with previously reported ones; 2) the proposed method with the one in [25]; and 3) the proposed method with CPLEX on large-size instances. Sensitive analysis for input parameters is performed as well.

A. Model Comparison

In order to evaluate the proposed model, we compare it with the previous models in the literature by solving a small number of instances. The proposed model, Wu-model [4], and Fang-model [8] after relaxing residual capacity constraint for these instances are all directly solved by CPLEX. Let T_W and T_F denote the time spent by Wu-model and Fang-model, respectively. The comparison results are reported in Table III. It can be seen from Table III that the computational time of model P_0 is less than that by Wu-model and Fang-model on all the sets, and the average time of our model represents only 21% and 65% that of the two existing models, respectively. This indicates that model P_0 outperforms them.

B. Comparison With the Work [25]

Our prior work [25] proposed a QEA to find near-optimal solutions for LRP. It can solve problems with up to 150 nodes in the network and 30 tasks. In this paper, we improve it. In order to evaluate IQEA, we first compare it with QEA in [25]

TABLE IV
COMPUTATIONAL RESULTS FOR THE INSTANCES WITH $|K| = 20$, 25, AND 30, $|N| = 110$–150

Set	$\|N\|$	$\|K\|$	QEA[25]		IQEA in this work		T/T'
			Gap(%)	T'(s)	*Gap* (%)	T(s)	
6	110		1.25	139.42	1.16	48.07	0.34
7	120		3.08	237.49	1.75	70.93	0.30
8	130	20	1.41	239.39	0.89	71.40	0.30
9	140		3.23	291.37	1.92	92.05	0.32
10	150		0.92	337.84	0.62	104.23	0.31
11	110		1.97	214.61	1.89	59.24	0.28
12	120		2.95	237.73	1.99	66.73	0.28
13	130	25	1.76	256.60	1.57	90.66	0.35
14	140		1.89	286.15	1.12	86.95	0.30
15	150		3.34	307.83	1.89	105.29	0.34
16	110		2.51	216.66	1.28	78.48	0.36
17	120		0.86	241.84	0.68	89.98	0.37
18	130	30	2.81	307.69	2.01	102.87	0.33
19	140		1.59	367.95	1.56	118.12	0.32
20	150		1.85	469.12	1.46	147.19	0.31
Average			2.09	276.78	1.45	88.81	0.32

by using the instances tested in [25]. The comparison is performed in terms of the gap and the computational time. Let T' denotes the time of QEA. In order to ensure the fairness of comparison, all the parameters of the two algorithms are set as the same.

Table IV reports the results for the instances with $|K| = 20$, 25, and 30 and $|N|$ increasing from 110 to 150. It can be seen from Table IV that the gap of QEA varies from 0.86% to 3.34% and the average value is 2.09%, whereas the gap of IQEA varies from 0.62% to 2.01% and the average value is 1.45%. This indicates that both QEA and IQEA can obtain high-quality solutions. Besides, IQEA has a smaller gap than QEA on all the sets. This demonstrates that IQEA is more effective than QEA in terms of solution quality.

In terms of computational time, we can observe that T' varies from 139.42 s to 469.59 s, while T varies from 48.07 s to 147.19 s. IQEA spends a much smaller amount of time than QEA for all the sets. On average, IQEA spends only 32% time of that spent by QEA. In addition, as $|N|$ increases for a fixed $|K|$, T' increases more quickly than T, especially for $|K| = 30$. Moreover, for the same $|N|$, T and T' both slightly increase with $|K|$, but T increases more slowly. This indicates IQEA is more efficient than QEA in terms of computional time. Especially, for the largest-size problem set 20, QEA finds its best solution with the gap of 1.85% within 469.12 s, whereas IQEA find its best solution with the gap of 1.46% within 147.19 s, which represents 31% of the time spent by QEA.

In summary, IQEA can obtain the solutions of higher-quality in smaller CPU time than QEA in [25].

C. Large-Size Instances

To further evaluate the performance of IQEA in terms of the solution quality and computational efficiency, we compare it with the well-known solver CPLEX using 46 new randomly generated problem sets under different scenarios. Choosing the commercial software CPLEX as a reference is because it is a well-known and strong IP solver for integer programming problems [30]. Note that the computational time of IQEA and CPLEX for each instance is limited to 18 000 s. The computational results are reported in Tables V–VII and Figs. 11–13.

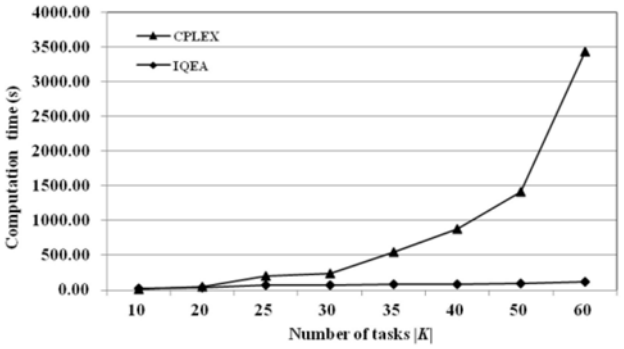

Fig. 11. Computational time for the instances with 100 fixed nodes.

TABLE V
COMPUTATIONAL RESULTS FOR THE INSTANCES WITH $|N| = 100$

Set	$\|K\|$	*Gap*(%)	T	T_C	T/T_C
21	10	0.90	21.72	7.58	2.87
22	20	1.87	34.06	48.04	0.71
23	25	2.29	73.28	207.03	0.35
24	30	2.23	78.58	237.94	0.33
25	35	2.25	83.88	548.74	0.15
26	40	2.66	97.85	885.55	0.11
27	50	2.42	104.32	1413.11	0.07
28	60	2.49	117.31	3438.43	0.03
Average		2.14	76.38	848.30	0.09

First, we randomly generate and test instances with a fixed number of nodes $|N| = 100$, while the number of tasks $|K|$ increases from 10 to 60. The results are summarized in Table V. We can observe that the gap varies from 0.9% to 2.66%, with the average gap being 2.14%. This indicates that IQEA provides high-quality feasible solutions that are very near to the optimal ones. The computational time of IQEA varies from 21.72 s to 117.31 s with its average value being 76.38 s, whereas the time spent by CPLEX varies from 7.58 s to 3438.43 s with its average value being 848.3 s. By comparing T with T_C, we can observe that IQEA significantly outperforms CPLEX in terms of computational time. On average, our algorithm spends only 9% time of CPLEX for all the sets with the average gap of 2.14%. From Table V and Fig. 11, we can see that the time of CPLEX rapidly increases with the number of tasks $|K|$, whereas that of IQEA gradually increases. Moreover, the ratio T/T_C decreases with the number of tasks $|K|$. This shows the advantage of IQEA in terms of computational time compared with CPLEX for a fixed $|N|$ and varying $|K|$. Take set 28 as an example. It takes only 117.31 s for IQEA to find a near-optimal solution

TABLE VI
COMPUTATIONAL RESULTS FOR THE INSTANCES WITH $|K| = 20$, 30, AND 40, $|N| = 160$–200

Set	$\|N\|$	$\|K\|$	*Gap*(%)	T	T_C	T/T_C(%)
29	160		3.24	92.60	334.88	27.65
30	170		2.43	87.38	385.54	22.66
31	180	20	3.44	99.95	462.39	21.62
32	190		4.86	146.10	773.62	18.89
33	200		4.52	168.58	1515.47	11.12
34	160		4.50	94.30	491.85	19.17
35	170		7.56	120.47	3375.22	3.57
36	180	30	6.36	139.99	4695.39	2.98
37	190		5.34	196.39	6650.25	2.95
38	200		6.86	220.45	8570.59	2.57
39	160		5.46	132.34	6054.21	2.19
40	170		5.93	140.76	8325.38	1.69
41	180	40	5.65	159.22	9565.31	1.66
42	190		7.06	246.35	11338.20	2.17
43	200		7.81	254.40	14401.20	1.77
Average			5.40	153.29	5129.30	2.99

with an acceptable average gap of 2.49%, whereas it takes more than 3400 s for CPLEX to find the optimal one.

Table VI presents the computational results for the instances with $|K|$ and $|N|$ varying from 20 to 40 and from 160 to 200, respectively. We can see from Table VI that the gap varies between 2.43% and 7.81% with the average gap being 5.4%. This indicates that our algorithm can find high-quality solutions. In addition, it can be seen that the gap slightly increase with $|N|$ and $|K|$. In general, the gap varies from 2.43% to 7.81%. This shows that the solution quality of the proposed algorithm is relatively stable.

On the other hand, we can observe from Table VI that for all the sets, T_C varies from 334.88 s to 14401.2 s with the average value being 5129.3 s, whereas T varies from 87.38 s to 254.4 s. T is less than T_C on all the sets, and on average T represents only 2.99% of that spent by CPLEX. Moreover, T_C sharply increases with $|N|$ for a fixed $|K|$, whereas T slightly increases. Fig. 12 illustrates the computational time of CPLEX and IQEA for the instances with $|K| = 20$. We can see from Fig. 12 that T increases much more slowly than T_C. For a fixed $|N|$, T_C also rapidly increases with $|K|$, whereas T increases quite slowly. Moreover, the ratio T/T_C decreases with increasing $|N|$ and $|K|$. This implies that our algorithm is more efficient for large-size problems than small-size ones. The results demonstrate that IQEA significantly outperforms CPLEX in terms of computational time. It is worthwhile to point out that it takes 14401.2 s for CPLEX to find the optimal solution for set 43, whereas our algorithm spends only 254.4 s, i.e., 1.77% the time of CPLEX in finding high-quality solutions with the average gap of 7.81%.

In Table VII, we report the results for the larger-size instances with $|N|$ increasing from 250 to 500 and $|K|$ increasing from 30 to 50. From Table VII, we can see that CPLEX cannot exactly solve all the instances within 18 000 s. Even for the smallest set with $|N| = 250$ and $|K| = 30$, one of the five instances cannot be exactly solved. It can be seen that the average gap over sets 44–47 between the solution found by our algorithm and the best upper bound found by CPLEX is 7.34%. Since four of the five instances of set 44 are exactly

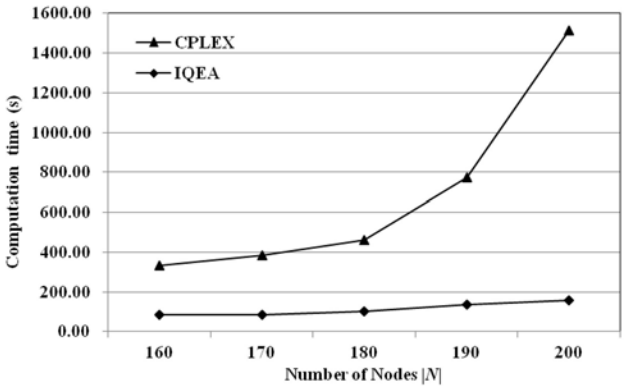

Fig. 12. Computational time for the instances with 20 fixed tasks.

TABLE VII
COMPUTATIONAL RESULTS FOR THE INSTANCES WITH $|N| = 250$–500 AND $|K| = 30$–50

Set	$\|N\|$	$\|K\|$	*Gap*(%)	T	T_C	T/T_C(%)
44	250	30	5.52	268.30	10927.17	2.46
45	300	30	8.76	361.48	18000.00	2.01
46	350	40	7.89	578.41	18000.00	3.21
47	400	40	7.20	960.56	18000.00	5.34
48	500	50	-	1443.88	18000.00	8.02
Average			7.34	722.53	16585.43	4.36

"–" represents out of memory

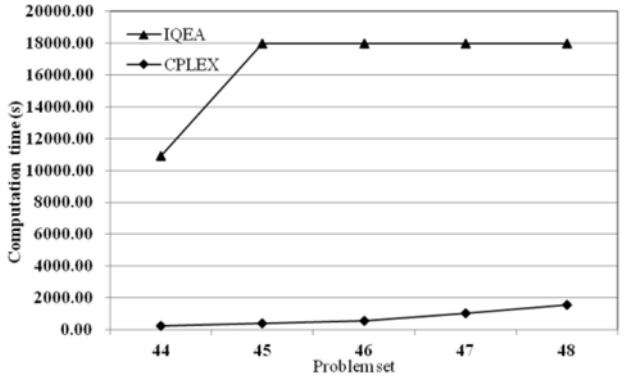

Fig. 13. Computational time for the instances with $|N| = 250$–500 and $|K| = 30$–50.

solved, set 44 has the smallest gap 5.52%. This implies that our algorithm provides high-quality solutions. The gap slightly decreases from set 45 to set 47. This may be because the quality of the upper bound provided by CPLEX declines with the increase of the problem size. As a result, the gap decreases from 8.76% to 7.2%.

It takes 722.53 s (4.36% that of CPLEX) on average for IQEA to obtain the solutions with the average gap of 7.34% from the optimal or best upper bound found by CPLEX. It can be found from Fig. 13 that T is far less than T_C. Especially, CPLEX runs out of memory for set 48 and no feasible solutions are obtained, whereas our algorithm finds them within 1443.88 s. This indicates that it is necessary to propose a metaheuristic method to obtain near-optimal solutions for large-size problems since exact methods fail to find even a feasible solution due to the NP-hard nature of LRP.

TABLE VIII
COMPUTATIONAL RESULTS FOR INSTANCES WITH DIFFERENT PARAMETERS OF TRAFFIC IMPACT $C_a = R_a/V_a$

Set	R_a	$\|N\|$	$\|K\|$	Gap(%)	T	T_C	T/T_C
49	[1, 10]	50	20	0.06	7.16	5.47	1.31
50	[1, 10]	60	20	0.15	8.12	9.89	0.82
51	[1, 10]	70	20	1.02	8.20	10.33	0.79
52	[1, 10]	80	20	2.13	11.13	29.44	0.38
53	[1, 10]	90	20	0.84	16.23	32.54	0.50
54	[1, 10]	100	20	1.58	30.89	48.36	0.64
55	[11, 20]	50	20	0.21	6.80	3.61	1.88
56	[11, 20]	60	20	0.59	8.11	11.50	0.71
57	[11, 20]	70	20	1.08	9.10	19.11	0.48
58	[11, 20]	80	20	2.04	11.25	33.64	0.33
59	[11, 20]	90	20	0.45	16.98	41.90	0.41
60	[11, 20]	100	20	1.87	30.22	48.04	0.63
61	[21, 30]	50	20	0.52	6.76	5.52	1.22
62	[21, 30]	60	20	0.53	7.55	13.67	0.55
63	[21, 30]	70	20	1.02	8.99	24.47	0.37
64	[21, 30]	80	20	1.88	10.84	36.55	0.30
65	[21, 30]	90	20	0.84	20.01	46.16	0.43
66	[21, 30]	100	20	2.26	33.46	75.21	0.44
Average				1.06	13.99	27.52	0.51

TABLE IX
COMPUTATIONAL RESULTS FOR INSTANCES WITH DIFFERENT PARAMETERS OF TRAVEL TIME $T_a = L_a/V_a$

Set	V_a	$\|N\|$	$\|K\|$	Gap(%)	T	T_C	T/T_C
67	[5, 10]	50	20	0.13	6.27	4.78	1.31
68	[5, 10]	60	20	0.17	7.16	10.12	0.71
69	[5, 10]	70	20	1.23	8.94	11.04	0.81
70	[5, 10]	80	20	1.12	10.33	25.86	0.40
71	[5, 10]	90	20	1.67	17.91	35.23	0.51
72	[5, 10]	100	20	2.11	33.70	52.31	0.64
73	[10, 80]	50	20	0.00	7.87	4.79	1.64
74	[10, 80]	60	20	0.21	9.25	12.37	0.75
75	[10, 80]	70	20	1.61	10.01	20.34	0.49
76	[10, 80]	80	20	1.92	11.02	34.78	0.32
77	[10, 80]	90	20	1.01	16.41	42.41	0.39
78	[10, 80]	100	20	1.22	32.31	46.32	0.70
79	[80, 120]	50	20	0.03	5.76	3.68	1.57
80	[80, 120]	60	20	0.39	8.57	9.24	0.93
81	[80, 120]	70	20	1.42	9.35	19.38	0.48
82	[80, 120]	80	20	0.96	13.47	34,20	0.39
83	[80, 120]	90	20	1.24	22.45	43.01	0.52
84	[80, 120]	100	20	1.48	30.12	51.28	0.59
Average				1.00	14.49	25.62	0.57

D. Sensitive Analysis

As previously analyzed, the traffic impact parameters of reserved lanes in a transportation network are critical to lane reservation decision. Therefore, in order to show that the performance of our algorithm is stable with regard to different traffic impact parameters, we test three scenarios for the traffic impact parameters. The traffic impact is defined as R_a/V_a. Parameter R_a is generated from the interval [11, 20], which is regarded as the baseline. In the other two cases, R_a is generated from intervals [1, 10] (smaller than the baseline) and [21, 30] (greater than the baseline), respectively, and the other parameters remains unchanged. The results are summarized in Table VIII.

In Table VIII, we can see that the gap varies from 0.06% to 2.26% with the average gap being 1.06%. This indicates that our algorithm can find very good solution, which is near the optimal solution. Moreover, the gap varies from 0.06% to 2.13% for the case of $R_a = [1,\ 10]$, from 0.21% to 2.04% for the case of $R_a = [11,\ 20]$, and from 0.52% to 2.26% for the case of $R_a = [21,\ 30]$, respectively. On the other hand, T is less than T_C for all the instances except sets 49, 55, and 61, which are the smallest size set in each scenario. The average time of our algorithm is 51% of that spent by CPLEX. Besides, the time of our algorithm ranges between 7.16 and 30.89 s, 6.8 and 30.22 s, and 6.76 and 33.46 s, respectively. In general, for all the scenarios, it remains almost the same. Similar results can also be found for T/T_C. This shows that the performance of IQEA is insensitive to changes in traffic impact parameters.

In addition, we also conduct sensitive analysis experiments for the parameters V_a and b_a, which are related with the travel time parameters on each road segment. We also test three scenarios for each of them. The computational results are reported in Tables IX and X, from which we can find similar results to those in Table VIII. This indicates that the performance of IQEA is also insensitive to changes of parameters T_a and T'_a.

TABLE X
COMPUTATIONAL RESULTS FOR INSTANCES WITH DIFFERENT TRAVEL TIME PARAMETERS $T'_a = b_a T_a$

Set	b_a	$\|N\|$	$\|K\|$	Gap(%)	T	T_C	T/T_C
85	[0.1, 0.5]	50	20	0.09	8.41	6.02	1.40
86	[0.1, 0.5]	60	20	0.12	9.01	10.06	0.90
87	[0.1, 0.5]	70	20	0.83	10.00	11.25	0.89
88	[0.1, 0.5]	80	20	1.37	11.20	25.74	0.44
89	[0.1, 0.5]	90	20	0.94	14.56	30.49	0.48
90	[0.1, 0.5]	100	20	1.67	28.32	43.02	0.66
91	[0.5, 0.8]	50	20	0.32	7.99	5.70	1.40
92	[0.5, 0.8]	60	20	1.05	9.03	10.80	0.84
93	[0.5, 0.8]	70	20	1.27	11.33	19.42	0.58
94	[0.5, 0.8]	80	20	1.93	12.08	36.29	0.33
95	[0.5, 0.8]	90	20	2.21	15.28	44.34	0.34
96	[0.5, 0.8]	100	20	1.94	31.23	70.26	0.44
97	[0.8, 0.95]	50	20	0.04	5.66	4.32	1.31
98	[0.8, 0.95]	60	20	0.42	6.78	10.20	0.66
99	[0.8, 0.95]	70	20	0.85	8.70	22.44	0.39
100	[0.8, 0.95]	80	20	1.27	11.20	32.18	0.35
101	[0.8, 0.95]	90	20	0.79	21.03	44.25	0.48
102	[0.8, 0.95]	100	20	1.77	29.04	50.35	0.58
Average				1.05	13.94	26.51	0.53

VI. CONCLUSION

This paper has investigated an NP-hard transportation planning problem called LRP. It makes four contributions: 1) an improved ILP formulation is developed for the LRP. Computational results on randomly generated instances have shown that the improved model runs faster than two previous models in the literature; 2) a new approach is proposed that can better assess the traffic impact of reserved lanes; 3) some analytical properties are obtained to reduce the search space for optimal solutions; and 4) a fast and efficient QEA is developed for large-size LRPs. Computational results on 485 randomly generated instances demonstrate that the proposed algorithm can yield high-quality solutions for large-size problems with up to 500 nodes in the network and 50 tasks within a relatively short time. The algorithm is promising for practical applications of lane reservation in real life.

Although IQEA proposed in this paper has achieved good performance for large-size LRPs, it still has some limitations. First, it is not difficult for IQEA to obtain a feasible solution for the LRP because usually its solution space is relatively big. Moreover, infeasible solutions in the evolution process can be repaired. However, when the solution space of an LRP is extremely small, the proposed algorithm may spend much time to find feasible solutions or even cannot find a feasible solution when it terminates. Consequently, construction methods based on the characteristic of LRP can be designed to generate initial feasible solutions for IQEA to avoid this issue in the future. Second, it can be seen from the computational results that both gap and computational time generally increase with the problem size. Based on our observation, we have found that the larger the problem size, the greater possibility for IQEA to trap into local optimum. Hence, effective local search techniques, such as tabu search [17], [18], [31], [32], can be introduced into IQEA to reduce the gap of the obtained solutions for larger-size instances within a shorter time. Third, IQEA is able to find feasible solutions for the instances with very large size (e.g., set 48), but its solutions cannot be evaluated because CPLEX is not able to find the optimal solution due to the NP-hardness of LRP. Therefore, effective relaxation method might be used to obtain good lower bounds to evaluate the obtained solutions.

In addition, in practical situations it is a quite challenging task to evaluate the traffic impact due to lane reservation because it is related with many factors, such as dynamic traffic flow, reserved time of a day and reservation status of other road segments. Although the assessment of traffic impact caused by a reserved lane has been improved compared with [4] and [8]–[11], its estimation deserves further studies and integrating the interaction among road segments into the traffic impact assessment is an important direction in the future. Other future directions include extending the problem addressed in this paper to other kinds of LRPs such as bus-LRP and introducing efficient search techniques [33]–[46] to solve them.

REFERENCES

[1] C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem," *Comput. Oper. Res.*, vol. 31, no. 12, pp. 1985–2002, Oct. 2004.

[2] Y. Yu, C. Chu, H. Chen, and F. Chu, "Large scale stochastic inventory routing problems with split delivery and service level constraints," *Ann. Oper. Res.*, vol. 197, no. 1, pp. 135–158, Aug. 2012.

[3] Z. Yang, F. Chu, and H. Chen, "A cut-and-solve based algorithm for the single source capacitated facility location problem," *Eur. J. Oper. Res.*, vol. 221, no. 3, pp. 521–532, Sep. 2012.

[4] Y. Wu, C. Chu, F. Chu, and N. Wu, "Heuristic for lane reservation problem in time constrained transportation," in *Proc. IEEE Int. Conf. Autom. Sci. Eng.*, Bangalore, India, Aug. 2009, pp. 543–548.

[5] J. Black, "Strategic transport planning, demand analysis of transport infrastructure and transport services for the 27th summer, Olympiad held in Sydney, Australia, 2000," *J. Transp. Syst. Eng. Inf.*, vol. 2, no. 2, pp. 14–30, 2004.

[6] E. Zagorianakos, "Athens 2004 Olympic games' transportation plan: A missed opportunity for strategic environmental assessment (SEA) integration," *J. Transp. Geogr.*, vol. 12, no. 2, pp. 115–125, Jun. 2004.

[7] T. Cova and J. Johnson, "A network flow model for lane-based evacuation routing," *Transp. Res. A Policy Pract.* vol. 37, no. 7, pp. 579–604, Aug. 2003.

[8] Y. Fang, F. Chu, S. Mammar, and M. C. Zhou, "Optimal lane reservation problem in transportation network," *IEEE Trans. Intell. Transp. Syst.,* vol. 13, no. 2, pp. 482–491, Jun. 2012.

[9] Y. Fang, F. Chu, S. Mammar, and A. Che, "An optimal algorithm for automated truck freight transportation via lane reservation strategy," *Transp. Res. C Emerg. Technol*., vol. 26, pp. 170–183, Jan. 2013.

[10] Y. Fang, F. Chu, S. Mammar, and A. Che, "A cut-and-solve based algorithm for optimal lane reservation with time-dependent travel time," *Int. J. Prod. Res.*, vol. 52, no. 4, pp. 1003–1015, 2014.

[11] Z. Zhou, F. Chu, A. Che, and M. C. Zhou, "ε-constraint and fuzzy logic-based optimization of hazardous material transportation via lane reservation," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 847–857, Jun. 2013.

[12] Y. Yu, H. Chen, and F. Chu, "A new model and hybrid approach for large scale inventory routing problem," *Eur. J. Oper. Res.*, vol. 189, no. 16, pp. 1022–1040, Sep. 2008.

[13] F. Yu, F. Tu, H. Tu, and K. Pattipati, "A Lagrangian relaxation algorithm for finding the MAP configuration in QMR-DT," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 5, pp. 746–757, Sep. 2007.

[14] S. Yan, C. Chen, and Y. Lin, "A model with a heuristic algorithm for solving the long-term many-to-many car pooling problem," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1362–1373, Jun. 2011.

[15] X. Hu and E. Di Paolo, "Binary-representation-based genetic algorithm for aircraft arrival sequencing and scheduling," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 2, pp. 301–310, Jun. 2008.

[16] L. Jiao and L. Wang, "A novel genetic algorithm based on immunity," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 30, no. 5, pp. 552–561, Sep. 2000.

[17] T. James, C. Rego, and F. Glover, "Multistart tabu search and diversification strategies for the quadratic assignment problem," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 3, pp. 579–596, May 2009.

[18] A. Schaerf, "Local search techniques for large high school timetabling problems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 29, no. 4, pp. 368–377, Jul. 1999.

[19] F. Chu, N. Labadi, and C. Prins, "A scatter search for the periodic capacitated arc routing problem," *Eur. J. Oper. Res.*, vol. 169, no. 2, pp. 586–605, Mar. 2006.

[20] K. Han and J. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," *IEEE Trans. Evol. Comput.*, vol. 6, no. 6, pp. 580–593, Dec. 2002.

[21] K. Han and J. Kim, "Quantum-inspired evolutionary algorithms with a new termination criterion, H-gate, and two-phase scheme," *IEEE Trans. Evol. Comput.,* vol. 8, no. 2, pp. 156–169, Apr. 2004.

[22] J. Gu, M. Gu, and C. Cao, "A novel competitive co-evolutionary quantum genetic algorithm for stochastic job shop scheduling problem," *Comput. Oper. Res.*, vol. 3, no. 5, pp. 927–937, May 2010.

[23] Y. Wang *et al.*, "A novel quantum swarm evolutionary algorithm and its applications," *Neurocomputing*, vol. 70, nos. 4–6, pp. 633–640, Jan. 2007.

[24] J. Zhang, Y. Zhao, D. Peng, and W. Wang, "A hybrid quantum-inspired evolutionary algorithm for capacitated vehicle routing problem," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*. Lecture Notes in Computer Science, vol. 5226. Berlin, Germany: Springer, 2008, pp. 31–38.

[25] P. Wu, A. Che, and F. Chu, "A quantum evolutionary algorithm for lane reservation problem," in *Proc. IEEE Int. Conf. Netw. Sens. Control*, Paris, France, Apr. 2013, pp. 264–268.

[26] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. New York, NY, USA: Plenum, 1972.

[27] J. Princeton and S. Cohen, "Impact of a dedicated lane on the capacity and the level of service of an urban motorway," *Procedia Soc. Behav. Sci.*, vol. 16, no. 1, pp. 196–206, 2011.

[28] L. Wang, F. Tang, and H. Wu, "Hybrid genetic algorithm based on quantum computing for numerical optimization and parameter estimation," *Appl. Math. Comput*., vol. 171, no. 2, pp. 1141–1156, Dec. 2005.

[29] B. M. Waxman, "Routing of multipoint connections," *IEEE J. Sel. Areas Commun.,* vol. 6, no. 9, pp. 1617–1622, Dec. 1988.

[30] *IBM ILOG: CPLEX 12.4*. [Online]. Available: http://www-01.ibm.com/software/integration/-optimization/cplex-optimizer, 2012

[31] M. Gendreau, A. Hertz, and G. Laporte, "A tabu search heuristic for the vehicle routing problem," *Manage. Sci.*, vol. 40, no. 10, pp. 1276–1290, Oct. 1994.

[32] L. Lee, K. Tan, K. Ou, and Y. Han, "Vehicle capacity planning system: A case study on vehicle routing problem with time windows," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 33, no. 2, pp. 169–178, Mar. 2003.
[33] L. Cao, R. Dai, and M. C. Zhou, "Metasynthesis: M-space, M-interaction and M-computing for open complex giant systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 5, pp. 1007–1021, Sep. 2009.
[34] H. Zhu, M. C. Zhou, and R. Alkins, "Group role assignment via a Kuhn–Munkres algorithm-based solution," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 42, no. 3, pp. 739–750, May 2012.
[35] H. Zhu and M. C. Zhou, "Efficient role transfer based on Kuhn–Munkres algorithm," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 42, no. 2, pp. 491–496, Mar. 2012.
[36] C. Tarantilis, E. Zachariadis, and C. Kiranoudis, "A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 255–271, Apr. 2009.
[37] X. Hu and E. Di Paolo, "Binary-representation-based genetic algorithm for aircraft arrival sequencing and scheduling," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 2, pp. 301–310, Aug. 2008.
[38] H. Zhu and M. C. Zhou, "M–M role-transfer problems and their solutions," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 2, pp. 448–459, Mar. 2009.
[39] H. Zhu, M. Hou, C. Wang, and M. C. Zhou, "An efficient outpatient scheduling approach," *IEEE Trans. Autom. Sci. Eng*., vol. 9, no. 4, pp. 701–709, Oct. 2012.
[40] H. Zhu and M. C. Zhou, "Role transfer problems and algorithms," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 6, pp. 1442–1450, Nov. 2008.
[41] Q. Kang, M. C. Zhou, J. An, and Q. Wu, "Swarm intelligence approaches to optimal power flow problem with distributed generator failures in power networks," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 343–353, Apr. 2013.
[42] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Trans. Ind. Inf.*, vol. 10, no. 2, pp. 1273–1284, May 2014.
[43] W. Dong and M. C. Zhou, "Gaussian classifier-based evolutionary strategy for multimodal optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 6, pp. 1200–1216, Jun. 2014.
[44] J. Zhang, C. Wang, and M. C. Zhou, "Last-position elimination-based learning automata," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2484–2492, Dec. 2014.
[45] X. Zuo, C. Chen, W. Tan and M. C. Zhou, "Vehicle scheduling of urban bus Line via an improved multi-objective genetic algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 1030–1041, Apr. 2015.
[46] X. Liang, W. Li, Y. Zhang, and M. Zhou, "An adaptive particle swarm optimization method based on clustering," *Soft Comput.*, vol. 19, no. 2, pp. 431–448, Feb. 2015.