TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 05/09 pp282–292
Volume 22, Number 3, June 2017

Optimal Combinations and Variable Departure Intervals for Micro Bus System

Jiaoyang Li, Jianming Hu*, and Yi Zhang

Abstract: It is becoming increasingly difficult for Chinese citizens to access traditional public transport because of overcrowded community structures. Therefore, novel ideas are required to improve the transport system. In this respect, this study considers the design of a public transport scheduling model for a micro system. The model aims to minimize passenger waiting time and maximize number of passengers one bus carries, by simultaneously optimizing departure intervals and use of traditional and rapid buses. The model is superior to traditional models, as it analyzes the phenomena of vehicle overtaking, vehicle capacity limit, and passenger determination uncertainty. In addition, the model is a sophisticated nonlinear multi-objective optimization problem and contains more than one type of decision variable, therefore two composite algorithms, HPSO and GAPSO, are proposed, which are improvements of the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). These two algorithms are compared to the classical GA with respect to stability and effect, and the results show them to be strong in both respects. In addition, the simultaneous optimization method has evident advantages compared to single-method optimizations.

Key words: bus scheduling; optimization model; rapid bus; Particle Swarm Optimization (PSO); Genetic Algorithm (GA)

1 Introduction

With the rapid expansion of cities and dramatic increase in the volume of traffic, urban transit systems are overloaded. It is thus necessary to rely increasingly on the use of public transport as an essential and efficient way of easing traffic pressure. Novel ideas are required to improve systems.

However, there are problems associated with developing public transport systems in China, because overcrowded communities have expanded and people now live at considerable distances from their places of work. Heated debates have previously been held to solve the "last one kilometer" problem. To address these issues, the concept of the micro bus system has evolved; this is essentially the deployment of buses connecting community centers with traditional public stations, which thus reduces walking distances and saves travel time. In this paper, we focus on bus scheduling for this micro system.

Previous well-studied methods have involved optimizing bus headway (time interval between two adjacent buses on the same route). Eberlein et al.[1] and Xu et al.[2] established deadheading scheduling models to reduce large headway, by allowing empty buses to miss a number of stations. In addition, Daganzo and Pilachowski[3] adjusted bus cruising speed with bus-to-bus cooperation, and Berrebi et al.[4] designed a holding mechanism to maintain stable headways.

However, micro bus systems are not traditional public transport systems on a smaller scale, and they

• Jiaoyang Li, Jianming Hu, and Yi Zhang are with Department of Automation, Tsinghua University, Beijing 100084, China. E-mail: lijiaoyang13@mails.tsinghua.edu.cn; hujm@mail.tsinghua.edu.cn; zhyi@mail.tsinghua.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2016-03-31; revised: 2016-09-26; accepted: 2016-10-20

face the rigorous challenge of passenger flow that is not in equilibrium, either temporally or spatially. In addition, they need to be adhering to strict punctuality, because of the requirement of transporting people to traditional public stations on time to catch buses or subways during rush hours. To cope with the pressure of non-equilibrium, researchers have proposed the idea of combining the scheduling of several types of buses on the same route. For example, Xu and Pei[5] proposed a preliminary method to design bus combinations and departure frequencies by passenger flow and the bus load ratio. Furthermore, Bai et al.[6], Sun et al.[7], and Hao et al.[8] built optimization models based on minimizing passenger travel time and bus operation costs, to determine the headway and scheduling combination. However, all these models only consider uniform departure intervals; this is not adequately efficient and the models cannot be improved. Therefore, in this study, we propose a scheduling method that includes the design of a combination of rapid buses and traditional buses, with optimizing flexible departure intervals simultaneously.

The problem of bus scheduling is a classical nonlinear NP-hard problem. Many intelligent algorithms have been applied to solve this kind of problem, such as the Genetic Algorithm (GA)[9,10] and BP neural network[4]. However, to our knowledge, none of these methods perform with high efficiency when simultaneously optimizing two types of variables. In this study, we use integer variables and Boolean variables, and make the best of the model's characteristics. We propose two composite algorithms named HPSO and GAPSO, which are upgrades of GA and Particle Swarm Optimization (PSO). The performance of these algorithms is then thoroughly tested and corroborated by a comparison with the classical GA.

The remainder of the paper is organized as follows. Section 2 reveals model concepts and descriptions; Section 3 describes the model mathematically, and Section 4 shows two proposed algorithms for solving the moel. Section 5 presents a detailed case study and makes comparisons with other methods, and Section 6 presents the conclusions.

2 Background

2.1 Rapid bus

Traditional transport is not efficient enough when facing passenger flow with extreme spatial non-equilibrium. To alleviate the unbalance, we propose a new kind of bus movement that picks up passengers at stops where there is large passenger flow but does not stop at other stops. This type of movement makes the bus journey quicker; we call the buses operating within such a system, "rapid buses". If a combination of rapid buses and traditional buses are then used, the movement of public transport could be significantly improved.

2.2 Variable departure intervals

The temporal non-equilibrium of passenger flow is another crucial factor leading to the unsatisfactory performance of conventional transport. Traditionally, bus agencies divide one day into several periods according to passenger flow, and buses are dispatched at a fixed frequency during each of these periods. This works well for the first few stops, but when the bus needs to stop for a large volume of passengers to board or disembark, the headway variance is increased. In such a case, some buses catch up with those in front, while others fall far behind. In this study, we take this issue into account and propose a dispatching schedule with variable departure intervals, where intervals are rationalized as much as possible.

2.3 Overtaking

Public transport models seldom consider the phenomenon of bus overtaking, as traditional schedules always avoid bus bunching. However, overtaking is considered in our model, because rapid buses are essentially faster than traditional buses. We therefore propose an original method to deal with bus overtaking problems by providing dynamic serial numbers for buses.

2.4 Uncertainty of passenger choice

When more than one type of bus is offered to the public, people are provided with a choice and some may choose to board a traditional bus while others may prefer to wait for a rapid one. Therefore, as we cannot assert categorically that all passengers are willing to board a bus when it arrives, we use the probability to describe passenger behavior.

3 Model

3.1 Decision variable

Since we plan to optimize headway and combination simultaneously, our decision variables are an integer

array, $\{G_k\}$, which symbols the subtracting of the departure time between Bus k and Bus $k-1$; and a Boolean array, $\{E_k\}$, which symbols the type of Bus k. In addition, $E_k = 0$ represents the traditional bus, and $E_k = 1$ represents the rapid bus.

3.2 Objective function

Problems inherent in public transport scheduling require multi-objective optimizations, as proposed below. We thus focus on the most crucial factors, including the average number of passengers that one bus carries (i.e., full-load ratio of buses), W_1; the average ratio of extra practical time that passengers spend on-bus to ideal minimum time that passengers have to spend on-bus, W_2; and the average time passengers spend at-stop, W_3. The objective function, W, is the algebraic sum of these three parts, and is expressed as

$$W = A_1 \times W_1 - A_2 \times W_2 - A_3 \times W_3 \tag{1}$$

where A_1, A_2, and A_3 are the weighting coefficients. We define the solution as an optimal solution which has a maximum W and subjects to all constraints discussed in the following.

(1) Objective 1: Number of passengers that one bus carries

The average number of passengers that one bus carries is used to measure the profitability of a bus agency and its formula is as follows:

$$W_1 = \frac{\sum_{i=1}^{n-1}\sum_{k=1}^{m}\sum_{j=i+1}^{n} U_{kij}}{m(n-1)} \tag{2}$$

where m represents the number of buses, n represents the number of stops, and U_{kij} is the number of passengers who get on Bus k at Stop i and get off at Stop j.

(2) Objective 2: Ratio of passenger extra on-bus time

On-bus time always accounts for the largest proportion of passenger total traveling time. We hypothesize that buses run at a constant speed throughout the route and thus the only factor influencing passenger on-bus time is the type of bus used and the time that the bus spends at each stop. To magnify this kind of influence, we use the ratio of extra practical time spent to ideal time spent, $\dfrac{H_{kj}-H_{ki}}{\sum_{i'=i+1}^{j} L_{i'}} - 1$. The entire formula is shown as follows:

$$W_2 = \frac{\sum_{k=1}^{m}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} U_{kij} \times \left(\frac{H_{kj}-H_{ki}}{\sum_{i'=i+1}^{j} L_{i'}} - 1\right)}{\sum_{k=1}^{m}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} U_{kij}} \tag{3}$$

where H_{ki} is the time when Bus k stops at Stop i, and L_i is the driving time from Stop $i-1$ to Stop i.

(3) Objective 3: Passenger at-stop time

W_3 reflects passenger average waiting time at stops, and is equal to the quotient of the total waiting time and total number of passengers. To calculate passenger total waiting time, we consider the period from when Bus k' (the bus which arrives just before Bus k) arrives at Stop i to when Bus k arrives at Stop i. In each period, we divide passengers into three groups based on differences among their behaviors. There are $\sum_{j=i+1}^{n}(D_{k'ij} - U_{k'ij})$ passengers in Group 1, which includes passengers who arrived before Bus k' and are still waiting at Stop i; there are $\sum_{j=i+1}^{n} O_{ij} F_{ki}$ passengers in Group 2, which includes passengers who arrive during this period; and there are $\sum_{j=i+1}^{n} U_{k'ij}$ passengers in Group 3, which includes passengers who get on Bus k'. We use $t_{ik}^{(1)}$, $t_{ik}^{(2)}$, and $t_{ik}^{(3)}$ to represent waiting time of these three groups, respectively. And the average at-stop waiting time is the ratio of sum of all at-stop waiting time to sum of all passengers as follows:

$$W_3 = \frac{\sum_{i=2}^{n}\sum_{k=1}^{m}(t_{ik}^{(1)} + t_{ik}^{(2)} + t_{ik}^{(3)})}{\sum_{i=1}^{n-1}(H_{mi}\sum_{j=i+1}^{n} O_{ij})} \tag{4}$$

$$t_{ik}^{(1)} = F_{ki} \sum_{j=i+1}^{n} (D_{k'ij} - U_{k'ij}) \tag{5}$$

$$t_{ik}^{(2)} = \frac{F_{ki}}{2} \sum_{j=i+1}^{n} O_{ij} F_{ki} \tag{6}$$

$$t_{ik}^{(3)} = \left(\frac{b}{2}\sum_{j=i+1}^{n} U_{k'ij} + c\sum_{j=1}^{i-1} U_{k'ji}\right)\sum_{j=i+1}^{n} U_{k'ij} \tag{7}$$

where O_{ij} is the number of passengers arriving at Stop i whose destination is Stop j, F_{ki} is the time from Bus k' arriving at Stop i to Bus k arriving at Stop i, D_{kij} is the number of passengers at Stop i who want to go to Stop j when Bus k arrives at Stop i, and b and c are the times spent by one passenger getting on and getting off, respectively. In particular, if Bus k is the first bus to arrive at Stop i, then $t_{ik}^{(1)} = t_{ik}^{(3)} = 0$.

3.3 Recursion formula

(1) Arrival at the stop

It is firstly necessary to calculate the arrival time when Bus k arrives at Stop i. It is of note that rapid buses might not require $\sum_{i'=i+1}^{j} L_{i'}$ time to drive from

Stop i to Stop j, for it may skip several stops between i and j. Therefore, we use a nonpositive number, L'_i, to represent the difference between the times. We then need to rank the arrival time, H_{ki}, to determine which bus arrives just before Bus k, and to determine the time difference between the arrival times of the two buses. The formulae are as follows:

$$H_{ki} = H_{k(i-1)} + L_i + E_k L'_i + T_{k(i-1)} \quad (8)$$

$$F_{ki} = \min\{H_{ki} - H_{xi} | H_{ki} > H_{xi}, x \in [1,n]\} \quad (9)$$

$$k' = x, F_{ki} = H_{ki} - H_{xi}, x \in [1,n] \quad (10)$$

where T_{ki} is the time that Bus k spends at Stop i. In particular, at the first stop, $H_{k1} = \sum_{s=1}^{k} G_s + L_1$. And if Bus k is the first bus to arrive at Stop i, then $F_{ki} = H_{ki}$ and $k' = 0$.

(2) Parking at the stop

When Bus k stops, passengers begin to get off and get on. There are D_{kij} passengers at Stop i who want to go to Stop j. However, some passengers may not choose to get on a traditional bus when a rapid bus is also available. In addition, if Bus k is a rapid bus, it may not stop at Stop i or Stop j. Therefore, only U'_{kij} passengers want to get on the bus. However not all of them are able to board the bus because of the limit of rated passenger capacity; ultimately, only U_{kij} passengers are able to get on. The details used to calculate each variable are presented as follows:

$$D_{kij} = D_{k'ij} - U_{k'ij} + O_{ij} F_{ki} \quad (11)$$

$$R_{ki} = M - X_{k(i-1)} + \sum_{j=1}^{i-1} U_{kji} \quad (12)$$

$$U'_{kij} = D_{kij}(E_k S_i S_j + (1-E_k)(S_i S_j p + (1-S_i S_j))) \quad (13)$$

$$U_{kij} = \begin{cases} U'_{kij}, & 0 \leqslant \sum_{j=i+1}^{n} U'_{kij} \leqslant R_{ki}; \\ R_{ki} \times \dfrac{U'_{kij}}{\sum_{j=i+1}^{n} U'_{kij}}, & R_{ki} < \sum_{j=i+1}^{n} U'_{kij} \end{cases} \quad (14)$$

$$T_{ki} = (1 - E_k + S_i E_k)(a + b \sum_{j=i+1}^{n} U_{kij} + c \sum_{j=1}^{i-1} U_{kji}) \quad (15)$$

where p is the probability of passengers getting a traditional bus when a rapid bus is also available, M is the maximum number of passengers one bus can carry, R_{ki} is the number of remaining seats after passengers have got on and off, X_{ki} is the number of passengers Bus k carries when it leaves Stop i, and a is the extra equivalent time spent by buses speeding down at stops. In particular, if Bus k is the fisrt bus to arrive at Stop i, then $D_{kij} = O_{ij} F_{ki}$. And at the first stop, $R_{k1} = M$. Furthermore, $S_i = 1$ means that Stop i is a rapid bus stop; $S_i = 0$, in contrast, means that it is not.

(3) Departing from the stop

We focus on the number of passengers on Bus k after it leaves Stop i, X_{ki}, and it is expressed as

$$X_{ki} = X_{k(i-1)} + \sum_{j=i+1}^{n} U_{kij} - \sum_{j=1}^{i-1} U_{kji} \quad (16)$$

In particular, at the first stop, $X_{k1} = \sum_{j=2}^{n} U_{k1j}$.

4 Solution Methodology

As an NP-hard problem is involved, we propose two methods for the solution: HPSO, which is a hybrid method of traditional PSO; and GAPSO, which is a combination of GA and PSO.

4.1 HPSO

PSO is an evolutionary computation technique developed by Kennedy and Eberhart[11] in 1995. The method is based on research involving swarms, such as fish schools and flocks of birds. When a flock searches for food, the simplest way is to search in the neighborhood of the bird that is the closest to the food. The key of PSO, therefore, is to search the locations of particles within the problem space. Each particle has its own location and velocity, as well as a fitness value decided by the optimal function. The current best particle is followed and the behavior is simultaneously recorded.

The particles in our model consist of an integer array, $\{G_k\}$, and a Boolean array, $\{E_k\}$, which represent the headways and types of buses, respectively. We thus propose a hybrid algorithm that combines Integer PSO and Binary PSO, which are defined as follows.

4.1.1 Integer PSO

For normal PSO, the locations of particles are multi-dimensional real vectors. To begin with, we initialize a population (array) of particles with random positions and velocities. Knowing its personal best value so far and global best value so far, each particle uses Eqs. (17) and (18) to update its position and velocity. We use m-dimensional vectors to record Agent i's information, and thus $\boldsymbol{X}_i = (x_{i1}, x_{i2}, ..., x_{im})$ represents its location

and $\boldsymbol{V}_i = (v_{i1}, v_{i2}, ..., v_{im})$ represents its velocity. The updating functions are shown as follows:

$$x_{id}^{k+1} = \text{round}(x_{id}^k + v_{id}^k) \tag{17}$$

$$v_{id}^{k+1} = \chi(\omega v_{id}^k + c_1\rho_1(\text{pb}_{id} - x_{id}^k) + c_2\rho_2(\text{gb}_d - x_{id}^k)) \tag{18}$$

where v_{id}^k represents the d-th dimension of the velocity of Agent i in the k-th iteration; c_1 and c_2 are coefficients of velocity, which in early experiements are usually equal to 2; ρ_1 and ρ_2 are random numbers between 0 and 1; x_{id}^k is the d-th dimension of the location of Agent i in the k-th iteration; pb_{id} is the d-th dimension of the location of Agent i's personal best value; gb_d is the d-th dimension of the location of global best value; and round$(\cdot)$ is an integer function.

To improve the effect and speed of convergence, we add a constriction factor and an inertia weight factor. Clerc[12] indicated that the constriction factor, χ, that satisfies the equation $\chi = \dfrac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}$, $\varphi = c_1 + c_2 > 4$, might help to guarantee the convergence of the PSO algorithm. Typically, in Clerc's experiment, c_1 and c_2 were both set to 2.05 and the constriction factor, χ, was thus 0.729. In addition, ω, the inertia weight, can control the previous velocity's effect on the present velocity, and a larger ω can enhance the global searching ability and correspondingly reduce the local searching ability[13]. Experiments show that a value of ω between 0.8 and 1.2 performs better in convergence.

In our model, x_{id}^k represents the headway. Thus, in the restriction formula, $x_{id}^k \in [x_{\min}, x_{\max}]$, $x_{\min}$ and $x_{\max}$ respectively represent the minimum and maximum headways. Similarly, v_{id}^k is restricted in $[-v_{\max}, v_{\max}]$ to prevent particles from moving too far away from the searching space. In our model we hypothesize that $v_{\max} = x_{\max}$.

4.1.2 Binary PSO

As its name implies, Binary PSO consists of binary location vectors, $\boldsymbol{X}_i$, although velocity vectors, $\boldsymbol{V}_i$, are same with those in Integer PSO. When updating locations by velocities, the larger the value of v_{id}^k, the greater the possibility that x_{id}^{k+1} will be 1. This is therefore reminiscent of the Sigmoid Function:

$$\text{sig}(v_{id}^k) = \frac{1}{1 + \text{e}^{-v_{id}^k}} \tag{19}$$

Therefore the location updating function for Binary PSO is obtained as

$$x_{id}^{k+1} = \begin{cases} 1, & \rho_{id}^{k+1} < \text{sig}(v_{id}^k); \\ 0, & \rho_{id}^{k+1} \geqslant \text{sig}(v_{id}^k) \end{cases} \tag{20}$$

where ρ_{id}^{k+1} is a random number between 0 and 1. In addition, we restrict v_{id}^k in the area $[-4, 4]$ to avoid saturation of the Sigmoid function. Other parts of Binary PSO are exactly the same as Integer PSO.

4.1.3 Hybrid PSO

In HPSO, the location of every particle consists of two parts: an m-dimentional integer array and an m-dimentional binary array. The first part uses Integer PSO to update its location and velocity, while the second part uses Binary PSO. The limits of locations and velocities are also considered separately. The updating function is shown as follows, where all parameters are decided by the experiments.

If $d \leqslant m$, then

$$v_{id}^{k+1} = 0.729(0.8v_{id}^k + 2.05\rho_1(\text{pb}_{id} - x_{id}^k) + 2.05\rho_2(\text{gb}_d - x_{id}^k)) \tag{21}$$

$$x_{id}^{k+1} = \text{round}(x_{id}^k + v_{id}^k) \tag{22}$$

else

$$v_{id}^{k+1} = 1.2v_{id}^k + 2\rho_1(\text{pb}_{id} - x_{id}^k) + 2\rho_2(\text{gb}_d - x_{id}^k) \tag{23}$$

$$x_{id}^{k+1} = \begin{cases} 1, & \rho_{id}^{k+1} < \text{sig}(v_{id}^k); \\ 0, & \rho_{id}^{k+1} \geqslant \text{sig}(v_{id}^k) \end{cases} \tag{24}$$

4.2 GAPSO

HPSO is good at problem solving using a simple method, and it makes full use of the differences between integers and binaries. However, it is unable to utilize the characteristics inside integers or binaries. It is well known that PSO is a direction-oriented random searching algorithm and particles are always moving towards the particles with a large fitness value. For a normal particle to move closer, it is necessary to adjust its location vector so that it is similar to that of the best one. However, in practice this does not work for vehicle types, because it is not feasible to change the type of bus in this respect. The type of bus and its headway have a dramatic influence on later buses, but they have no influence on former ones. To some degree, a single crossover operator may fit better in such circumstances, and in this respect we propose a new kind of algorithm called GAPSO. The process for implementing GAPSO is shown in Algorithm 1.

4.2.1 Record list

GAPSO is a 2-layer searching algorithm that is heavily involved in reducing computational complexity. We therefore create a record list to reduce times of running the PSO algorithm. The record list consists of three

Algorithm 1 GAPSO

// Every agent a_i represents a schedule, $1 \leqslant i \leqslant n$
// N is the max number of iterations;
// n' is the scale of strongPSO's population, $n' \ll n$
Generate binary vector a_i.type randomly;
for each $j \in [1, N]$ **do**
  for each $i \in [1, n]$ **do**
    Check the RecordList; // It records agents ever appeared
    if a_i is recorded **then**
      Update a_i's value and interval by RecordList;
    else
      $[a_i.\text{value}, a_i.\text{interval}] \leftarrow \text{weakPSO}(a_i)$;
      Add a_i to RecordList;
    end if
  end for
  // Use evolution method to evolve a new generation
  $\{a_i\} \leftarrow \text{selection}(\{a_i\})$;
  $\{a_i\} \leftarrow \text{crossover}(\{a_i\})$;
  $\{a_i\} \leftarrow \text{mutation}(\{a_i\})$;
  if Condition of convergence meets **then**
    break;
  end if
end for
$a'_i \leftarrow$ top n' agents ranking by value, $1 \leqslant i \leqslant n'$;
// Use strongPSO to recalculate intervals
for each $i \in [1, 3]$ **do**
  $[a'_i.\text{value}, a'_i.\text{interval}] \leftarrow \text{strongPSO}(a_i)$;
end for
return a'_i with max value

parts: m binaries represent bus types, m integers represent bus headways, and a real number represents fitness value. The length of the list equals the times for all different combinations of bus types that have ever occurred, and this is also equal to the times that the weakPSO algorithm runs. It is therefore evident that the length is much smaller than the product of the population number and iteration times, which can accelerate the computing speed dramatically.

4.2.2 Evolution method

Selection, crossover, and mutation are three crucial evolution methods within GA[14]. We use them here to evolve the combination of bus types. It is easy to encode because bus types consist of 0 and 1.

Roulette Wheel Selection is the most common method employed in a diverse selection algorithm. It first calculates the relative fitness value of each individual $p_i = f_i / \sum f_i$, and then divides the roulette into n pieces, according to choosing probability, p_i. It is evident that when fitness value is larger, an individual is more likely to be chosen. This method performs well under most conditions, but is unable to work well when the mean of all fitness values is large while the deviation is small. Therefore, we define a substitute value $f'_i = f_i - \min\{f_i\} + 1$ and use f'_i to calculate choosing probabilities, which amplifies the diversity of individuals and makes the best of selection processing.

As demonstrated a single-point crossover fits well in our model. We therefore mix the genetic sequences in a random order and crossover the adjacent two individuals under a constant possibility at one random point.

Mutation is not only the main way to maintain species diversity in nature, but also the key to avoiding local convergence. A larger mutation rate is better for determining the global optimal point, but is worse in relation to the speed of convergence. In our experiments, we set crossover rate as 0.5 and mutation rate as 0.1.

4.2.3 WeakPSO and strongPSO

Algorithm 1 contains two functions, weakPSO(·) and strongPSO(·) (although the "weak" and "strong" here are relative and reflect differences in iteration times and particle amounts). At the beginning of the algorithm, our main purpose is to determine the best bus types, rather than to determine whether the bus headways are optimal; here we use weakPSO algorithm. In contrast, when we already have the top three type sequences, our objective is to search for the best headway sequence; here we use the strongPSO algorithm.

5 Simulation and Analysis

5.1 Parameters

The parameters in this paper are hypothesized as follows: total number of stops, n, is 9; total number of buses, m, is 10; weighting coefficient for number of passengers on-bus, A_1, is 1; weighting coefficient for ratio of passenger extra on-bus time, A_2, is 100; weighting coefficient for passenger at-stop time, A_3, is 2 min^{-1}; extra equivalent time cost by buses speeding down at stops, a, is 1 min; time spent in one passenger getting on, b, is 0.1 min; time spent in one passenger getting off, c, is 0.05 min; maximum number of passengers on bus, M, is 50; probability of passengers getting on a normal bus when rapid buses are also available, p, is 0.9.

In addition, the driving time between every two adjacent stops, L_i, and the distribution of passenger flow, O_{ij}, are shown in Tables 1 and 2. We assume that rapid buses have the same driving ruote and speed with

Table 1 Driving time. (min)

Stop	Driving time	Stop	Driving time
1-2	2	5-6	4
2-3	7	6-7	9
3-4	2	7-8	2
4-5	3	8-9	2

Table 2 Passenger OD matrix. (h^{-1})

Origin stop	Destination stop							
	2	3	4	5	6	7	8	9
1	7	5	13	49	5	93	28	56
2	–	3	3	4	33	6	7	26
3	–	–	1	27	25	4	19	3
4	–	–	–	7	4	7	18	15
5	–	–	–	–	2	87	15	88
6	–	–	–	–	–	6	18	8
7	–	–	–	–	–	–	5	95
8	–	–	–	–	–	–	–	4

traditional buses; that is to say, $L'_i = 0, i = 1, 2, ..., n$. We also set the minimum and maximum headways as 1 min and 16 min, respectively.

5.2 Stop design

According to the data in Table 2, we compile a line chart of the travel demand (Fig. 1). Using the chart, it can be inferred that Stops 1, 5, 7, and 9 have larger passenger flows than the others. We can therefore design these stops as rapid bus stops, as shown in Table 3.

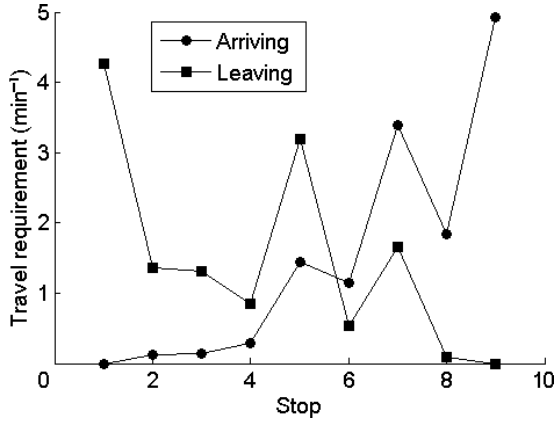

Fig. 1 Travel demand.

Table 3 Rapid bus stops.

Stop k	S_k	Stop k	S_k	Stop k	S_k
1	1	4	0	7	1
2	0	5	1	8	0
3	0	6	0	9	1

5.3 Model comparisons

Traditional models optimize either departure intervals or bus combinations. We compare these two models with our comprehensive optimization model, which optimizes both intervals and combinations simultaneously, based on the same assumptions.

5.3.1 Optimizing combination

Considering driver ease and vehicle management, conventional models use uniform departure frequencies. Under these circumstances, it is only necessary to run optimization algorithms at several fixed headways. Variables are defined the same as before: E_k representing types and G_k representing intervals (unit: min). We use Binary PSO to optimize E_k under the different conditions that array $\{G_k, k = 1, 2, ..., m\}$ is equal to $\{2, ..., 2\}$, $\{3, ..., 3\}$, $\{4, ..., 4\}$, $\{5, ..., 5\}$, and $\{7, ..., 7\}$, respectively. We run each one 10 times; the best results are recorded in Table 4. Here W is the objective value, while W_1, W_2, and W_3 represent the number of passengers one bus carries, the ratio of passenger extra on-bus time, and the passenger at-stop time, respectively.

The results show that as the departure interval increases, the rapid buses appear more frequently and that the combinations for rapid buses are somehow able to diminish the spatial unbalance of passenger flow. However, the influence cannot be completely eradicated, and therefore the full-load ratio, on-bus time, and at-stop time all increase. The best objective value of -42.31 is found when the departure interval equals 4 min.

5.3.2 Optimizing headway

Optimizing headway is one of the most common models used in practice. It is most usual to set bus combination manually according to prior experience, and then to run headway-optimizing algorithms to determine the most suitable schedule. We thus experiment using combinations of traditional buses, one rapid bus for every four traditional ones, one rapid bus for every two traditional ones, and then alternate rapid and traditional buses. Similarly, we run Integer PSO 10 times under each condition and select the best solutions (Table 5).

As a whole, optimizing headway alone delivers a superior performance to optimizing combination alone. To be more specific, passenger waiting time can be saved, while ensuring a relatively high full-load ratio. However, differing from the optimizing combination, an increase in the proportion of rapid buses does not

Table 4 Optimizing combination.

$\{G_k\}$ (min)	$\{E_k\}$	W	W_1	W_2	W_3 (min)
{2, 2, 2, 2, 2, 2, 2, 2, 2, 2}	{0, 1, 0, 0, 0, 0, 0, 0, 0, 0}	−45.41	17.15	0.51	5.75
{3, 3, 3, 3, 3, 3, 3, 3, 3, 3}	{0, 1, 0, 0, 0, 0, 0, 0, 0, 0}	−43.32	24.00	0.55	6.06
{4, 4, 4, 4, 4, 4, 4, 4, 4, 4}	{0, 1, 0, 0, 0, 0, 0, 0, 0, 0}	**–42.31**	30.86	0.60	6.65
{5, 5, 5, 5, 5, 5, 5, 5, 5, 5}	{0, 1, 0, 1, 0, 0, 0, 0, 0, 0}	−46.06	36.85	0.64	9.21
{7, 7, 7, 7, 7, 7, 7, 7, 7, 7}	{0, 0, 1, 0, 0, 0, 0, 0, 1, 1}	−51.59	38.40	0.65	12.26

Table 5 Optimizing headway.

$\{G_k\}$ (min)	$\{E_k\}$	W	W_1	W_2	W_3 (min)
{2, 3, 2, 4, 5, 5, 5, 5, 5, 5}	{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}	−39.61	31.78	0.60	5.67
{1, 1, 3, 4, 8, 1, 5, 5, 5, 9}	{0, 0, 0, 0, 1, 0, 0, 0, 0, 1}	−37.84	29.69	0.57	5.36
{1, 1, 4, 8, 1, 5, 9, 1, 5, 9}	{1, 0, 0, 1, 0, 0, 1, 0, 0, 1}	**–37.09**	31.02	0.57	5.55
{1, 1, 1, 6, 1, 9, 1, 8, 1, 9}	{0, 1, 0, 1, 0, 1, 0, 1, 0, 1}	−37.78	26.96	0.53	5.90

result in any obvious changes of those sub goals, because flexible headway erases the difference in the proportion of rapid buses. We can also infer from Table 5 that bus combinations play a part, as they all have better solutions than when only traditional buses are used. In summary, both flexible headway and bus combinations give good results.

5.3.3 Simultaneous optimization

In consideration of the two experiments conducted in Sections 5.3.1 and 5.3.2, the use of bus combination is vital, so is the departure interval. Clearly, optimizing both of them simultaneously gives a better result than the solo play. We therefore use HPSO to prove this assertion. Setting the iteration time as 100, and the number of particles as 100, we run HPSO 10 times and obtain the best solution: $\{E_k\} = \{0, 1, 0, 0, 0, 0, 0, 0, 0, 1\}$ and $\{G_k\} = \{1, 2, 1, 3, 4, 4, 4, 4, 4, 8\}$. The objective value, W, equals -35.54 with $W_1 = 25.15$, $W_2 = 0.51$, and $W_3 = 4.70$ min.

Figure 2 makes a comparison among the best solutions using the above methods: optimizing combinations with uniform headway, optimizing departure intervals without rapid buses, optimizing departure intervals within rapid buses, and optimizing combinations and intervals simultaneously. The results show that the uniform-headway policy involves a lot of time waiting at stops. The solutions of the two headway-optimization methods are similar, but the one with the rapid bus is slightly better. Simultaneous optimization contributes to saving passenger waiting time at the cost of a full-load ratio. In general, it is evident that the simultaneous optimization model performs best on an objective value.

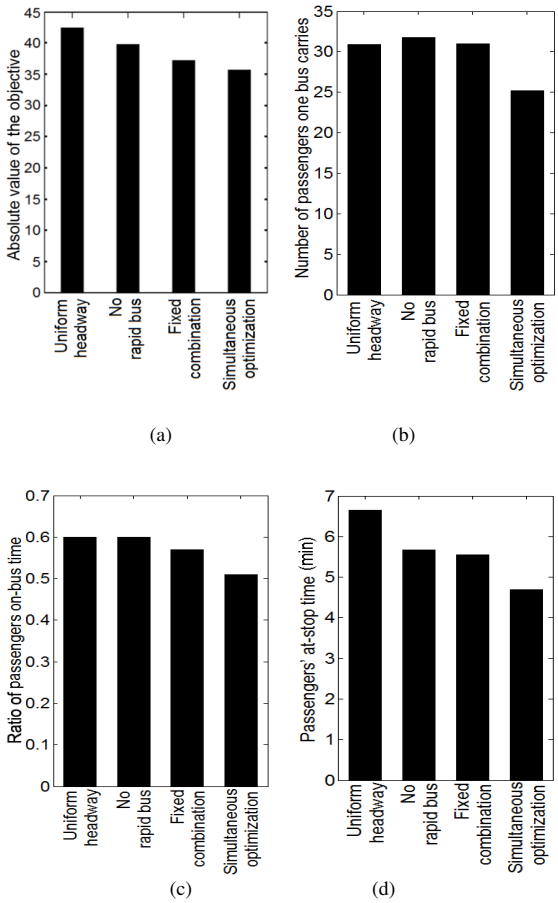

Fig. 2 Results of different scheduling methods: (a) Absolute value of the objective; (b) Number of passengers one bus carries; (c) Passenger on-bus time coefficient; and (d) Passenger at-stop time.

5.4 Algorithm comparison

In this paper, we propose two intelligent algorithms to solve our model. To prove the superiority and high

efficiency over traditional algorithms, a comparison is made with GA for computing speed, solution quality, and stability; some of the default parameters are shown in Table 6.

5.4.1 Solution quality and stability

It is understood that intelligent algorithms regard local optimal solutions as the approximate global optimal solution. However, it is of great importance to evaluate the stability of algorithms. Experiments are performed using GA, HPSO, and GAPSO 10 times, respectively, and the objective value distributions are compared in Fig. 3.

It can be categorically concluded that GAPSO has a much better solution quality and stability than the other two. The solution quality of HPSO on average is better than that of GA, while its stability is close to that of GA's, except several bad points. Both GA and HPSO wave so sharply that it is necessary to run them several times before finding a good solution in practice.

5.4.2 Computing speed and stability

Solution quality and computing speed are two opposite concepts, and improvements of one needs to be made at the expense of the other. According to Fig. 3, it is evident that GAPSO has the best value and that GA has the worst. Figure 4 represents computing time, and based on this it is evident that GA and HPSO have similar computing time, which are both much smaller than that of GAPSO. In addition, GA has the smallest computing time variance out of the three algorithms.

Table 6 Default parameters.

Algorithm	Iteration times	Population size
GA	200	50
HPSO	100	100
GAPSO	20 (GA), 30 (weakPSO), 100 (strongPSO)	50 (GA), 20 (weakPSO), 100 (strongPSO)

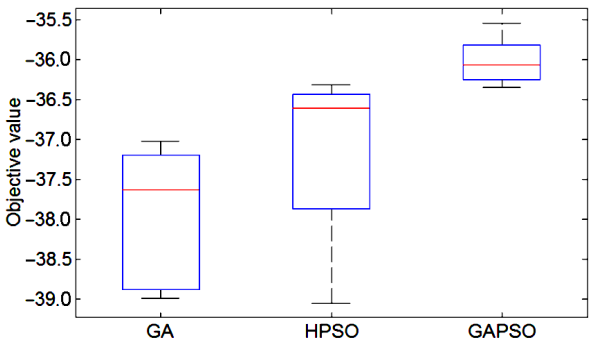

Fig. 3 Solution quality comparison.

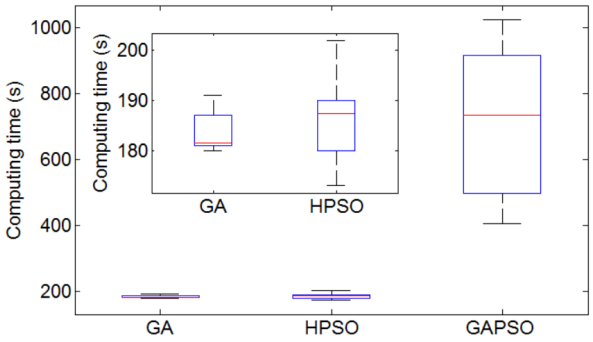

Fig. 4 Computing time comparison.

In conclusion, HPSO has a small computing time and GAPSO has the best searching ability and stability. They can thus be applied on different occasions: HPSO for obtaining solutions within a short time frame, and GAPSO for obtaining high quality solutions. Although the classical GA also has a small computing time, it has an inferior ability when finding the optimal value.

5.5 Sensitivity analysis

5.5.1 Weighting coefficient

Our scheduling model is a typical multi-objective optimization problem, and we transform it into a single-objective one by assigning different weight coefficients to each sub-goal and making a linear weighted summation of all sub-goals. To discuss their influence, we alter the three weight coefficients one by one and run HPSO; the results are recorded in Tables 7–9.

When A_1 increases, it can be seen that all three sub-goals increase. When A_2 increases, W_1 and W_2 decrease while W_3 increases. With an increase in A_3, all the sub-goals decrease. Out of all these, the influence of A_1 has the highest weight, and that of A_2 the lowest. It is thus necessary to carefully analyze and determine

Table 7 Weighting coefficient of number of passengers on one bus with A_2=100 and A_3=2 min^{-1}.

A_1	W_1	W_2	W_3 (min)	A_1	W_1	W_2	W_3 (min)
0	14.84	0.48	5.21	5	42.91	0.68	14.78
1	26.91	0.53	5.32	10	47.53	0.74	27.74
2	35.98	0.64	6.33	50	49.99	0.72	57.30

Table 8 Weighting coefficient of ratio of extra on-bus time with A_1=1 and A_3=2 min^{-1}.

A_2	W_1	W_2	W_3 (min)	A_2	W_1	W_2	W_3 (min)
0	36.90	0.65	5.89	100	26.91	0.53	5.32
10	35.98	0.65	5.89	500	12.52	0.37	9.38
50	35.63	0.62	6.08	1000	12.92	0.37	9.76

Table 9 Weighting coefficient of at-stop time with A_1=1 and A_2=100.

A_3 (min^{-1})	W_1	W_2	W_3 (min)	A_3 (min^{-1})	W_1	W_2	W_3 (min)
0	43.24	0.62	39.08	1	30.07	0.55	7.11
0.1	37.96	0.57	27.93	10	21.01	0.51	5.10
0.5	33.51	0.57	8.13	100	23.61	0.53	4.60

a reasonable coefficient for the sub-goals, to obtain the best objective function for traffic management.

5.5.2 Willing proportion

In our model, we estimate the probability of passengers getting on a traditional bus when rapid buses are also available. The influence of this parameter is now discussed. We choose the best solution out of the above analysis, setting $\{E_k\} = \{0, 1, 0, 0, 0, 0, 0, 0, 0, 1\}$ and $\{G_k\} = \{1, 2, 1, 3, 4, 4, 4, 4, 4, 8\}$. Then we increase the probability p from 0 to 1, the changes of three sub-goals are shown in Fig. 5. The at-stop time is continually reduced with an increase in p, which indicates that, on average, using this scheduling strategy will not save time if passengers give up getting on a traditional bus and wait for a rapid bus. When p is relatively small, it also reduces the average number of passengers that one bus carries because it is difficult for traditional buses to attract passengers at rapid bus

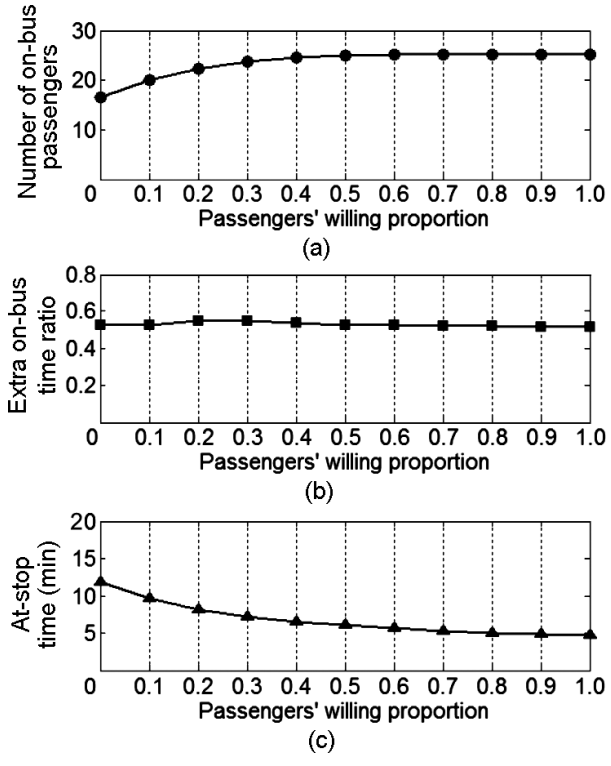

Fig. 5 Sensitivity of objective function to passengers' willing proportion.

stops. However, p has little influence on average time passengers spend on-bus. To conclude, the estimation of passengers' willing proportion influences the final objective function value dramatically and thus matters a lot to our scheduling strategy.

6 Conclusion

In this paper, we propose a bus schedule model that considers the phenomenon of vehicle over-taking, the limit of the vehicle's capacity, and the uncertainty of passenger choice. By improving the combination of bus types and departure intervals, we attempt to reduce passenger waiting time and engage the interest of bus agencies. It is determined that HPSO and GAPSO are two efficient algorithms that can be used with our model; a comparison shows that HPSO has a superior computing time and GAPSO has a better searching ability.

Acknowledgment

This work was supported by the National Key Basic Research and Development Program in China (No. 2016YFB0100906), the National Key Technology Research and Development Program (No. 2014BAG03B01), the National Natural Science Foundation of China (Nos. 61273238 and 61673232), and Beijing Manicipal Science and Technology Program (No. D15110900280000).

References

[1] X. J. Eberlein, N. H. M. Wilson, C. Barnhart, and D. Bernstein, The real-time deadheading problem in transit operations control, *Transportation Research Part B: Methodological*, vol. 32, no. 2, pp. 77–100, 1998.

[2] X. Xu, D. Xu, and H. Ma, Research on the real-time deadheading problem in transit operations control, *Transportation and Computer*, vol. 21, no. 4, pp. 19–21, 2003.

[3] C. F. Daganzo and J. Pilachowski, Reducing bunching with bus-to-bus cooperation, *Transportation Research Part B: Methodological*, vol. 45, no. 1, pp. 267–277, 2011.

[4] S. J. Berrebi, K. E. Watkins, and J. A. Laval, A realtime bus dispatching policy to minimize passenger wait on a high frequency route, *Transportation Research Part B: Methodological*, vol. 81, pp. 377–389, 2015.

[5] D. Xu and Y. Pei, Express bus scheduling model and application, (in Chinese), *Journal of Harbin Institute of Technology*, vol. 40, no. 4, pp. 580–584, 2008.

[6] Z. Bai, R. Song, G. He, and J. Lin, Simulation of tabu simulated annealing algorithm for optimizing BRT line combination frequency, (in Chinese), *Application Research of Computers*, vol. 25, no. 2, pp. 355–358, 2008.

[7] C. Sun, W. Zhou, and Y. Wang, Scheduling combination and headway optimization of bus rapid transit, (in

Chinese), *Journal of Transportation Systems Engineering and Information Technology*, vol. 8, no. 5, pp. 61–67, 2008.
[8] X. Hao, W. Jin, and Y. Yang, Scheduling combination optimization research for bus lane line, *TELKOMNIKA Indonesian Journal of Electrical Engineering and Computer Science*, vol. 12, no. 1, pp. 809–817, 2014.
[9] J. Hu, J. Song, Z. Yang, and Y. Zhang, Study on determination method of real-time transit vehicle dispatching form, (in Chinese), *Journal of Highway and Transportation Research and Development*, vol. 20, no. 6, pp. 113–117, 2003.
[10] F. Zhang, X. Cao, and D. Yang, Intelligent scheduling of public traffic vehicles based on a hybrid genetic algorithm, *Tsinghua Science and Technology*, vol. 13, no. 5, pp. 625–631, 2008.
[11] J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proc. IEEE International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942–1948.
[12] M. Clerc, The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization, in *Proc. the Congress on Evolutionary Computation*, Piscataway, NJ, USA, 1999, pp. 1951–1957.
[13] K. E. Parsopoulos, V. P. Plagianakos, G. D. Magoulas, and M. N. Vrahatis, Improving particle swarm optimizer by function stretching, in *Advances in Convex Analysis and Global Optimization*. Kluwer Academic Publishers, 2001, pp. 445–457.
[14] D. E. Goldberg, *Genetic Algorithm In search, Optimization and Machine Learning*, Addison Wesley, 1989.

Jiaoyang Li is currently an undergraduate student in Department of Automation, Tsinghua University. Her recent research interests focus on planning and scheduling in intelligent transportation system. Her research fields also cover intelligent decision making in multi-agent system.

Yi Zhang received the BEng degree in 1986 and MEng degree in 1988 from Tsinghua University in China, and earned his PhD degree in 1995 from the University of Strathclyde in UK. He is a professor in control science and engineering at Tsinghua University with his current research interests focusing on intelligent transportation systems. Active research areas include intelligent vehicle-infrastructure cooperative systems, analysis of urban transportation systems, urban road network management, traffic data fusion and dissemination, and urban traffic control and management. His research fields also cover the advanced control theory and applications, advanced detection and measurement, and systems engineering.

Jianming Hu is currently an associate professor with Department of Automation, Tsinghua University. He got the BE degree in 1995 and ME degree in 1998 from Harbin University of Science and Technology, and the PhD degree in 2001 from Jilin University. He worked in Chinese University of Hong Kong from 2004 to 2005 as a research assistant and visited PATH, University of California at Berkeley for one year as a visiting scholar. He has presided and participated over 20 research projects granted from MOST, NSFC, and other large companies with over 30 journal papers and over 80 conference papers. Based on the research achievements, he obtained a Science and Technology Improvement Award of ITS Association of China in 2012, the First Class Award of Excellent Teaching Skill Contest in Beijing and many awards from Tsinghua University. Dr. Hu's recent research interests include networked traffic flow, large scale traffic information processing, intelligent vehicle infrastructure cooperation systems, and urban traffic signal control, etc.