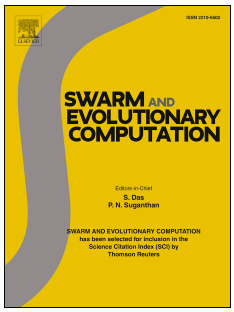

A multi-objective genetic algorithm based approach for dynamical bus vehicles scheduling under traffic congestion

Chunlu Wang, Hongyi Shi, Xingquan Zuo

PII: S2210-6502(19)30238-X

DOI: https://doi.org/10.1016/j.swevo.2020.100667

Reference: SWEVO 100667

To appear in: *Swarm and Evolutionary Computation BASE DATA*

Received Date: 31 March 2019

Revised Date: 3 February 2020

Accepted Date: 15 February 2020

Please cite this article as: C. Wang, H. Shi, X. Zuo, A multi-objective genetic algorithm based approach for dynamical bus vehicles scheduling under traffic congestion, *Swarm and Evolutionary Computation BASE DATA* (2020), doi: https://doi.org/10.1016/j.swevo.2020.100667.

A Multi-objective Genetic Algorithm based Approach for Dynamical Bus Vehicles Scheduling under Traffic Congestion

Chunlu Wang[a,c], Hongyi Shi[a,c], Xingquan Zuo[b,c]

[a]*School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China.*
[b]*School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China.*
[c]*Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, China.*

Abstract

Bus vehicle scheduling is very vital for bus companies to reduce operation cost and guarantee quality of service. Many big cities face the problem of traffic congestion, which leads to the planed vehicle scheduling scheme becoming infeasible. It is significant to study bus vehicle scheduling approaches under uncertain environments, such as traffic congestion. In this paper, a bus vehicle scheduling approach is proposed to handle the traffic congestion. It consists of three phases: firstly, a set of candidate vehicle blocks is generated once traffic congestion happens. Secondly, a non-dominated sorting genetic algorithm is adopted to select a subset of vehicle blocks from the set of candidate blocks to generate a set of Non-dominated solutions. Finally, a departure time adjustment procedure is applied to the Non-dominated solutions to further improve the quality of solutions. Experiments on a real-world bus line show that the proposed approach is able to dynamically generate scheduling schemes and significantly improve the quality of service compared to the comparative approaches.

Keywords: Bus vehicle scheduling, dynamic vehicle scheduling, multi-objective genetic algorithm, urban bus scheduling.

1. Introduction

Bus vehicles scheduling is an important part of public transportation operation plan [1-4]. Each bus line has two control points (CPs), and drivers can have a rest at each CP. Each control point has a timetable [5]. A trip is a travel of a vehicle from a CP to the other. A timetable contains a large number of start times. There must be a trip starting from each start time in the timetable. A start time is covered by a trip if the trip departs from that start time. Each start time in the timetable must be covered by exactly one trip. Bus vehicle scheduling problem (BVSP) is to arrange a number of vehicles to make their trips cover all start times in the timetable. BVSP is a key issue for vehicle companies to guarantee service quality and meanwhile reduce operational cost [6-7].

In recent years, traffic congestion occurs frequently in big cities. Vehicles may be blocked due to traffic congestion, and the travel time of some trips of vehicles is prolonged. As a result, vehicles cannot arrive at the destination CP as planed due to the congestion, thus that they cannot depart from the destination CP according to the planed times. That will lead to a large number start times in the timetable not being covered, resulting in significantly reducing the passengers' satisfaction. To ensure the quality of service, the vehicles must be rescheduled to make the times in the timetable be well covered by used vehicles. Dynamic bus vehicle scheduling problem (DBVSP) is to adjust the original schedule scheme and schedule vehicles according to the adjusted scheme to make departure times of vehicles coincide with all times in the timetable when uncertain events happen.

Different from the dynamic vehicle routing problem [54] that re-plans vehicle routes under uncertain environment, DBVSP considers the buses' travels between two CPs and adjusts the departure times of vehicles' trips when uncertain events happen. Most of existing approaches focus on the studies on dynamical scheduling approaches for vehicles breakdown or uncertain travel time of vehicles. There are few researches

on the uncertain events of traffic congestion that lasts for a relatively long period. Traffic congestion on a road for a long period will prolong the travel time of all currently running vehicles, resulting in the original scheduling scheme becoming infeasible completely. In this case, it is not easy to adjust the original scheduling scheme slightly to cope with the traffic congestion, and a new vehicle scheduling scheme must be regenerated quickly. Most of existing bus vehicle dynamic scheduling approaches are based on the adjustment of the original scheduling scheme to handle uncertain travel time and vehicles breakdown, such that they are not suitable for the traffic congestion that prolongs all vehicles' travel time for a long period.

In this paper, we propose a new dynamic bus vehicle scheduling approach to deal with the traffic congestion. It can quickly generate a new vehicle scheduling scheme once traffic congestion occurs. The approach consists of three phases: First, all times in the original timetable that have not been covered form a new timetable, and a set of candidate vehicle blocks are generated based on the new timetable. Secondly, a multi-objective genetic algorithm is used to select some subsets of vehicle blocks from the candidate blocks set to obtain a set of Non-dominated solutions. Finally, a departure time adjustment procedure (DTAP) is applied to those Non-dominated solutions to further improve them.

Contributions of this paper include: (1) Study a dynamic bus scheduling problem under large-scale traffic congestion, which affects a large number of bus schedules; (2) Propose a dynamic bus vehicle scheduling approach that is able to quickly generate a scheduling solution to the problem; (3) Compare the proposed approach with a state-of-the-art approach and a multi-objective optimization approach to show the effectiveness of the approach.

The paper is organized as follows. Section 2 reviews the related works. Section 3 introduces the dynamic bus vehicle scheduling problem. In section 4, the proposed approach is presented in details. Section 5 presents the experimental results. Section 7 summarizes the paper.

2. Related work

2.1. Bus vehicle scheduling

There are numerous approaches for BVSPs to automatically generate vehicle scheduling schemes. Some of those approaches have been proposed to solve the single depot bus vehicle scheduling problems. Haased *et al.* [9] presented an exact approach to schedule buses and crews in an urban mass transit system simultaneously. Chao *et al*. [10] proposed a modified multi-objective optimization method based on the non-dominated sorting genetic algorithm (NSGA- II) to solve the electric bus scheduling problem. Sun *et al.* [11] proposed a genetic algorithm base method for bus rapid transit scheduling problems. Kwan *et al.* [12] proposed a heuristic algorithm for bus vehicle scheduling problems. Lin *et al.* [13] established a programming model for the bus and crew scheduling problem and solve the problem by a branch-and-bound algorithm. Two objectives (the numbers of vehicles and drivers) are incorporated into a single objective. Rodrigues *et al.* [14] presented a suite of integrated algorithms and heuristics for a bus and crew scheduling problem in urban transportation systems. Haghani *et al.* [15] presented a comparative analysis of three vehicle scheduling models. Shui *et al.* [16] proposed a clonal selection algorithm based approach for a sing-depot bus vehicle scheduling problem, and applied the approach to a real-world problem.

Many approaches have been proposed for multi-depot bus vehicle scheduling problems. Gintner *et al*. [17] propose a two-phase method to solve a multiple-depot multiple-vehicle-type scheduling problem. Liu *et al.* [18] formulated a synchronized optimization model that integrates timetabling and bus scheduling, and designed a bi-level heuristic algorithm for it. Forbes *et al.* [19] proposed an exact algorithm for the multiple depot vehicle scheduling problem. Banihashemi *et al.* [20] proposed a heuristic procedure for solving real world multiple depot bus scheduling problems considering route time constraints. Wei *et al.* [21] proposed an improved ant colony optimization, which initially incorporates ant colony system, Max-min ant system and Best-worst ant system to solve the bus vehicle scheduling problem. Surapholchai *et al.*

[22] proposed an Eligen-algorithm that uses the techniques of column elimination and column generation for multiple-depot bus scheduling problems. Ceder *et al.* [23] proposed a heuristic algorithm based on the deficit function theory to solve the vehicle scheduling problem. Heuvel *et al.* [24] proposed multiple integer linear programming models and a local search algorithm to solve a timetabling problem and bus vehicle scheduling problem. Kliewer *et al.* [25] proposed a time–space network flow model for a bus vehicle scheduling problem. The advantage of such time–space network model is the reduction of the number of decision variables. Haghani *et al.* [26] presented an integer program for the bus vehicle scheduling problem with consideration of route time constraints. An exact algorithm and two heuristics are proposed for it. Guo *et al.* [27] proposed a genetic algorithm based column generation approach to solve a multi-depot electric bus vehicle scheduling problem.

2.2. Uncertain bus vehicle scheduling

Uncertain bus vehicle scheduling problems (UBVSP) are very vital since uncertain events happen frequently during the operation of vehicles. Compared to traditional bus vehicle scheduling problems, relatively limited researches have been conducted on the problem due to its complexity. Current studies on this problem can be divided into two categories: stochastic programming, and genetic algorithm.

Stochastic programming is to maximize (minimize) the mathematical expectation of the benefit (total cost). Yan *et al.* [28] developed an integrated framework for the intercity bus carriers to plan bus routes and schedule buses under stochastic travel time. The framework takes 1-6 hours for different problem instances. Naumann *et al.* [29] presented a new stochastic programming model for a robust vehicle scheduling in public transportation. Yan *et al.* [30] developed a stochastic optimization model and a robust optimization model for an inter-city bus scheduling problem. They proposed an iterative solution algorithm that takes about 5 minutes to get a solution to the models.

Genetic algorithms have been applied to UBVSP recently. Wei *et al.* [37] established a chance-constrained programming model to minimize the operation cost

of buses, and solve the model by a genetic algorithm improved based on the features of bus scheduling. Wei *et al.* [38] presented a bi-level programming model of UBVSP and solved the problem by a genetic algorithm. There exist some other methods for UBVSP. For example, Shen *et al.* [39] provided a dynamic vehicle scheduling approach based on a hierarchical task network. Wei *et al.* [40] studied a regional bus scheduling problem with grey travel time to meet some side constraints such as multi-vehicle-type, depots capacities and fueling, etc. Xuan *et al.* [41] proposed dynamic holding strategies that can improve the reliability of bus schedules. Dávid *et al.* [42] proposed a recursive and a local search algorithm to solve a dynamic vehicle scheduling problem.

For some uncertain events such as traffic congestion, it is necessary to generate a vehicle scheduling scheme quickly to satisfy the need of dynamical scheduling. For example, assume that traffic congestion happens at 9:30 AM and lasts for one hour. If we take 30 minutes to generate a vehicle scheduling scheme, it is 10:00 AM when we obtain the scheduling scheme. As a result, the obtained scheduling scheme cannot be used to schedule vehicles in the period between 9:30 and 10:00 AM. In [42], Dávid *et al.* stated that "As a result of a disruption, a new schedule has to be produced as soon as possible." However, it is not easy for existing dynamic programming and stochastic programming methods to regenerate a scheduling scheme quickly. Most of those approaches need a relative long computational time to generate a vehicle scheduling scheme. On the other hand, most of current UBVSPs consider uncertain events of vehicles breakdown and variable travel time, and seldom consider the case of traffic congestion lasting for a long period. Such traffic congestion prolongs the travel time of all running vehicles and requires regenerating a new scheduling scheme quickly, which cannot be handled by most of existing bus vehicle dynamic scheduling approaches.

3. Bus vehicle dynamic scheduling under traffic congestion

Generally, there are two CPs (CP_1 and CP_2) in most of bus lines. Assume that the

minimum rest time of a drive at a CP is R. The time from the moment of a vehicle arriving at a CP to the moment of its departure is not allowed to exceed a maximum time limit, $(R+W)$, where W is the maximum waiting time.

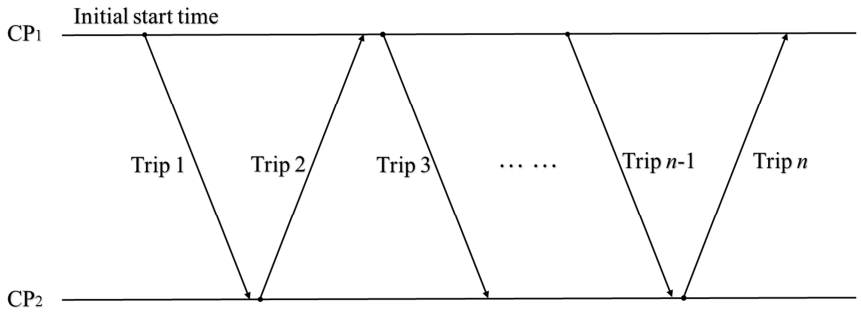

Figure 1: A vehicle block

A trip represents the travel of a vehicle from a CP to the other. A trip's travel time refers to the time that it takes to execute the trip. There is a timetable for each CP. A timetable contains the departing times of trips (start times). For each start time of a CP, there must be a vehicle that departs from the CP at the start time. To do so, we must schedule all vehicles to make all start times in the CPs be covered by trips of vehicles. The timetable is determined by the passenger's flow of a bus line. In the timetable, some segments have large time intervals while some have small ones. Large (small) time intervals mean that the passengers' flow in that time segment is small (large). Timetables of two CPs are usually different since the up and down directions of a bus line have different passenger's flow.

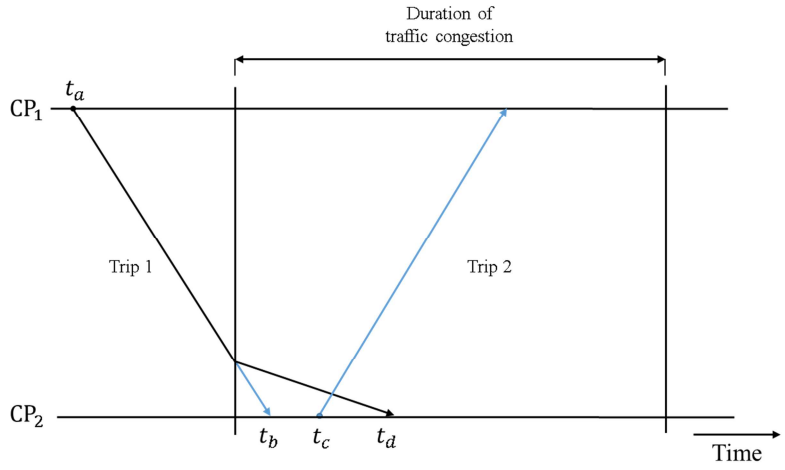

Figure 2: Traffic congestion prolongs the travel time of trips

As shown in Figure 1, a vehicle block is composed of a collection of successive trips performed by a vehicle within a day. Only after a trip of a vehicle arrives at a CP, its next trip can depart from the CP at a start time. Each trip must start from a start time in the timetable. The vehicle block in Figure 1 contains n trips. There are typically two types of blocks, namely long block and short block. A long block requires one driver and a short block requires two. The operation time of a short block equal the maximum working time of a driver, T_w, and the operation time of a long block is $2T_w$.

When traffic congestion occurs, the travel time of all trips that are running on the road becomes longer during the period of congestion. Figure 2 shows the influence of longer travel time on the vehicle scheduling scheme. According to the original scheduling scheme, trip 1 will reach CP_2 at time t_b and depart at time t_c. The t_c is a start time in the timetable of CP_2. However, traffic congestion happens before trip 1 reaches CP_2, resulting in increasing the travel time of trip 1 to make it arrive at CP_2 at t_d. Thus, it is impossible for trip 2 of the vehicle to depart from t_c. As a result, the t_c cannot be covered. Actually, when traffic congestion continues for a relatively long period, all running trips cannot arrive at the destination CP on time. It means that all start times after the beginning of traffic congestion cannot be well covered any more,

such that the original vehicle scheduling scheme becomes infeasible. The purpose of our dynamic vehicle scheduling approach is to regenerate a new scheduling scheme quickly, to cover start times that have not been covered after the traffic congestion.

The dynamic bus vehicle scheduling problem is to reschedule all used vehicles to cover all those start times that do not be covered before traffic congestion. We consider two optimization objectives: minimizing the number of uncovered start times and the number of drivers. In this problem, no extra vehicles are used after traffic congestion to save bus companies' operational cost.

One characteristic of this problem is that traffic congestion prolongs the travel time of all trips in a certain period and no running vehicles can arrive at the destination CPs as planned. As a result, those vehicles cannot depart from the destination CPs at the planed start times, resulting in many following start times not being covered. In this case, all running vehicles must be rescheduled to cover all start times that have not be covered before. Another characteristic of this problem is that traffic congestion is unpredictable and a scheduling approach must be able to generate a scheduling solution very quickly to meet the real-time need. Thus, the computational time of the solution approach must be very short. Above features of this problem make it be a challenging problem.

4. Dynamic Bus vehicle Scheduling Approach

In this paper, we propose a genetic algorithm based dynamic bus vehicle scheduling approach (GA-DBVSA) to quickly generate vehicle scheduling schemes once traffic congestion occurs. It is based on our previous work [8] and consists of three phases: (1) Generate a set of candidate vehicle blocks based on the uncovered start times in the timetable after the traffic congestion; (2) a multiobjective genetic algorithm is used to choose some block subsets from the set of candidate blocks to construct a set of Non-dominated solutions; and (3) a departure time adjustment procedure is used to further improve the Non-dominated solutions to achieve the final scheduling schemes.

The problem has two objectives. One objective is to minimize the number of uncovered start times. It tries to make as many start times as possible be covered. A large number of uncovered start times would greatly decrease the quality of service. The other objective is to minimize the number of drivers. As vehicle companies will pay more salary for more drivers, reducing the number of drivers will decrease the cost of bus companies. Note that our approach does not use extra vehicles when traffic congestion happens, and the number of used vehicles is fixed. In this case, the two objectives are conflict each other because: (1) if we want more start times to be covered, vehicles prefer to run long blocks, each of which need two drivers, resulting in increasing the number of drivers; (2) if we want to reduce the number of drivers, then vehicles prefer short blocks (each requires one driver). As a short block has half of the number of trips of a long block, the number of uncovered start times tends to increase.

4.1. Generate Candidate Blocks

When traffic congestion happens, all start times that are not covered at the moment construct a new timetable T. The purpose of our approach is to cover all start times in T by the currently used vehicles (no more extra vehicles are used). Assume that the set of start times in CP_1 is T_1, and the set for CP_2 is T_2. The new timetable $T = T_1 \cup T_2$ contains n start times. Initial start times in T refer to those start times, each of which is the departure time of the first trip of a vehicle block. Let the set of initial start times be $S = \{s_1, s_2, s_3, \dots, s_{ns}\} \subset T$, where ns is the number of initial start times.

All currently running vehicles arrive at the destination CP later than their planed arriving time due to the congestion. For each of those running vehicles, when it arrives at the destination CP, it chooses an uncovered start time as its initial start time s_i and its first trip departs from s_i. As shown in Figure 3, the trip 0 of a vehicle starts from t_a and arrives at t_b, later than the planed arriving time. Then, a start time in $[t_b + R, t_b + R + W]$ is selected as the initial start time, s_i, and the trip 1 departing

from s_i is considered as the first trip of the vehicle. After trip 1 reaches CP_1, we have two start times t_c and t_d to be selected as the vehicle's next trip after trip 1. Each such selection constructs a possible vehicle block.

The depth first search is used to generate the set of candidate blocks starting from s_i, denoted by $B_{s_i} = \{b_{s_i}^1, b_{s_i}^2, \dots, b_{s_i}^{n_i}\}$, where $b_{s_i}^j$ $(j = 1,2, \dots, n_i)$ is the jth vehicle block starting from s_i and n_i is the number of vehicle blocks whose first trip starts from s_i. The set of vehicle blocks for all initial start times in S is B=$\bigcup_{s_i \in S} B_{s_i} = \{b_1, b_2, \dots b_{nb}\}$, where $nb = \sum_{i=1}^{ns} n_i$ is the total number of candidate blocks.

During the procedure of constructing a block, the total work time of a vehicle is compared with T_w (for short block) or $2T_w$ (for long block). The construction procedure ends once the work time exceeds T_w ($2T_w$). W is set to be $(t_q - t_b)\chi$, where t_q is the start time closest to t_b and $\chi \geq 1$ is a control factor to control the number of candidate blocks in B.

The s_i is chosen to be the start time closest to the original departure time of trip 1, to make the regenerated scheduling scheme is as similar as possible to the original one. If the original departure time of trip 1 is later than t_b, we just choose the original departure time as s_i. Otherwise, the smallest start time in $[t_b + R, t_b + R + W]$ (i.e., the time closest to t_b) is chosen as s_i.

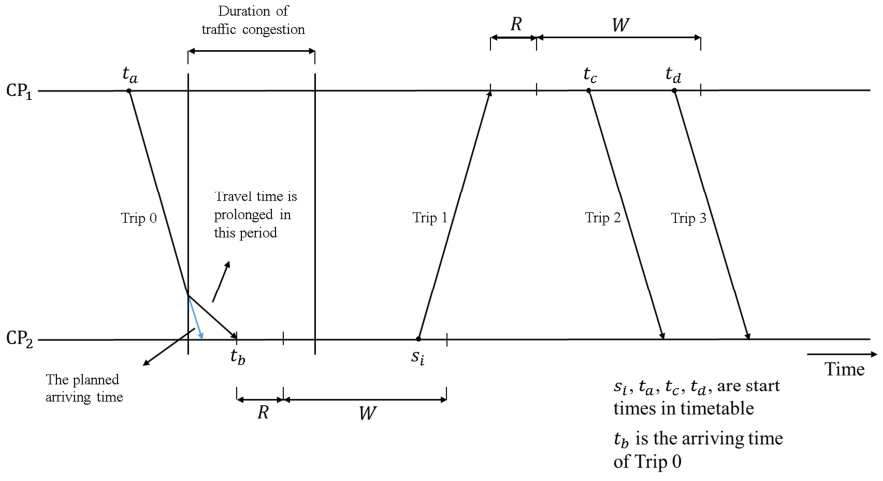

Figure 3: Generate a set of candidate vehicle blocks when traffic congestion happens.

The maximum waiting time *W* is related to the size of vehicle blocks set and can control the set size. A reasonable size of blocks set facilitates NSGA-II to find high-quality solutions.

4.2. Select block subsets to form Pareto solutions

Non-dominated sorting genetic algorithm (NSGA-II) is one of the most famous multi-objective genetic algorithms [43-44]. Many researchers have proved that NSGA-II is suitable for both of combinatorial and continuous optimization problems, including dynamic multi-objective optimizing problems. For examples, Pasandideh *et al.* [45] proposed a NSGA-II based method to solve a multi-product multi-period three-echelon supply-chain-network problem. Haghighi *et al*. [46] proposed a NSGA-II based approach to do uncertainty analysis of water supply networks. Shukla *et al*. [47] proposed an efficient multi-objective evolutionary algorithm viz. NSGA-II to solve the pollution routing problem under uncertainty. Xu *et al.* [48] proposed a cooperative co-evolutionary strategy based on environment sensitivities for solving dynamic multi-objective optimization problems. Kamjoo *et al.* [49] proposed a multi-optimization NSGA-II based approach for the design of a standalone hybrid renewable energy system under uncertainties.

There are a large number of vehicle blocks in the set *B*. A subset of vehicle blocks represents a candidate solution to the problem. Selecting block subsets from *B* to form a set of Pareto solutions is a combinatorial optimization problem. NSGA-II is adapted to select block subsets to form a set of Pareto solutions, with optimization objectives of minimizing the number of start times and drivers.

In the algorithm, each individual (solution) represents a subset of blocks. An individual (solution) refers to a vehicle scheduling scheme, which is composed of a number of vehicle blocks. Each block covers a number of start times in the timetable *T*. The roulette selection [50] is adopted, instead of the binary tournament selection in the original NSGA-II. An archive, *A*, is used to store elite solutions found during the

search procedure. After generating a set of Non-dominated solutions, DTAP is applied to those solutions to improve their quality significantly.

The steps of selecting block subsets are as follows:

Step 1: Let *gen* to be the number of generations, *gen* = 0. Generate *N* individuals by an initialization method (to be introduced next) to form the population of the first generation $P(gen)$. Set the archive set, *A*, to be empty.

Step 2: Individuals in $P(gen)$ perform crossover and mutation operations to obtain the population $Q(gen)$.

Step 3: Combine $P(gen)$ and $Q(gen)$ to construct a new population $R(gen)$, and calculate the objective function values of each individual in the population $R(gen)$. The roulette method is used to select *N* individuals from $R(gen)$ to form the population of next generation $P(gen+1)$. Update the archive set, *A*, by some elite individuals in the current population.

Step 4: Let $gen = gen + 1$. The algorithm stops if *gen* reaches the given number of generations; otherwise return to Step 2.

4.2.1. Solution coding

As per [8], each individual is coded as an integer coding and represents a subset of vehicle blocks. As shown in Figure 4, each gene in the coding corresponds to an initial start time in the timetable *T*. There are *ns* initial start times, such that the length of solution coding equals *ns*. The ith gene, b_i, in the coding corresponds to the initial start time, s_i, in the timetable. The value of b_i is a block number in B_{s_i}. $b_i = j$ indicates that the *j*th block in B_{s_i} is selected as a part of the solution represented by the coding; $b_i = 0$ indicates that the *j*th block is not selected.

b_1	b_2	… …	b_{ns}

Figure 4: Solution coding.

4.2.2. Fitness function

A solution is evaluated by two objectives, namely the number of uncovered start times in the timetable and the number of drivers. Both two objectives are expected to be as small as possible. Each solution is decoded into a vehicle scheduling scheme. The number of uncovered start times in the solution can be accounted as the first objective function value, Obj_1. A long block needs two drivers, and a short block needs one. Using the type of each block in the solution, the total number of drivers can be accounted as the second objective function value, Obj_2.

Utilizing two objective function values, the fast nondominated ranking method [43] is used to rank all individuals in the population and each individual is assigned a nondominated level represented by its ranking value. Individuals with the same ranking value indicate that they have the same nondominated hierarchy in the target space. Assume that an individual X has a ranking value i, and r is the largest ranking value. Then, the fitness value of X is calculated by

$$Fitness(X) = r - i \tag{7}$$

If two individuals have the same ranking value, their fitness values are the same. Individuals with smaller rankings have greater fitness values. This leads the algorithm to converge toward the optimal Non-dominated solutions.

4.2.3. Initializing population

Each individual in the initial population is initialized randomly, i.e., a blocks subset is randomly selected from the set B to construct an individual. In literature [8], the first trip of each vehicle must depart from CP_1. In this paper, we adapt the method in [8] to make it fit the dynamic vehicle scheduling problem. That is, the first trip of each used vehicle is allowed to depart from either CP_1 or CP_2, depending on the destination CP where it arrives. If a vehicle is on the way to a CP when traffic congestion occurs, its first trip will depart from that CP.

4.2.4. Crossover, mutation, and selection operations

Each individual in *P* (*gen*) performs the single-point crossover operation according to the crossover probability, P_c. If an individual performs the crossover operation, another individual is randomly selected from the population to perform the crossover operation with the individual, producing two new individuals. In our approach, the crossover operation is performed by the following steps: (1) select a gene randomly as the crossover point; (2) two individuals swap all their genes after the crossover point to generate two new individuals.

The mutation operation is to replace the value of a randomly chosen gene of an individual with another vehicle block according to the mutation probability P_m. For each gene with a nonzero value, its value is set to be a randomly selected block starting from the initial start time represented by the gene. All new individuals produced by the crossover and mutation operations construct the population of *Q* (*gen*).

$P(gen)$ and $Q(gen)$ are combined to construct the population $R(gen)$. The roulette selection [50] is used to select *N* individuals from $R(gen)$ to form the population of next generation, $P(gen+1)$. The roulette selection makes the algorithm has the chance to select individuals with lower fitness values.

4.2.5. Updating archive set

Different from the original NSGA-II, an archive set is used to keep elite solutions found so far. Suppose that the individual with the smallest number of uncovered start times in the population has *x* uncovered start times. Then, in each generation, all those individuals that have less than *(x+4)* uncovered start times are added into the archive set.

When the algorithm stops, the individuals with rank value 1 in the archive set construct the set of Non-dominated solutions. Note that the archive set is not updated

until 40 generations, since it is not possible to generate elite individuals within a small number of generations.

4.3. Departure time adjustment procedure

The DTAP [8] is used to improve vehicle scheduling schemes (solutions) generated by the genetic algorithm. That is, DTAP is applied to each individual in the population of the last generation of NSGA-II. The procedure includes two stages. The first stage is to adjust the trips' departure time so as to cover those uncover start times in the solution. The second one is to remove those redundant trips with the same departure times in the same CP, thereby reducing the case of covering a start time by more than one trip.

Suppose that DTAP is applied to a solution *X*, we have the following procedure.

1) Move the departure times of trips

Solution X is decoded into a block subset U. Suppose that U has c_i trips starting from $t_i \in T$. Each block $u_j \in U$ has a set of relaxation times $L_j = \{l_j^1, l_j^2, \ldots, l_j^{v_j}\}$, where v_j is the number of trips in u_j. l_j^i is the relaxation time of the ith trip of u_j and defined as the interval between the end time of the ith trip and the departure time of the $(i+1)$th trip.

Each start time $t_i \in T_1$ is checked to judge whether c_i is zero, and then, check each time $t_i \in T_2$. If $c_i = 0$, i.e., t_i is not covered by any trip, then, the start times of some trips are moved to try to cover t_i.

Step 1: Let $k = i - 1$.

Step 2: If $c_k = 0$, go to Step 6. If $c_k = 1$, then there must exist one trip starting from t_k, and suppose that this trip is the vth trip of $u_j \in U$. If $l_j^v - R_{min} \geq t_{k+1} - t_k$, where R_{min} is drivers' minimum rest time, moving the departure time of this trip from t_k to t_{k+1} is feasible, and let $k = k - 1$.

Step 3: Repeat Step 2 until $c_k > 1$ or $l_j^v - R_{min} < t_{k+1} - t_k$. If $c_k > 1$ and $l_j^v - R_{min} \geq t_{k+1} - t_k$, go to Step 4. If $l_j^v - R_{min} < t_{k+1} - t_k$, go to Step 6.

Step 4: Select the trip with the largest relaxation time from those trips starting from t_k, and suppose that it is the *n*th trip in $u_w \in U$. Move the departure time of this trip from t_k to t_{k+1}, and let $l_w^n = l_w^n - (t_{k+1} - t_k)$, $l_w^{n-1} = l_w^{n-1} - (t_{k+1} - t_k)$, $c_k = c_k - 1$, $c_{k+1} = c_{k+1} + 1$, and $k = k + 1$.

Step 5: Repeat Step 4 until $k = i$ (forward adjustment stops). Go to Step 7.

Step 6: Start backward adjustment, which is similar to the forward adjustment, except that each start time after t_i is checked.

Step 7: The adjustment procedure stops.

After executing the aforementioned procedure for each uncovered start time, there may also exist uncovered start times in T. Then, the aforementioned procedure is repeated again for each uncovered start time until the number of uncovered start times in T cannot be reduced.

2) Remove redundant trips

Suppose that the departure times of trips in each block $u_j \in U$ are denoted by $\{t_j^1, t_j^2, \ldots, t_j^{v_j}\}$. The following steps are used to remove redundant trips from u_j.

Step 1: Let $k = 1$.

Step 2: If both t_j^k and t_j^{k+1} are covered by more than one trip, then the *i*th and $(i + 1)$th trips of u_j are removed from u_j. Let $k = k + 1$.

Step 3: If $k = v_j$, then stop; otherwise, return to Step 2.

4.4. Time complexity analysis

The approach involves three phases. The time complexity of each phase is analysed as follows. In the first phase, the depth-first search is utilized to generate a candidate block set, and its time complexity is $O(n^2)$. In the second phase, the multi-objective genetic algorithm is used to select block subsets to form non-dominated solutions. During the search of genetic algorithm, evaluating an individual needs to compare with all other individuals in the population, and its time complexity is $O(n)$. Since each individual needs to be evaluated, the genetic algorithm's time complexity

is $O(n^2)$. In the third phase, running DTAP to cover a start time needs to move at most n start times, such that the time complexity is $O(n)$. For the worst case, each start time needs to be adjusted and the time complexity is $O(n^2)$. Therefore, the total time complexity of GA-DBVSA is $O(n^2)$.

5. Experimental Results

The proposed approach (GA-DBVSA) is applied to a real-world bus line in Nanjing city, China. The bus line information is shown in Table 1. During the peak and flat peak periods, the travel time of trips is 35 minutes, and the travel time during the low peak period is 32 minutes. The peak periods are 6:30-9:00 and 17:00-20:00, the flat peak periods are 9:00-17:00 and 20:00-21:00, and the low peak periods are 4:30-6:30 and 21:00-01:05 (next day). The driver's minimum rest time is 8 minutes. A driver's maximum working time is not allowed to exceed 8 hours.

To show the effectiveness of GA-DBVSA in solving DBVSP, we assume that traffic congestion happens twice in a day: first happens at 9:30 AM and lasts for one hour, and secondly at 11:30 AM and lasts for one hour too. That means when the congestion happens at 9:30 AM, GA-DBVSA generates a new scheduling scheme based on the original scheme; and when the congestion happens at 11:30 AM, GA-DBVSA creates another new scheduling scheme based on the scheduling scheme generated at 9:30 AM.

The travel time of all trips in the periods of congestions becomes longer and is set to be the normal travel time multiplied by an abnormal coefficient 1.2, which means the travel time in the most serious congestion period. Such setting can improve the robustness of the scheduling solution created by GA-DBVSA. That is because the actual travel time of some trips may vary during the congestion periods but must be shorter than the travel time in the most congestion period. In this case, vehicles performing those trips will arrive at the destination CP earlier than planned. It means that the created scheduling solution is still feasible for the uncertain travel time in practice.

Table 1: Information of the bus line

Parameters	Values
Number of CPs	2
Total distance	10.7 km
Number of stations	17
The first start time	4:30
The last start time	1:05, next day
Number of start times for CP_1	397
Number of start times for CP_2	397
The maximum working time of the driver T_w	8 h
Maximum waiting time	4 min

5.1. Comparative approaches

In [42], a local search-based algorithm (LS-A) was proposed for DBVSP. As LS-A is a state-of-the-art algorithm for DBVSP, we compare GA-DBVSA against it. In LS-A, trips unable to be run (dTrips) form an infeasible block called duty *dt.* All blocks form a set called vDuties. tRange represents periods of occurring congestions. A local search with two steps neighborhood transformation is used to minimize the number of trips in *dt* to obtain a feasible solution. The first step moves a trip from block *i* to *j* during tRange. The second step swaps a trip in block *i* and a corresponding trip in block *j* during tRange. LS-A cannot change the number of drivers, such that LS-A has the only objective of minimizing the number of uncovered start times.

As GA-DBVSA is a multi-objective algorithm, we also compare it with a popular multi-objective evolutionary algorithm -- MOEA/D [50]. In the MOEA/D based approach (MOEA-A), MOEA/D is used to select block subsets to obtain a set of Pareto solutions, instead of NSGA-II. The MOEA-A adopts the same solution coding as GA-DBVSA.

5.2. Algorithm parameters

For GA-DBVSA, the code length of an individual equals the number of initial start times, which is chosen as 280. The minimum rest time R is set as 8 minutes and the

controlling factor χ is set to be 1.2. Parameters of the genetic algorithm are set as N=800, $P_c = 0.7$ and $P_m = 0.00125$ after brief experiments. The maximum number of generations is given by 80.

For LS-A, the maximum number of iterations is set as 50, as suggested in [42]. For MOEA-A, the population size is set to be 800 and the number of neighbors is set as 8. Other parameters of MOEA-A are set to be the same as those of GA-DBVSA. To make a fair comparison, MOEA/D is given the same number of evaluations as NSGA-II.

5.3. Experimental results

The GA-DBVSA is coded in Microsoft Visual C++ and run on a PC with Windows 10 operating system, 2.8GHz i7 CPU and 16G RAM.

The method in [8] is applied the vehicle scheduling of the bus line without considering traffic congestion to generate a vehicle scheduling scheme as the original vehicle schedule scheme, which is executed before the traffic congestions. The scheduling scheme is shown in Figure 5. The total number of used vehicles is 46. The vehicles are scheduled according the original scheduling scheme before traffic congestions. We observe that almost no start times can be well covered after the traffic congestion if the vehicles are still scheduled according to the original scheme.

When the first traffic congestion occurs at 9:30AM, Each of GA-DBVSA and MOEA-A performs 30 independent runs to generate 30 sets of non-dominated solutions (scheduling schemes). LS-A is not a stochastic algorithm, such that it is run only once to generate a feasible solution. When the second traffic congestion occurs at 11:30 AM, each of GA-DBVSA and MOEA-A is also run for 30 times, and LS-A is used to generate a feasible solution again.

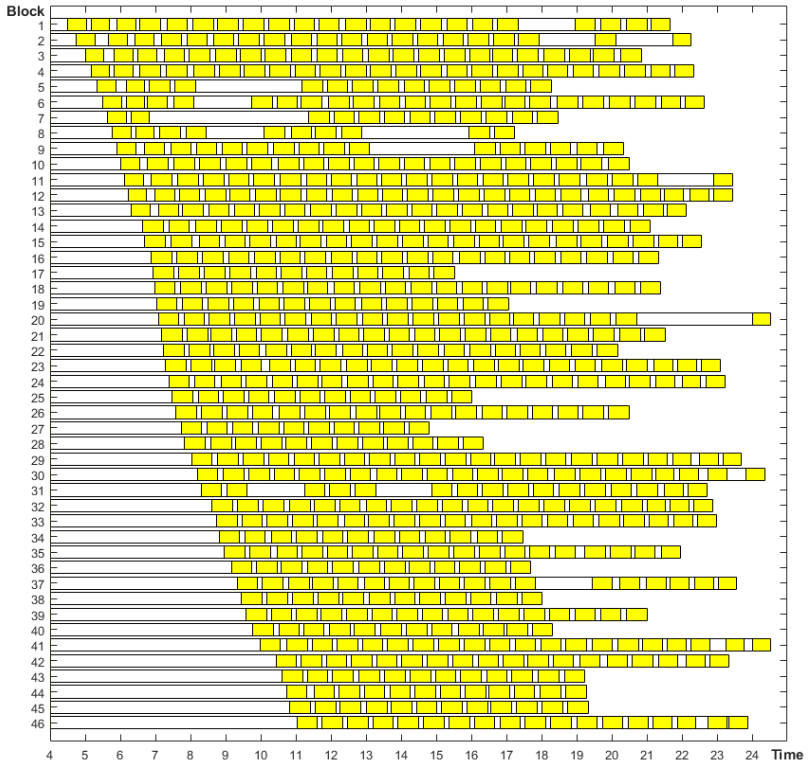

Figure 5: The original vehicle scheduling scheme.

Tables 2 (3) shows the average and standard deviation of objective function values of solutions created by GA-DBVSA and MOEA-A, objective function values of the solution created by LS-A and the average used time for the congestion at 9:30 (11:30) AM.

For each congestion, the numbers of used vehicles (blocks) of solutions generated by three approaches are all 46 and equal the numbers of used vehicles before traffic congestions. The average Obj_2 values of GA-DBVSA are about 87 for both congestions, which are smaller than those of LS-A (91) and MOEA-A (about 90). The average Obj_1 values of GA-DBVSA are 5.5 and 14.1 for the first and second congestion, respectively, which are similar than those of MOEA-A and LS-A. Compared to MOEA-A, GA-DBVSA can obtain scheduling schemes with smaller number of drivers. Compared to LS-A, GA-DBVSA can obtain scheduling schemes

with smaller number of drivers and uncovered start times. The runtime of GA-DBVSA is very short (less than 1 minute) and it is faster than MOEA-A (about 6 minutes) and LS-A (about 2 minutes).

Table 2: Objective function values of solutions created by each approach for the congestion at 9:30 AM.

		Obj_1	Obj_2	Used time (s)
GA-DBVSA	Average	5.5	86.9	48.5
	Standard deviation	0.92	1.31	1.43
MOEA-A [50]	Average	6.2	90.5	347.9
	Standard deviation	1.75	0.89	10.89
LS-A [42]		18	91	126

Table 3: Objective function values of solutions created by each approach for the congestion at 11:30 AM.

		Obj_1	Obj_2	Used time (s)
GA-DBVSA	Average	14.1	87.3	47.4
	Standard deviation	2.46	1.53	3.26
MOEA-A [50]	Average	12.8	90.0	352.1
	Standard deviation	1.28	1.06	12.17
LS-A [42]		17	91	146

Hypervolume (HV) and C-metric are two common metrics to evaluate the performance of multi-objective optimization algorithms [52], [53]. In the objective space, let (r_1, r_2) be the reference point which is dominated by any Pareto solution generated by GA-DBVSA, MOEA-A and LS-A. In this paper, the r_i (i=1, 2) is set as one plus the greatest value of the ith objective among all the Pareto solutions created by the three approaches. The HV value of an approach is the area of the region dominating (r_1, r_2) and dominated by the Pareto solutions generated by the approach, and is calculated by 2 steps: (1) sorting the Pareto solutions according to Obj_1 in descending order, and let (s_1^i, s_2^i) be the ith solution after sorting. Let $i = 1$, $Area = (r_1 - s_1^i) \times (r_2 - s_2^i)$ and $min = s_1^i$; (2) $i = i + 1$, $Area = Area + (min - s_1^i) \times (r_2 - s_2^i)$, and $min = s_1^i$. Repeat (2) until i reaches the size of the solution set, then $HV = Area$. The average HV is the mean of the HV values of 30 Pareto solution sets generated by each approach. A higher HV value means higher solution quality. The C-metric value $C(A, B)$ represents the percentage of solutions

generated by approach B that are dominated by at least one solution generated by approach A. The average C-metric is calculated by comparing each pair of Pareto solution sets from approaches A and B.

The average HV and C-metric values of the three approaches are presented in Table 4 and 5, respectively. Table 4 shows that GA-DBVSA obtains much greater average HV values than other two approaches for both congestions, indicating that GA-DBVSA finds the best Pareto solutions amongst three approaches. Table 5 shows that a large number of solutions found by other approaches are dominated by those founded by GA-DBVSA, and few solutions found by GA-DBVSA are dominated by those of other approaches. Table 6 presents the nonparametric statistical hypothesis tests (Sign and Wilcoxon signed ranks tests) on HV and C-metric values of the three approaches. Table 6 shows that p-values are less than significant level (0.05), indicating that GA-DBVSA significantly outperforms MOEA-A and LS-A in both terms of HV and C-metric values.

Table 4: Average HV values of the three approaches.

Approaches	HV values for 9:30 AM congestion	HV values for 11:30 AM congestion
GA-DBVSA	**83.8**	**37.1**
MOEA-A [50]	27.6	15.1
LS-A [42]	2	7

Table 5: Average C-metric values of the three approaches.

	C-metric for 9:30 AM congestion	C-metric for 11:30 AM congestion
C(GA-DBVSA, MOEA-A [50])	**0.5166**	**0.3055**
C(MOEA-A [50], GA-DBVSA)	0.0000	0.0150
C(GA-DBVSA, LS-A [42])	**1.0000**	**1.0000**
C(LS-A [42], GA-DBVSA)	0.0000	0.0000

Table 6: Nonparametric statistical hypothesis test of GA-DBVSA, MOEA-A and LS-A.

	HV values (9:30 AM/11:30 AM)		C-metric values (9:30 AM/11:30 AM)	
GA-DBVSA versus	MOEA-A [50]	LS-A [42]	MOEA-A [50]	LS-A [42]
Wins	30/24	30/30	23/17	30/30
Equals	0/5	0/0	7/11	0/0
Loses	0/1	0/0	0/2	0/0
Sign test (p-value)	<0.001/<0.001	<0.001/<0.001	<0.001/0.001	<0.001/<0.001

Wilcoxon (*p*-value)	<0.001/<0.001	<0.001/<0.001	<0.001/<0.001	<0.001/<0.001

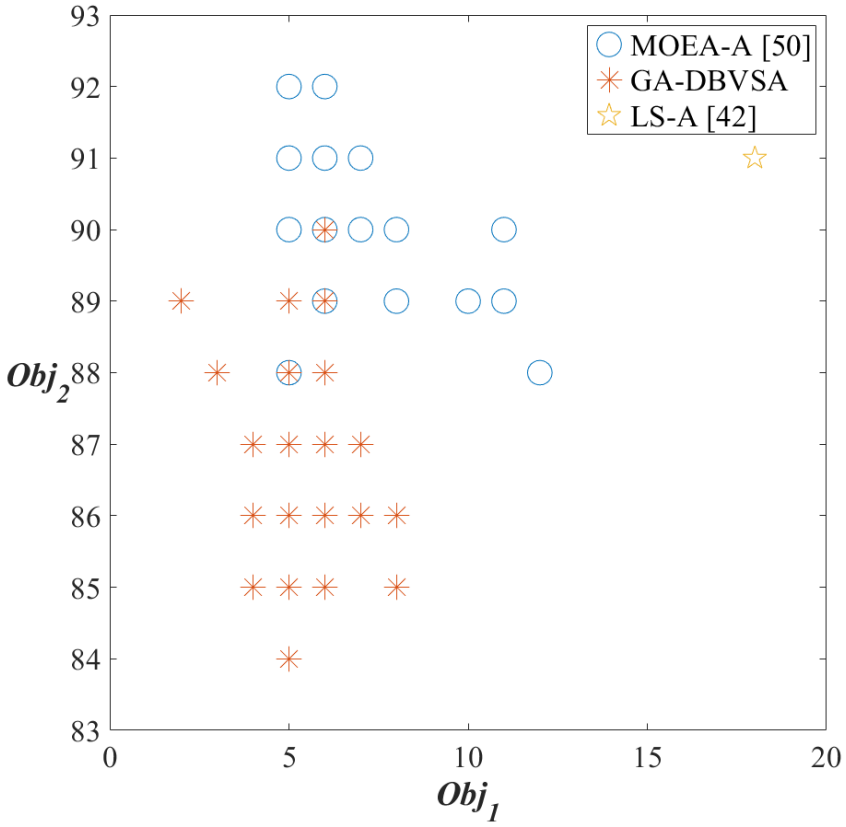

Figure 6: Non-dominated solutions obtained by the three approaches for 9:30 AM congestion.

Solutions produced by 30 runs of each approach are shown in Figures 6 and 7. Figure 6 demonstrates the non-dominated solutions obtained by the three approaches for congestion at 9:30 AM. Figure 7 shows those solutions for the congestion at 11:30 AM. Note that all solutions generated in one run of GA-DBVSA are non-dominated, but they may be dominated by solutions from other runs of GA-DBVSA. Figures 6 and 7 show that GA-DBVSA is able to find better solutions than MOEA-A and LS-A.

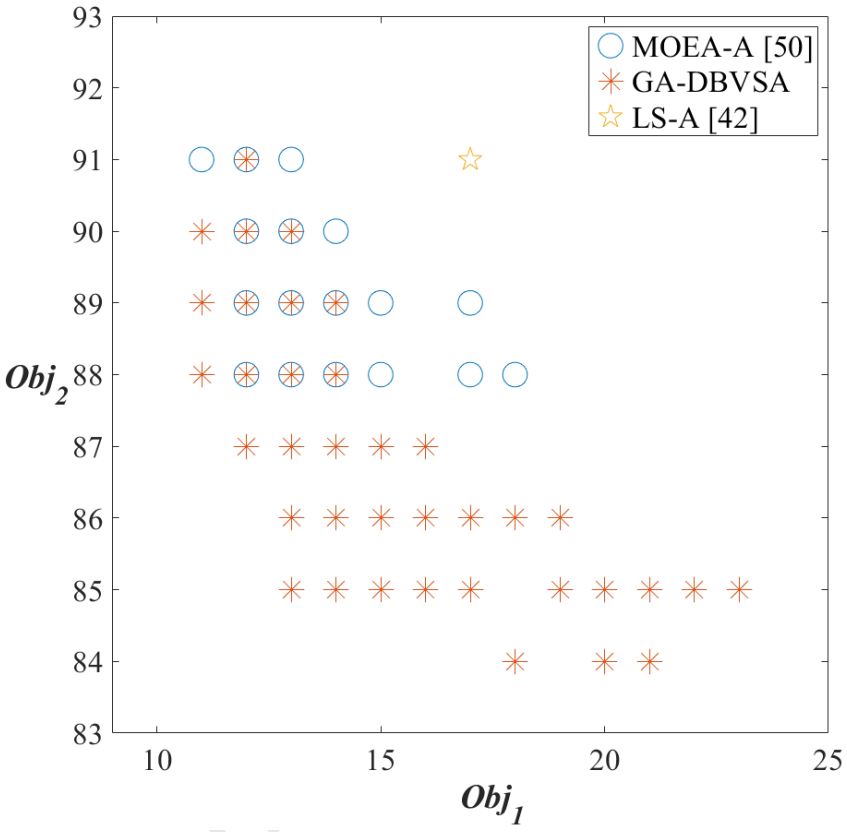

Figure 7: Non-dominated solutions obtained by the three approaches for 11:30 AM congestion.

To demonstrate the scheduling schemes generated by GA-DBVSA, a solution in the set of non-dominated solutions created by the first run for the congestion at 9:30 AM is shown in Figure 8. During the execution of the solution, another new solution is created by the first run of GA-DBVSA for the congestion at 11:30 AM and is shown in Figure 9.

In Figures 8 and 9, rectangles in yellow represent trips that have been finished before the first traffic congestion (9:30 AM). Rectangles marked in orange colour are trips that are running on the road when the congestion happens. The travel speed of those trips is normal before the traffic congestion occurs and becomes slow after the congestion happens. Thus, the travel time of those trips is quite different (those rectangles do not have the same length). Brown rectangles represent those trips that start during the period of traffic congestion (9:30 AM-10:30 AM or 11:30 AM-12:30

AM). The speed of all those trips becomes slow, and those trips' travel time is enlarged correspondingly. Those trips have the same travel time since the release of traffic congestion is considered: even if a trip starts at the end of the time slot of traffic congestion, its travel time is still influenced become the traffic congestion cannot disappear immediately. The rectangles marked in green are those trips that start after 10:30 AM in Figure 8. In Figure 9, those green rectangles are those trips that start between 10:30 AM and 11:30 AM. In Figure 9, the pink rectangle are those trips after 12:30 AM. Those trips in green and pink have normal travel speed. Green and brown rectangles before 11:30 AM represent the scheduling scheme generated for the congestion at 9:30 AM. Pink and brown rectangles after 11:30 AM represent the scheduling scheme created for the congestion at 11:30 AM.

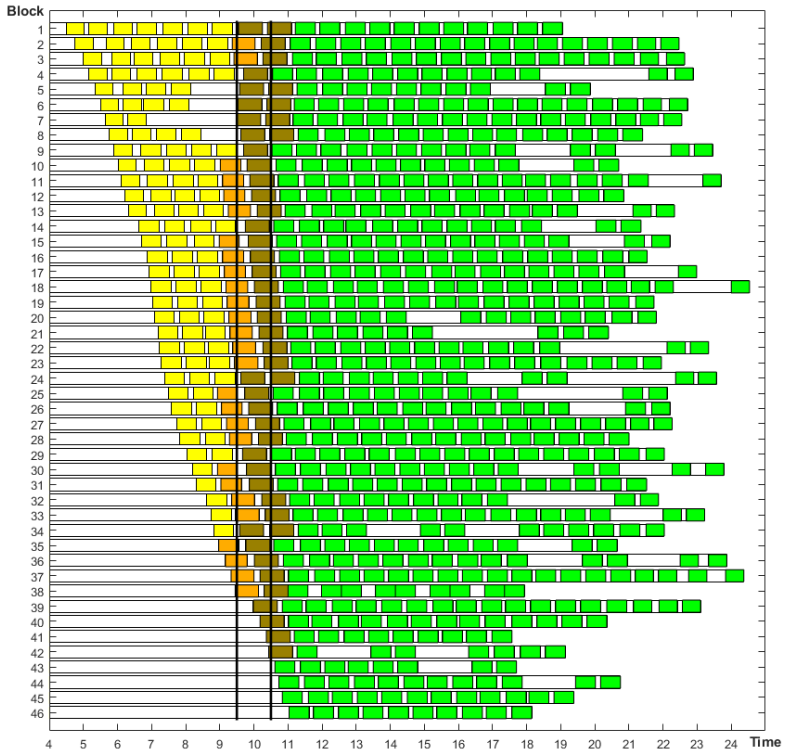

Figure 8: A new generated scheduling scheme (solution) by GA-DBVSA for congestion at 9:30 AM.

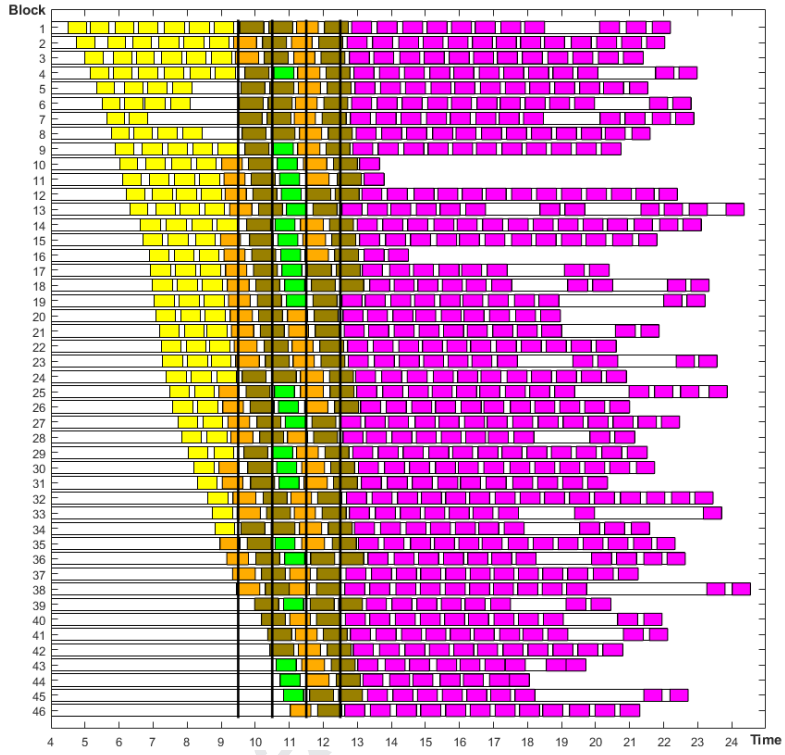

Figure 9: A new generated scheduling scheme (solution) by GA-DBVSA for congestion at 11:30 AM.

5.4. Analysis of approach's performance

In GA-DBVSA, DTAP is a key operator to improve the approach's performance. To observe the effect of DTAP on improving the solution quality, DTAP is removed from GA-DBVSA to form an algorithm named GA-DBVSA without DTAP. The GA-DBVSA without DTAP performs 30 runs for each congestion (9:30 AM and 11:30 AM) and experimental results are given in Table 7-10. Table 7 shows that the average number of uncovered start times of the generated solutions is about 106 for the congestion at 9:30 AM, and about 69 for the congestion at 11:30 AM. The numbers of uncovered start times for both congestions are much greater than that of GA-DBVSA. Tables 8 and 9 give the average HV and C-metric values of GA-DBVSA with/without DTAP. Table 8 shows that GA-DBVSA obtains much greater average HV values than GA-DBVSA without DTAP for both congestions, indicating that GA-DBVSA finds

better solutions than GA-DBVSA without DTAP. Table 9 shows that a large number of solutions found by GA-DBVSA without DTAP are dominated by those of GA-DBVSA. Table 10 presents Sign and Wilcoxon signed ranks tests on HV and C-metric values of the two approaches. The produced *p*-values are less than significant level 0.05, which indicates that GA-DBVSA significantly outperforms GA-DBVSA without DTAP and DTAP plays an important role in improving the solution quality.

Table 7: Objective function values of solutions created by GA-DBVSA without DTAP for each congestion.

GA-DBVSA without DTAP		***Obj*$_1$**	***Obj*$_2$**	**Used time(s)**
9:30 AM	Average	105.8	86.8	47.4
	Standard deviation	7.47	1.26	2.09
11:30 AM	Average	69.1	86.4	47.6
	Standard deviation	7.41	1.57	2.61

Table 8: Average HV values of the two approaches.

Approaches	HV values for 9:30 congestion	HV values for 11:30 congestion
GA-DBVSA	**495.8**	**264.3**
GA-DBVSA without DTAP	78.9	112.6

Table 9: Average C-metric values of the two approaches.

	C-metric for 9:30 congestion	C-metric for 11:30 congestion
C(GA-DBVSA, GA-DBVSA without DTAP)	**0.6472**	**0.5805**
C(GA-DBVSA without DTAP, GA-DBVSA)	0.0000	0.0000

Table 10: Nonparametric statistical hypothesis tests of GA-DBVSA with/without DTAP.

GA-DBVSA versus GA-DBVSA without DTAP	HV values (9:30 AM/11:30 AM)	C-metric values (9:30 AM/11:30 AM)
Wins	30/29	29/29
Equals	0/0	1/1
Loses	0/1	0/0
Sign test (*p*-value)	<0.001/<0.001	<0.001/<0.001
Wilcoxon (*p*-value)	<0.001/<0.001	<0.001/<0.001

GA-DBVSA has two main parameters of crossover rate (P_c) and mutation rate (P_m). To observe the effect of the two parameters on the algorithm's performance, we set

different values of P_c and P_m. For each parameter value, 30 runs of GA-DBVSA are conducted while keeping other parameters unchanged. Tables 11 and 12 show the influence of P_c on the algorithm's performance for each congestion. GA-DBVSA with different crossover rates generates solutions with similar numbers of uncovered start times and drivers. The runtime of DBVSA with different crossover rates is also similar. From Tables 13 and 14, we observe that GA-DBVSA with mutation rates in the range of [0.001, 0.01] can also obtain similar solutions. That indicates that setting different crossover and mutation rates has little influence on the algorithm's performance.

Table 11: Influence of the crossover rates on the algorithm's performance for 9:30 AM congestion.

Crossover rates	Obj_1	Obj_2	Used time (s)
0.5	5.8	87.3	41.7
0.7	5.5	86.9	48.5
0.9	5.6	87.1	58.1

Table 12: Influence of the crossover rates on the algorithm's performance for 11:30 AM congestion.

Crossover rates	Obj_1	Obj_2	Used time (s)
0.5	13.8	87.5	41.7
0.7	14.1	87.3	47.4
0.9	14.0	87.6	56.7

Table 13: Influence of the mutation rates on the algorithm's performance for 9:30 AM congestion.

Mutation rates	Obj_1	Obj_2	Used time (s)
0.00125	5.5	86.9	48.5
0.005	5.6	87.3	48.3
0.01	5.5	87.3	48.9

Table 14: Influence of the mutation rates on the algorithm's performance for 11:30 AM congestion.

Mutation rates	Obj_1	Obj_2	Used time (s)
0.00125	14.1	87.3	47.4
0.005	13.4	87.7	47.6
0.01	13.4	87.8	48.1

6. Conclusions

In this paper, we study a dynamic bus vehicle scheduling problem under traffic congestion and propose a multi-objective genetic algorithm based dynamic scheduling approach (GA-DBVSA) for the problem. The approach consists of three stages: first, all uncovered start times in the timetable after the traffic congestion construct a new timetable, and then a set of candidate vehicle blocks is created based on the new timetable; secondly, a non-dominated sorting genetic algorithm is used to select the subsets of blocks from the set of candidate blocks as non-dominated solutions; finally, a departure time adjustment procedure is applied to the solutions to achieve the final solutions.

GA-DBVSA is applied to a real-world bus vehicle dynamic scheduling problem. Experiments show that GA-DBVSA is very effective. It is able to generate high quality scheduling schemes (solutions) quickly (within one minute) once the traffic congestion occurs and does not need extra vehicles, thereby ensuring the quality of service under the condition of not increasing the operation cost.

In the future, we plan to study DBVSP from two aspects: (1) improve GA-DBVSA to solve the dynamic electric bus vehicle scheduling problem considering the recharging time and limited travel range; and (2) study robust and preference-based multi-objective scheduling approaches [55-56] for DBVSP.

Acknowledgment

This work was supported by National Natural Science Foundation of China under Grants 61873040 and supported in part by National Key Research and Development Program of China under Grant 2017YFB0803301-3.

References

[1] V. Guihaire, J. Hao, Transit Network Design and Scheduling: A Global Review, Transportation Research Part A: Policy and Practice 42 (2008) 1251-1273.

[2] A. Ceder, Optimal Multi-Vehicle Type Transit Timetabling and Vehicle Scheduling, Procedia - Social and Behavioral Sciences 20 (2011) 19-30.

[3] A. S. Mohaymany, S. M. M. Amiripour, Creating Bus Timetables under Stochastic Demand, International Journal of Industrial Engineering 20 (2009) 83-91.

[4] K. Kepaptsoglou, M. Karlaftis, Transit Route Network Design Problem: Review, Journal of Transportation Engineering 135 (2009) 491-505.

[5] P. Gao, X. Zuo, Q. Bian, X. Zhao, A Memetic Algorithm to Optimize Bus Timetable with Unequal Time Intervals, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2019, pp. 1336-1344.

[6] F. Milla, D. Saez, C. E. Cortes, A. Cipriano, Bus-Stop Control Strategies Based on Fuzzy Rules for the Operation of a Public Transport System, IEEE Transactions on Intelligent Transportation Systems 13 (2012) 1394-1403.

[7] Y. Fang, F. Chu, S. Mammar, M. Zhou, Optimal Lane Reservation in Transportation Network, IEEE Transactions on Intelligent Transportation Systems 13 (2012) 482-491.

[8] X. Zuo, C. Chen, W. Tan, M. C. Zhou, Vehicle Scheduling of an Urban Bus Line via an Improved Multiobjective Genetic Algorithm, IEEE Transactions on Intelligent Transportation Systems 16 (2015) 1030-1041.

[9] K. Haase, G. Desaulniers, J. Desrosiers, Simultaneous Vehicle and Crew Scheduling in Urban Mass Transit Systems, Transportation Science 35 (2001) 215-343.

[10] Z. Chao, X. Chen, Optimizing Battery Electric Bus Transit Vehicle Scheduling with Battery Exchanging: Model and Case Study, Procedia - Social and Behavioral Sciences 96 (2013) 2725-2736.

[11] C. Sun, W. Zhou, Y. Wang, Scheduling Combination and Headway Optimization of Bus Rapid Transit, Journal of Transportation Systems Engineering and Information Technology 8 (2008) 61-67.

[12] R. S. K. Kwan, M. A. Rahin, Object Oriented Bus Vehicle Scheduling -- The Boost System, in: Computer-Aided Transit Scheduling, 1999, pp.177-191.

[13] Y. Lin, S. Pan, L. Jia, N. Zou, A Bi-Level Multi-objective Programming Model for Bus Crew and Vehicle Scheduling, in: World Congress on Intelligent Control and Automation, 2010, pp. 2328-2333.

[14] M. M. Rodrigues, C. C. D. Souza, A.V. Moura, Vehicle and Crew Scheduling for Urban Bus Lines, European Journal of Operational Research 170 (2006) 844-862.

[15] A. Haghani, M. Banihashemi, K. Chiang, A Comparative Analysis of Bus Transit Vehicle Scheduling Models, Transportation Research Part B: Methodological 37 (2003) 301-322.

[16] X. Shui, X. Zuo, C. Chen, A. E. Smith. A Clonal Selection Algorithm Based Bus Vehicle Scheduling Approach, Applied Soft Computing 36 (2015) 36-44.

[17] V. Gintner, N. Kliewer, L. Suhl, Solving Large Multiple-Depot Multiple-Vehicle-Type Bus Scheduling Problems in Practice, OR Spectrum, 27 (2005) 507–523.

[18] Z. Liu, J. Shen, Regional Bus Operation Bi-Level Programming Model Integrating Timetabling and Vehicle Scheduling, Systems Engineering - Theory & Practice 27 (2007) 135-141.

[19] M. A. Forbes, J. N. Holt, A. M. Watts, An Exact Algorithm for Multiple Depot Bus Scheduling, European Journal of Operational Research 72 (1994) 115-124.

[20] M. Banihashemi, A. Haghani, Optimization Model for Large-Scale Bus Transit Scheduling Problems, Transportation Research Record: Journal of The Transportation Research Board 1733 (2000) 23-30.

[21] M. Wei, W. Jin, W. Fu, X. Hao, Improved Ant Colony Algorithm for Multi-Depot Bus Scheduling Problem with Route Time Constraints, in: World Congress on Intelligent Control and Automation, 2010. pp. 4050-4053.

[22] C. Surapholchai, G. Reinelt, H.G. Bock, Solving City Bus Scheduling Problems in Bangkok by Eligen-Algorithm, in: Modeling, Simulation and Optimization of Complex Processes, 2008, pp. 557-564.

[23] R. Ceder, Public-Transport Vehicle Scheduling with Multi Vehicle Type, Transportation Research Part C: Emerging Technologies 19 (2011) 485-497.

[24] A. Heuvel, J. Akker, M. Niekerk, Integrating Timetabling and Vehicle Scheduling in Public Bus Transportation, Department of Information and Computing Sciences, Utrecht University, Holanda (2008).

[25] N. Kliewer, T. Mellouli, L. Suhl, A Time–Space Network Based Exact Optimization Model for Multi-Depot Bus Scheduling, European Journal of Operational Research 175 (2006) 1616-1627.

[26] A. Haghani, M. Banihashemi, Heuristic Approaches for Solving Large Scale Bus Transit Vehicle Scheduling Problem with Route Time Constraints, Transportation Research Part A: Policy and Practice 36 (2002) 309-333.

[27] C. Guo, C. Wang, X. Zuo, A Genetic Algorithm Based Column Generation Method for Multi-Depot Electric Bus Vehicle Scheduling, in: Proceedings of The Genetic and Evolutionary Computation Conference Companion, 2019, pp. 367-368.

[28] S. Yan, C. Tang, An Integrated Framework for Intercity Bus Scheduling under Stochastic Bus Travel Times, Transportation Science 42 (2008) 263-404.

[29] M. Naumann, L. Suhl, S. Kramkowski, A Stochastic Programming Approach for Robust Vehicle Scheduling in Public Bus Transport, Procedia - Social and Behavioral Sciences 20 (2011) 826-835.

[30] S. Yan, C. Tang, Inter-City Bus Scheduling Under Variable Market Share and Uncertain Market Demands, Omega 37 (2009) 178-192.

[31] D, Lawrence, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York (1991).

[32] J. Horn, N. Nafpliotis, D. E. Goldberg, A Niched Pareto Genetic Algorithm for Multiobjective Optimization, in: IEEE World Congress on Computational Intelligence, 1994, pp. 82-87.

[33] C. Qian, Y. Yu, Z. H. Zhou, An Analysis on Recombination in Multi-objective Evolutionary Optimization, Artificial Intelligence 204 (2013) 99-119.

[34] Y. Qi, F. Liu, M. Liu, M. Gong, L. Jiao, Multi-objective Immune Algorithm with Baldwinian Learning, Applied Soft Computing 12 (2012) 2654-2674.

[35] X. Zuo, H. Mo, J. Wu, A Robust Scheduling Method Based On Multiobjective Immune Algorithm, Information Sciences 179 (2009) 3359-3369.

[36] L. Tang, X. Wang, A Hybrid Multiobjective Evolutionary Algorithm for Multiobjective Optimization Problems, IEEE Transactions on Evolutionary Computation 17 (2013) 20-45.

[37] M. Wei, W. Jin, B. Sun, Model And Algorithm for Regional Bus Scheduling with Stochastic Travel Time, Journal of Highway and Transportation Research and Development 10 (2011): 151-156.

[38] M. Wei, B. Sun, W. Jin, A Bi-Level Programming Model for Uncertain Regional Bus Scheduling Problems, Journal of Transportation Systems Engineering and Information Technology 13 (2013) 106-112.

[39] Y. Shen, Z. Zeng, Z. Wu, Dynamic Vehicle Scheduling Based on HTN, in: Chinese Control Conference, 2017, pp. 3066-3071.

[40] M. Wei, B. Sun, W. Jin, Model and Algorithm of Regional Bus Scheduling with Grey Travel Time, Journal of Transportation Systems Engineering and Information Technology 12 (2012) 106-112.

[41] Y. Xuan, J. Argote, C. F. Daganzo, Dynamic Bus Holding Strategies for Schedule Reliability: Optimal Linear Control and Performance Analysis, Transportation Research Part B: Methodological 45 (2011) 1831-1845.

[42] B. Dávid, M. Krész, The Dynamic Vehicle Rescheduling Problem, Central European Journal of Operations Research 25 (2017) 809-830.

[43] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2002) 182 – 197.

[44] F. A. Kidwai, B. R. Marwah, K. Deb, M. R. Karim, A Genetic Algorithm Based Bus Scheduling Model for Transit Network, in: Proceedings of the Eastern Asia Society for Transportation Studies, 2005, pp. 477-489.

[45] S. H. R. Pasandideh, S. T. A. Niaki, K. Asadi, Bi-Objective Optimization of A Multi-Product Multi-Period Three-Echelon Supply Chain Problem under Uncertain Environments: NSGA-II and NRGA, Information Sciences 292 (2015) 57-74.

[46] A. Haghighi, A.Z. Asl, Uncertainty Analysis of Water Supply Networks Using the Fuzzy Set Theory and NSGA-II, Engineering Applications of Artificial Intelligence 32 (2014) 270-282.

[47] A. K. Shukla, R. Nath, P. K. Muhuri, NSGA-II Based Multi-objective Pollution Routing Problem with Higher Order Uncertainty, in: IEEE International Conference on Fuzzy Systems, 2017, pp. 1-6.

[48] B. Xu, Y. Zhang, D. Gong, Y. Guo, M. Rong, Environment Sensitivity-Based Cooperative Co-Evolutionary Algorithms for Dynamic Multi-objective Optimization, IEEE/ACM Transactions on Computational Biology and Bioinformatics 15 (2018) 1877-1890.

[49] A. Kamjoo, A. Maheri, A. M. Dizqah, G. A. Putrus, Multi-objective Design under Uncertainties of Hybrid Renewable Energy System Using NSGA-II and Chance Constrained Programming, International Journal of Electrical Power & Energy Systems 74 (2016) 187-195.

[50] D. Goldberg, Genetic Algorithm in Search, Optimization and Machine Learning, Massachusetts. Addison-Wesley, Boston (1989).

[51] Q. Zhang, H. Li, MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition, IEEE Transactions on Evolutionary Computation 11 (2007) 712-731.

[52] C. Lu, L. Gao, X. Li, S. Xiao, A Hybrid Multi-objectives Grey Wolf Optimizer for Dynamic Scheduling in a Real-World Welding Industry, Engineering Applications of Artificial Intelligence 57 (2017) 61-79.

[53] J. Long, Z. Sun, P. M. Pardalos, Y. Hong, S. Zhang, C. Li, A Hybrid Multi-objective Genetic Local Search Algorithm for the Prize-Collecting Vehicle Routing Problem, Information Sciences 478 (2019) 40-61.

[54] Y. Guo, J. Cheng, S. Luo, D. Gong, Y. Xue, Robust Dynamic Multi-objective Vehicle Routing Optimization Method, IEEE/ACM Transactions on Computational Biology and Bioinformatics 15 (2018) 1891-1903.

[55] Y. Guo, H. Yang, M. Chen, J. Cheng, D. Gong, Ensemble prediction-based dynamic robust multi-objective optimization methods, Swarm and Evolutionary Computation 48 (2019) 156-171.

[56] Y. Guo, X. Zhang, D. Gong, Z. Zhang, J. Yang. Novel interactive preference-based multi-objective evolutionary optimization for bolt supporting networks. IEEE Transactions on Evolutionary Computation. 2019. DOI: 10.1109/TEVC.2019.2951217.

Author statement

Chunlu Wang: Conceptualization; Methodology; Validation; Writing - Original Draft

Hongyi Shi: Software; Methodology; Writing - Original Draft

Xingquan Zuo: Methodology; Writing - Review & Editing; Supervision; Funding acquisition

Declaration of Interest statement

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email from zuoxq@bupt.edu.cn.

Signed by all authors as follows:

Chunlu Wang, Hongyi Shi, Xingquan Zuo

February 3, 2020.