Hindawi
Discrete Dynamics in Nature and Society
Volume 2022, Article ID 2086717, 13 pages
https://doi.org/10.1155/2022/2086717

Research Article

Comparison of Two Algorithms for Multiline Bus Dynamic Dispatching

Yingxin Liu,[1,2] Xinggang Luo,[1] Shengping Cheng,[1] Yang Yu,[1] Jiafu Tang,[1] and Xuanzhu Shang[3]

[1]*College of Information Science and Engineering, Northeastern University, Shenyang, China*
[2]*Graduate School, Shenyang University, Shenyang, China*
[3]*School of Mathematics, University of Alberta, Edmonton, Canada*

Correspondence should be addressed to Xinggang Luo; xgluo@mail.neu.edu.cn

Received 26 October 2021; Revised 26 December 2021; Accepted 7 January 2022; Published 29 January 2022

Academic Editor: Tingsong Wang

Dynamic bus scheduling refers to adjusting the departure time according to the latest time-varying information or adjusting bus speed in the process of operation. These control strategies can prevent bus bunching and alleviate traffic pressure. The paper studies the multiline bus dynamic scheduling with consideration of departure time and speed meanwhile. The hyperheuristic algorithm is proposed, and low-level heuristics (LLH) operators are designed. The simulation experiment is performed for the passenger flow distribution of different strengths and types of different scenarios. By comparing the experimental results of genetic algorithm (GA) and hyperheuristic algorithm in solving different scenarios, the results show that in smooth, increasing, decreasing, and multiconvex passenger flow mode, the performance of the hyperheuristic algorithm is higher than that of GA. The promotion rate reaches 18~28%, and especially the average value of the hyperheuristic algorithm designed under multiconvex passenger flow is up to 28.62%, significantly reducing passengers' waiting time. By comparing the stability of the three passenger flow modes, the results illustrate that the stability of the hyperheuristic algorithm is lower than that of GA. For the smooth passenger flow mode, the stability of medium and lower density of GA is higher than that of the hyperheuristic algorithm. In comparison, the high-density stability of the hyperheuristic algorithm is better than that of GA.

1. Introduction

The number of urban private vehicles is rising year by year in China. It is reported that the number of vehicles reached 372 million in 2020, including 281 million cars [1], an increase of 1.14 million more than 2019, and it continued to show a rapid growth trend. In 2020, there were more than 1 million cars in 70 cities, and more than 2 million in 31 cities [1].

Traffic congestion has become a common phenomenon in cities. To alleviate traffic congestion in big cities has been an urgent problem to be solved, while developing public transportation is an excellent solution. Public transport refers to all modes open to the public and is part of urban infrastructure. For buses, the advantages of large capacity, energy-saving and environmental protection, and low per capita oil consumption make an excellent solution to reduce the urban traffic conditions. To make traffic smooth and reduce pollutants emissions, the government takes active measures to encourage citizens to take public transportation as far as possible, which is in line with China's green and sustainable development strategy.

Attracting more passengers to choose bus travel is an enduring hot topic in the field of public transportation. Dynamic bus dispatching is one of the positive and effective measures to solve the problem. It is the core part of public transportation, which refers to a reasonable scheduling scheme under fixed driving routes and bus facilities to make the bus system operate more rational, efficient, and intelligent.

This study is based on a multiline dynamic bus dispatching model with departure time and speed in [2], which is to make the passengers' waiting time achieve minimum by determining the bus departure time at the first station and average speed between stations. The study designed a

hyperheuristic algorithm, through simulation experiments conducted for passenger flow distribution of different strengths and types in different scenarios, comparing the experimental results with a genetic algorithm (GA).

The remainder of this study is organized as follows. Section 2 is literature review. The multiline model for achieving passengers' waiting time minimum in [2] is quoted in Section 3. Section 4 presents the hyperheuristic algorithm. The experiments and results analysis are described in Section 5. Section 6 is the conclusions.

2. Literature Review

Multiline bus dynamic scheduling is also known as regional scheduling. There are generally two methods to solve the transfer coordination problem of regional scheduling in the research, respectively, based on the best adjustment value of the scheduling scheme and achieving the bus number of simultaneous arrivals maximum.

Furth and Wilson [3] built a mathematical optimization model, which includes the passenger rate, bus rating, and total operating cost; the departure frequency as the decision variate; and the minimum passenger waiting time and maximum social benefit as a target. Ceder [4] studied the departure intervals for buses and proposed four methods to solve the departure intervals. Two of them are solved based on the passenger arrival volume and the rated passenger-carrying rate of a station during the peak time. The other two apply the relationship between the turnover rate and the bus capacity at a certain period of the bus line. Ceder and Stern [5] developed an integer programming model for interval vehicles timetable generation problem and developed a heuristic algorithm. Ceder et al. conducted a detailed study of departure intervals based on different types and different ways and considered considering traffic accidents. They established a mixed-integer planning model for targeting the maximum number of buses simultaneously at a transfer station and developed suitable GA as a solution [6–8].

Dessouky et al. [9] considered transfer waiting time of passengers while studying holding scheduling, making the shortest by holding, and developing eight reasonable holding strategies. A simulation model was established for verification. They proposed the calculation method of vehicle optimal holding time based on the original research, added the prediction method of bus arrival time, and compared seven holding methods through simulation [10, 11]. Chowdhury and Chien [12] established the shortest total time model, including the passenger transfer delay time, the delay time of transfer vehicle, and waiting time of vehicle departure. They optimized the scheduling by calculating the remaining time and the vehicle departure time. Synchronous transfer was studied by Liebchen and Möhring [13], in which the number of buses required in the bus system, the number of buses between stations, and passengers' transfer waiting times were explored. A mixed-integer planning model was constructed and solved using CPLEX.

Fleurent et al. [14] constructed a network flow model when studying the timetable scheduling problem of regional vehicles, introduced three transfer time types of maximum, minimal, and optimal, and developed a Lagrangian relaxation and heuristic algorithm for solution optimization. Li and Li [15] studied the real-time dispatching optimization method in the bus hub, established the real-time dispatching optimization model in the bus hub according to transfer efficiency, presented the optimization model for minimizing the total cost, and developed the random perturbation approximation algorithm to optimize the solution.

Ulusoy et al. [16] studied the passenger flow model and considered the uneven distribution of passenger arrival rate in space and time. They constructed an extensive station express combination scheduling and interval vehicle, the minimum total vehicle as the target, and departure interval as the decision variable. Sun [17] conducted a detailed and comprehensive study on the dynamic optimization of urban multiline bus transmission under the Internet of Things environment, considered the impact of various factors such as vehicle capacity limit, multimodel, bus company operating expenses on the model, and constructed mathematical models and heuristic algorithm to solve the optimization respectively. Song and Zhang [18] analyzed the topological characteristics of the public transport networks, based on the graph theory method, clarified the significance of the shortest path in the bus network, studied the minimum number of transfers as the optimization goal, and designed and implemented a bus query system with the minimum transfer algorithm.

Chen et al. [19] designed the bus network with a continuous approximate modeling framework and proposed an optimization model for two different planning scenarios. The model optimized the bus network by minimizing the total cost of bus companies and passengers under the premise of considering the interval bus operation strategy. Song et al. [20] have proposed a two-layer planning model to artificially solve the scheduling problem of bus lines for multiple operators in overlapping intervals. The upper model is the government agency. By dispatching the bus route allocation scheme to minimize the total running time, as well as by dispatching departure intervals for buses to maximize the benefits of the operators, an NSGA-II algorithm is developed for the solution.

Bourbonnais et al. [21] performed the genetic algorithm to solve the problems of traffic network design and frequencies setting by the data of three medium-sized cities in Quebec; the efficiency increased between 10 and 20% compared to the existing model, and the parameters and fleet sizes of which are the same. Liu et al. [2] established the multiline model considering the bus departure interval and the bus speed adjustment, and the joint optimization algorithm was used.

The proposed bus dynamic scheduling model in the previous study mainly considered the departure time and was mainly solved with the genetic algorithm. The paper considers the departure time and speed adjustment meanwhile, puts forward hyperheuristic algorithm, and designs low heuristic operator. Under different scenarios, different simulation experiments are designed, and the results are compared with GA and hyperheuristic algorithm, which provides a reference for bus dynamic scheduling model establishment.

3. Multiline Model for Minimizing Passengers' Waiting Time

3.1. Problem Description. The CEA Model Treats Multiple Transport Lines as a Special Single Line. Because of the Following Assumption, the Departure of the Bus Is Considered as Single-Line Problem. Take any one of the studies on bus lines as an example, as shown in Figure 1. The travelling direction of bus is where the arrow indicates. The total numbers of bus stations are N_p^s on the running line; meanwhile, the number of the travelling buses is N_p^D. The buses waiting to start from departure station are denoted by from $N_p^D + 1$ to $N_p^D + N_p^F$; the total number is N_p^F. During the planning cycle of buses, the running bus collects the road condition and passenger information to optimize start time and interstation speed of next bus.

3.2. Assumptions. To establish CAE optimization model, the following assumptions are made, which were previously made in many papers [22–25]:

(1) All research buses are of the same type.

(2) There are no road accidents.

(3) The time for passengers on and off are equal.

(4) All buses must stop when they arrive the station, and the inbound time and outbound time of bus are the same.

(5) Any travelling bus cannot overtake.

(6) The start time of the last train must remain unchanged.

(7) Passenger-flow and bus speed could be detected. The function relation between passenger flow and time of each bus station can be calculated by historical passenger flow information.

3.3. Parameters. The parameters in this model were defined as follows:

i: bus code

j: station code

p: line code

$H^{\min}$: minimum headway

$H^{\max}$: maximum headway

N_P^D: the number of buses travelling on line p

N_P^F: the number of buses to be planned of line p

N_P^s: the number of bus stations on line p

$V^{\min}$: minimum running speed on line p

$V^{\max}$: maximum running speed on line p

T_{pij}: leaving time from stop j of bus i on line p

V_{pij}: average running speed of bus i on line p between adjacent stations j and j–1

$T_p^{\text{avg\_}w}$: average waiting time of line p for passengers failing to catch the last bus [2]

N_{pij}^{left}: the number of passengers left by bus i on line p as it gets to station j [2]

N_{pqij}: the number of transfer passengers on bus i on line p who transfer to line q at transfer station j; the value is 0 when station j online p is not a transfer station [2]

T_{pqij}^{wait}: waiting time of transferring from line p to line q

T^{total}: all passengers' waiting time

T^f: the total waiting time of waiting for the first bus [2]

T^l: the total waiting time of delayed passengers waiting for subsequent buses [2]

T^{tr}: the waiting time for transfer passengers

$|S_{pq}|^+ S_{pq}$ is 1 if there is at least one element in the set $|S_{pq}|^+$ [2]

3.4. Model Construction. The objective function of the research is to achieve the all passengers' waiting time minimum. The expression is as follows:

$$\text{Min} T^{\text{total}} = T^f + T^l + T^{tr}, \tag{1}$$

s.t.

$$\begin{aligned} H_{\min} \le T_{pij} - T_{pi(j-1)} \le H_{\max} \\ p = 1, 2, \ldots, N; \\ i = 1, 2, \ldots, N_p^D + N_p^F; j \\ = 2, \ldots, N_p^S, \end{aligned} \tag{2}$$

$$\begin{aligned} V_{\min} \le V_{pij} \le V_{\max} \\ p = 1, 2, \ldots, N; \\ i = 1, 2, \ldots, N_p^D + N_p^F; j \\ = 1, \ldots, N_p^S - 1, \end{aligned} \tag{3}$$

$$\begin{aligned} T_{pij} - T_{p(i-1)j} \ge 0 \\ p = 1, 2, \ldots, N; \\ i = 1, 2, \ldots, N_p^D + N_p^F; j \\ = 1, \ldots, N_p^S - 1. \end{aligned} \tag{4}$$

T^{total} of (1) refers to all passenger waiting time, consisting of T^f, T^l, and T^{tr}. T^f means the time of the passengers waiting the first bus.

$$T^f = \sum_{p=1}^{N} \sum_{i=2}^{N_p^D + N_p^F} \sum_{j=1}^{N_p^S} \int_{T_{p(i-1)j}}^{T_{pij}} \left(T_{pij} - t\right) f_{pj}(t) \mathrm{d}t. \tag{5}$$

T^l represents the time of stranded passengers waiting the following bus with vacant capacity, which is expressed as follows:

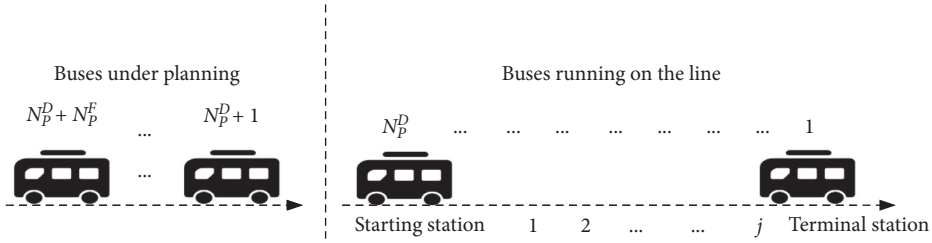

Figure 1: Bus departure operation diagram of a single line.

$$T^{l} = \sum_{p=1}^{N} \sum_{i=2}^{N_p^D+N_p^F} \sum_{j=1}^{N_p^S} N_{pij}^{\text{left}} \left[T_{pij} - T_{p(i-1)j}\right] + \sum_{p=1}^{N} \sum_{j=1}^{N_p^S} T_p^{\text{avg\_}w} N_{p\left(N_p^D+N_p^F\right)j}^{\text{left}}. \tag{6}$$

T^{tr} means all passengers waiting bus time in the transfer station, which is expressed as follows:

$$T^{tr} = \sum_{p=1}^{N} \sum_{q=1}^{N} \sum_{i=1}^{N_p^D+N_p^F} \sum_{j \in S_{pq}} \left|S_{pq}\right|^{+} N_{pqij} T_{pqij}^{\text{wait}}. \tag{7}$$

The value of the set $|S_{pq}|^{+}$ is 1 when it is not empty; when there are no passengers to transfer bus line from p to q, the value of which is 0. In addition, equations (2) and (3) indicate the departure interval and speed of the all buses must meet the maximum and minimum constraints, respectively, and equation (5) must meet constraint (4).

4. Hyperheuristic Algorithm

4.1. The Fitness Calculation Rules. The hyperheuristic algorithm includes High-Level Strategy (HLS) and Low-Level Heuristics (LLH) algorithm libraries, where LLH operator to be designed is the focus of the algorithm. In this study, HLS first initializes high-level individuals, and the low-level problem is described in a chromosome form. A series of LLH operators are selected through GA, and the selected LLH operators act on the specific problem domain to optimize the problem by using information such as the problem description and evaluation function provided by the low level.

The fitness calculation rules for each high-level individual are as follows.

Step 1. To apply the heuristic operator, corresponding to the first locus of this high-level individual, on the initial low-level population S1, yields the evolved population S2 and the current optimal individuals.

Step 2. To apply the heuristic operator corresponding to the second locus on the evolved population S2, the current optimal individual can be updated if the individual is excellent. So, the corresponding heuristic operator is on the corresponding locus of the last heuristic operator; the optimal low-level individual is updated constantly. Finally, the optimal individual corresponding to the high-level individual fitness is found. Then, the high-level population is selected.

High-level individuals are chromosomes composed of several loci, each of which being a number, each representing a different LLH operator. After crossover, variation operation, top populations produce several individuals representing different combinations of heuristic operators, after which each individual in the chromosome pool is evaluated. Later, the next generation of individuals is selected to perform crossover, variation, and selection until the termination conditions are met.

4.2. Design of High-Level Population Coding Rules and Population Initialization. High-level individuals are integer-encoded, and the encoded chromosome schematic is shown in Figure 2 in [2].

A high-level chromosome is a number of codes with seven loci. Each locus (mn-i) is an integer between intervals [1, 7], and each number decoded represents a specific low-level heuristic operator. For example, one can be decoded as an LLH1 operator, and two can be decoded as an LLH2 operator, and so on. In particular, the values of each locus can be repeated or missing.

4.2.1. High-Level Population Crossover, Variation, and Selection Operator. The high-level strategy used GA. The championship method is used as the selection operator for the population individual. The population individuals are some discrete number combinations, so the single-point exchange method is adopted as a cross operator, and the variant operator also adopts the single-point variation method. Because individual loci represent different low-layer heuristic operator operations, no nonfeasible solutions exist. The algorithmic stop criterion is that the population has evolved a specific algebra.

4.2.2. Selection Operator. Selection mechanisms work by copying individuals with high fitness values to the next generation to preserve the trait good individuals and improve the overall average fitness values of the population. The selection operator determines the rate of convergence of GA. The binary championship selection method is used as the selection operator.

In championship selection, there are s individuals from the population, which are first randomly sampled, but they can be put back after sampling. Then, the optimal individual

mn1	mn2	mn3	mn4	mn5	mn6	mn7

Figure 2: Schematic diagram of high-level chromosomes encoding.

is selected to go to the next generation. If an individual's fitness is the better than other *s-1* competitors' value, the operation will be repeated until a specific number of surviving individuals are obtained. And the worst individual cannot survive, and the best individual wins in all the tournaments. The pressure of individual survival can be changed by changing the length of the tournament. For *s* with larger values, the chance of the weak being selected is smaller, common with binary championships and ternary tournaments.

4.2.3. Crossover Operator. The study employs a uniform crossing method as a crossover operator, and the schematic diagram of crossing is shown in Figure 3.

Two paternal individuals P1 and P2 that required cross manipulation are selected from the population to randomly generate a 0–1 mask with the same chromosome length, corresponding one to one to each of the two parents, where the two paternal loci with the corresponding mask one were exchanged. Two new subgenerations of O1 and O2 are generated by cross-operation and added to the progeny population, waiting for subsequent operation.

4.2.4. Variation Operator. Single-point random variants were used, with a schematic representation of the variant steps shown in Figure 4.

Picking the parent individual P1 requiring the variant operation from the population also randomly generates a 0–1 mask consistent with the chromosome length. The locus corresponding to the one mask is the part of the variant operation to be performed. The variation operator taken in this study randomly replaces the locus to be varied with another value that meets the requirements.

4.2.5. Adaptive Improvement. The hyperheuristic algorithm designed in this study enables indirect problem domain optimization through the control of low-level heuristic operators. Sometimes, high-level individuals do not need to go through all the operators to find the optimal combination representing the current individual fitness. For example, the high-level individual [1, 1–5, 7] optimizes according to the fitness calculation rules introduced in Section 4.1. When the optimal individual found by the heuristic operators is located at the low level, that is, the optimal individual after a combination of 2-3-5-1, the later 1-7-4 heuristic operators can be considered invalid optimization. In contrast, 2-3-5-1 can be viewed as a better combination of heuristic operators. The study made an adaptive improvement on this part, assuming that the high-level individuals illustrated above are optimal individuals of the current generation. A combination starting with 2-3-5-1 operators will have a greater probability of being chosen when the following population individual is to be saved.

4.3. Low-Level Heuristic Operator Design. The LLH operator of the hyperheuristic algorithm is directly applied to the problem domain solution space for the domain optimization of the problem; that is, the selected characteristic LLH operators are evolutionarily updated for individuals in the problem domain. In order to improve algorithm's efficiency, the study presents low-level heuristic operators set as different heuristic sequences. Two variables to be decided are departure moment and vehicle speed, using the problem domain individuals defined in GA of [2] as a solution representation. The specific schematic diagram is described in Figure 5.

H_{111} indicates the departure interval planned between the first bus and the previous bus on line 1 and departure intervals to be planned between each bus and the previous bus. V_{111} to $V_{1N_1^F N_1^S}$ refer to the travel speed of the first bus to the N_P^F bus between all stations on line 1. Other lines were designed in the same design pattern as the line. For example, in line 1, the first previous N_P^F locus states the departure interval between the bus to be decided and the previous bus at the first stop [2]. Subsequent N_1^S locus is the average speed of the decided first bus between stations, and then the N_1^S level is the second bus speed between stations so until the N_1^F bus travels between stations on the line.

Seven different LLH operators are designed according to the problem characteristics:

(1) LLH1 means random cross. It means randomly exchanging two different loci in one part for all parts of the problem. As shown in Figure 4, the departure interval of one bus line is a part, the average driving speed of a bus at each station is another part, and the driving speed of the second bus at each station is another part. The operator is adopted for each different part. The other parts of the operators below are divided by the method.

(2) LLH2 means insert-delete. For each part, a position is randomly selected. A reasonable random gene is inserted ahead of that position, and another site in the same section is randomly chosen for deletion.

(3) LLH3 means insert-backward. Two positions are selected in each part to move the latter locus to the preceding gene, and the preceding original gene and the gene between the two loci are moved the whole back one.

(4) LLH4 means adjacent intersections. A locus is randomly selected and swapped with the previous one or the latter one.

(5) LLH5 operates in inverse sequence, randomly selecting consecutive subsequences from 3 to 4 bits and arranging this sequence in reverse sequence.

(6) LLH6 interchanges. Select two individuals, for each section, and exchange the same positional genes for these two individuals.

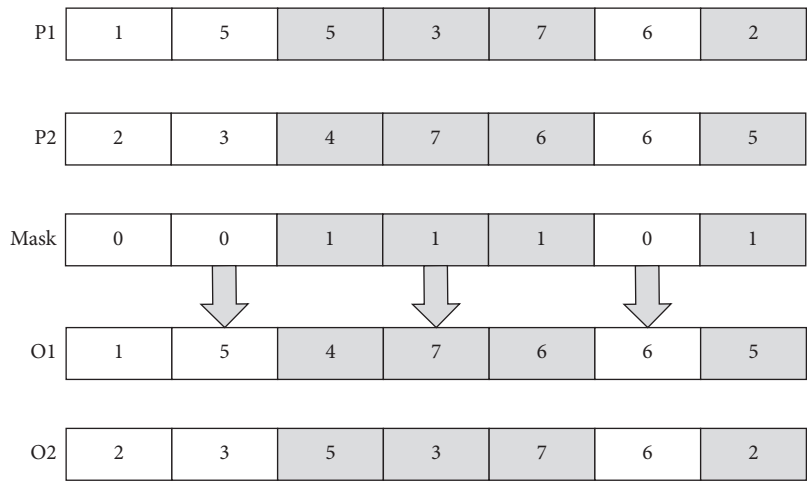

Figure 3: Schematic diagram of high-level chromosome crossover.

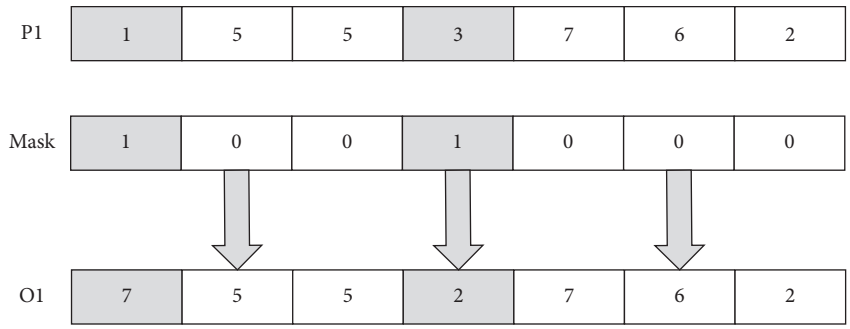

Figure 4: Schematic diagram of high-level chromosome mutations.

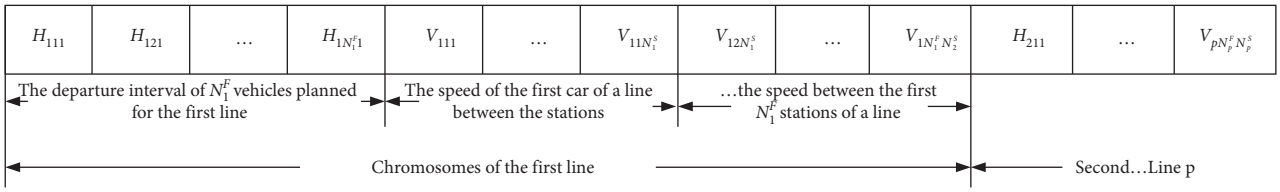

Figure 5: Schematic diagram of low-level heuristics chromosomes.

(7) LLH7 selects a random position, turns it into a random number between the maximum-minimum departure interval or vehicle speed, and finds another random position in the same part to increase or decrease the same value.

5. Numerical Experiments

The case selected is the same as that of [2]. The data of six bus lines and passenger flow rate in four scenarios in [2] are also used. Experimental design and parameter design are the same as in [2].

According to the bus company, the departure interval of buses is from 5 min to 20 min, the average bus speed between stations is between 10 km/h and 30 km/h, the average time per passenger on and off is 5 s, and buffering time of vehicle deceleration and accelerate is about 30 s. Eight vehicles are planned for each line. The departure interval and the interstation speed of these buses are optimized.

The parameter values are as follows: the high-level control strategy uses GA for scheduling in [2], the crossover rate equals 0.8, the variation rate equals 0.1, and the number of low-layer problem domain populations *low-pop-scale* and individuals per generation of the population *pop scale* are both 15.

The passenger flow situation on the lines during different periods of the day is different. After counting the passenger flow data, the passenger flow distribution has a certain rule, and there are four common changes of passenger flow as follows. Passenger flow in four scenarios is as shown in Figures 6(a)–6(d).

5.1. Smooth Passenger Flow Case. To avoid the contingency of the experimental data, the GA and hyperheuristic algorithm were optimized ten times, respectively, and the experimental data results are described in Table 1.

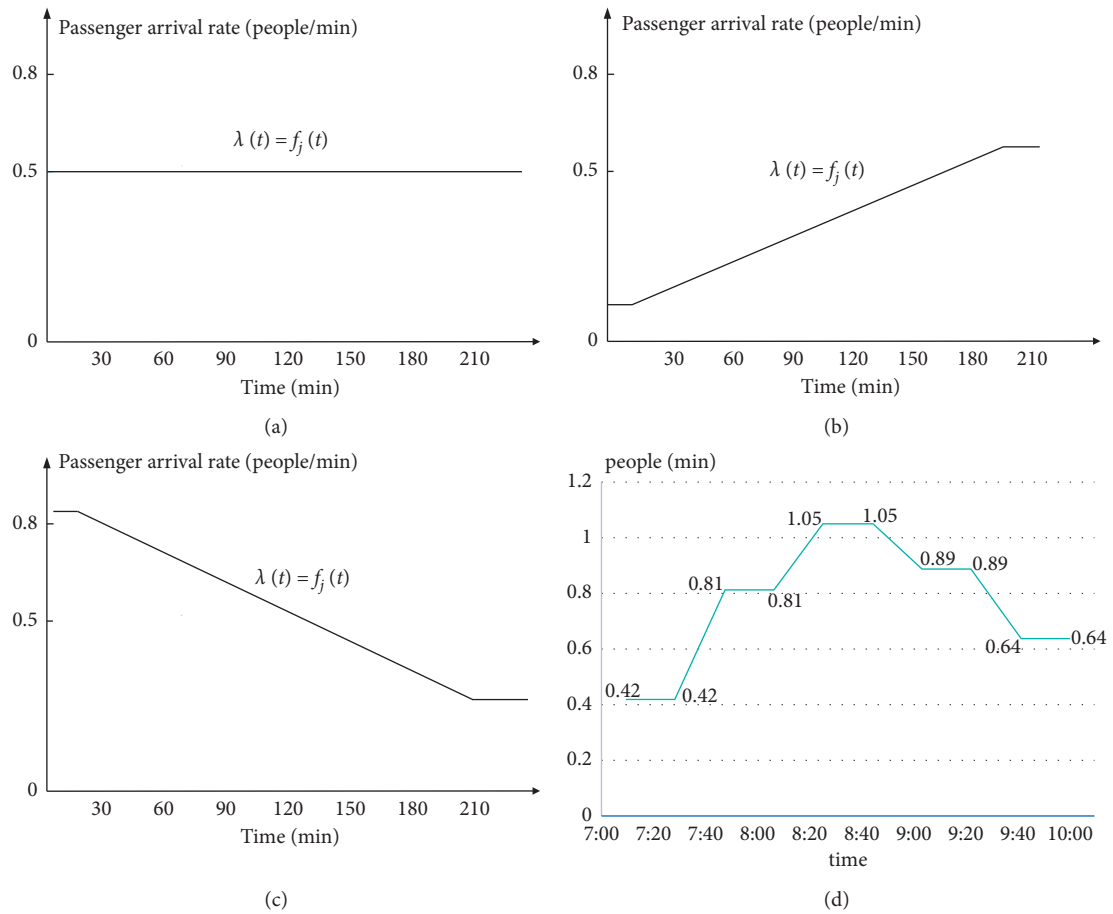

Figure 6: Passenger arrival rate under different scenarios. (a) Smooth passenger flow. (b) Increasing passenger flow. (c) Decreasing passenger flow. (d) Multisection convex passenger flow.

Table 1: Results of GA and hyperheuristic algorithm under smooth passenger flow.

Goal value	Passenger waiting time (min)					
Status	High intensity		Medium intensity		Low intensity	
Groups	GA	Hyperheuristic	GA	Hyperheuristic	GA	Hyperheuristic
1	329311	236758	130397	95405	42986	31683
2	299713	221585	108388	88564	44676	37413
3	281075	239199	125027	95591	44446	31204
4	281262	240604	124354	94134	45572	31931
5	293368	232556	123672	89288	45917	30235
6	294682	227240	124710	95649	47380	34739
7	287362	210243	134979	106300	43900	33171
8	310073	214497	130351	107433	41574	34502
9	324412	235954	123422	88144	45835	33429
10	315976	234687	124605	95760	41269	35507
Average value	301723.4	229332.3	124990.5	95626.8	44355.5	33381.4
Promotion rate	—	23.99%	—	23.48%	—	24.74%

In Table 1, the optimal performance of the hyperheuristic algorithm under smooth passenger flow is better than that of GA strategy, which can be improved by 24.74% under low-intensity passenger flow, and about 23.48~23.99% under medium- and high-intensity passenger flow, respectively. Although the results of the hyperheuristic algorithm fluctuate, the optimal results of each intensity of the hyperheuristic algorithm under the smooth passenger flow model are better than those of GA. We can conclude that the hyperheuristic algorithm is used as a solution and can find good combinations of heuristic operators and problem domain optimal results.

The experimental data obtained above, shown in Figures 7(a)–7(c), show the solution amplitude curve of GA

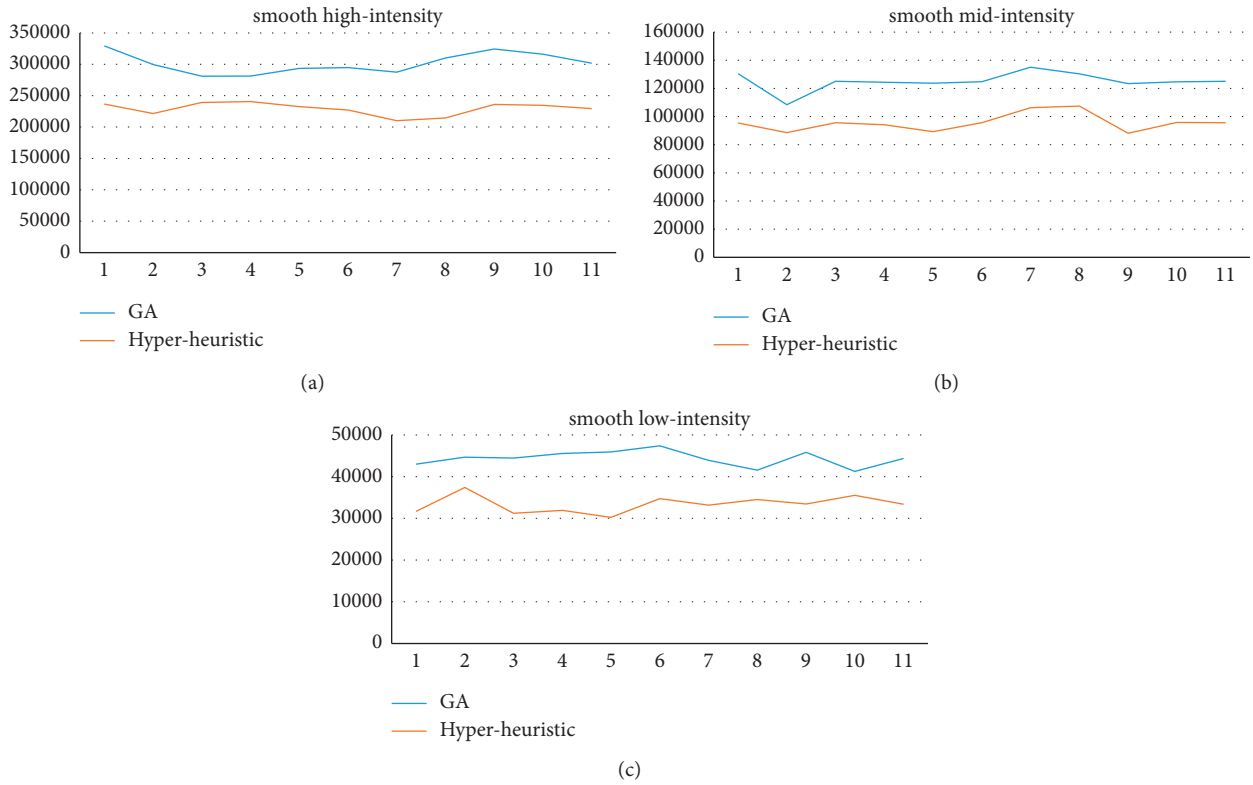

Figure 7: Result curve of GA and hyperheuristic algorithm of smooth passenger flow. (a) High-intensity passenger flow. (b) Medium-intensity passenger flow. (c) Low-intensity passenger flow.

and hyperheuristic algorithm under smooth high-, medium-, and low-intensity passenger flow models.

According to the result data obtained in Table 1, the data in Table 2 is calculated using

$$SE = \frac{\sqrt{(1/N)\sum_{i=1}^{N}(x_i - \overline{x})^2}}{\overline{x}} \times 100\%. \quad (8)$$

By analyzing the stability, it can be seen that there is some instability in both the GA and hyperheuristic algorithm.

The smaller the index, the more stable. In the smooth passenger flow model, the stability of medium and low intensity of hyperheuristic algorithm is lower than that of the GA. In comparison, the stability of the high-intensity heuristic algorithm is higher than that of GA strategy.

5.2. Increasing Passenger Flow Case. The GA and hyperheuristic algorithms were optimized ten times, respectively, and Table 3 offers the experimental results.

In Table 3, all the average value of the second algorithm is better than that of the first. The target value of the high-intensity passenger flow can be reduced by 26.44%. The target value of the medium-intensity and low-intensity passenger flow can be reduced by 27.17% and 21.84%. The hyperheuristic algorithm can find better solutions that GA cannot find. But the same hyperheuristic algorithm may also only find poor results, as shown in the 9th group of low-intensity experiments. For the increasing passenger flow model, the optimal capability of the hyperheuristic algorithm is far higher than that of the GA, and the target value can be reduced by 21% to 27%.

According to the experimental data obtained above, Figures 8(a)–8(c) show the solution amplitude curve of GA and hyperheuristic algorithm under the increasing high, medium, and low passenger flow.

Table 4 is stability analysis table under increasing passenger flow.

In Table 4, the stability of the hyperheuristic algorithm under the increasing passenger flow model is poor compared with that of the GA. The stability indicators of GA are lower about 3% than those of hyperheuristic algorithm, and the stability of the latter is slightly worse than that of the GA.

5.3. Decreasing Passenger Flow Case. The experimental data obtained after ten groups of experiments are listed in Table 5.

All the average values of the hyperheuristic algorithm are smaller than those of GA in Table 5. The passenger waiting time of the high-intensity, medium-intensity, and low-intensity passenger flow can be reduced by 22.59%, 18.83%, and 20.29%, respectively. The promotion rate of the medium-intensity is the lowest of the three types. The hyperheuristic algorithm can find better solutions that GA cannot find. But the hyperheuristic algorithm may also find poor results as shown in the 10th group of high-intensity and the 8th group of low-intensity experiments. For the decreasing

Table 2: Stability analysis table of GA and hyperheuristic algorithm under smooth passenger flow.

Status	High intensity		Medium intensity		Low intensity	
Algorithm	GA	Hyperheuristic	GA	Hyperheuristic	GA	Hyperheuristic
Standard deviation	16502.76	10056.55	6619.75	6337.64	1863.11	2086.78
Average value	301723.4	229332.3	124990.5	95626.8	44355.5	33381.4
Stability indicators	5.47%	4.39%	5.30%	6.63%	4.20%	6.25%

Table 3: Results of GA and hyperheuristic algorithm under increasing passenger flow.

Goal value	Passenger waiting time (min)					
Status	High intensity		Medium intensity		Low intensity	
Groups	GA	Hyperheuristic	GA	Hyperheuristic	GA	Hyperheuristic
1	457448	349016	245116	180496	103686	78420
2	434936	301537	243755	178577	98125	74349
3	413338	274837	237078	192921	106780	73530
4	422933	327430	258363	163234	93435	65532
5	402986	359045	245019	192164	95030	67523
6	457998	309307	237237	157055	95030	84561
7	439372	299749	250711	181272	95338	78232
8	428577	348443	238049	174488	100008	69985
9	442198	337899	235589	185132	98200	92451
10	459302	299474	248547	171264	101046	87478
Average value	435908.8	320673.7	243946.4	177660.3	98778.6	77206.1
Promotion rate	—	26.44%	—	27.17%	—	21.84%

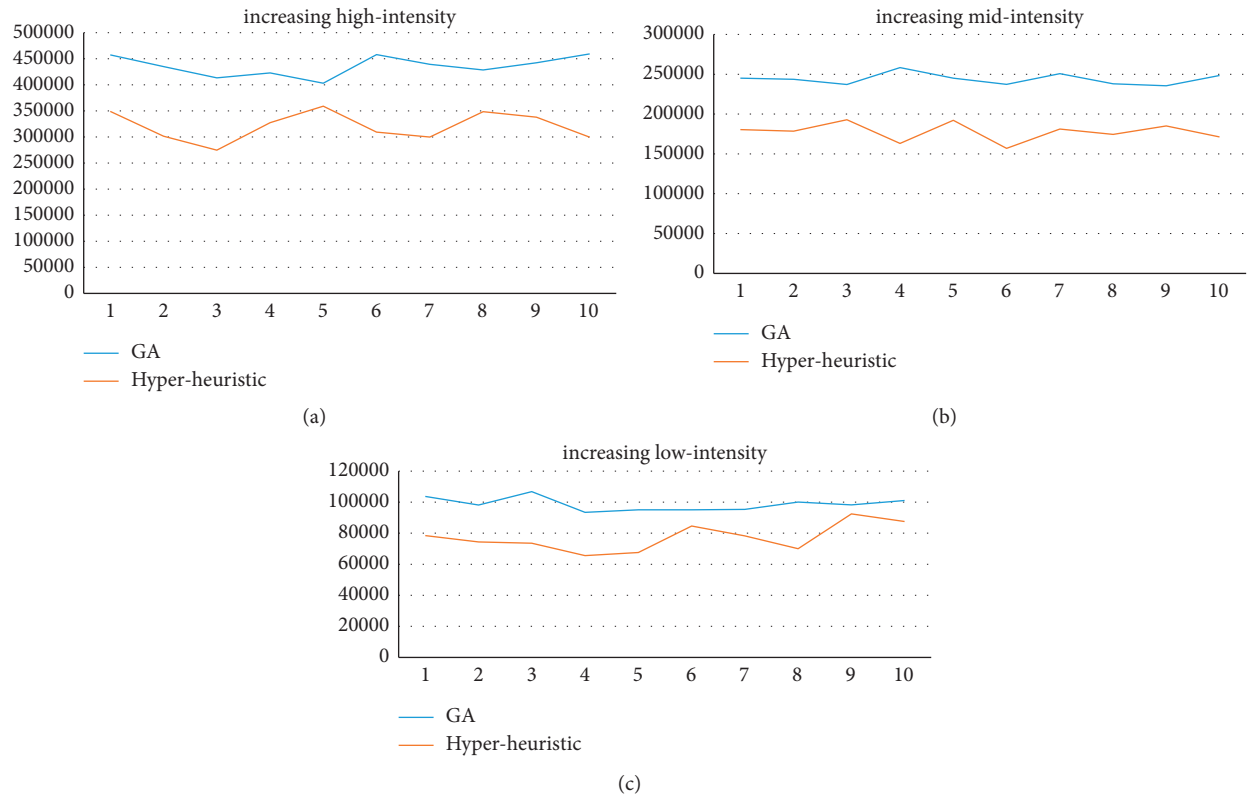

Figure 8: Result curve of GA and hyperheuristic algorithm of increasing passenger flow. (a) High-intensity passenger flow. (b) Medium-intensity passenger flow. (c) Low-intensity passenger flow.

Table 4: Stability analysis table of GA and hyperheuristic algorithm under increasing passenger flow.

Status	High intensity		Medium intensity		Low intensity	
Algorithm	GA	Hyperheuristic	GA	Hyperheuristic	GA	Hyperheuristic
Standard deviation	18349.62	26216.07	6885.75	10975.75	4048.36	8357.08
Average value	435908.8	320673.7	243946.4	177660.3	98778.6	77206.1
Stability indicator	4.21%	8.18%	2.82%	6.18%	4.10%	10.82%

Table 5: Results of GA and hyperheuristic algorithm under decreasing passenger flow.

Goal value	Passenger waiting time (min)					
Status	High intensity		Medium intensity		Low intensity	
Groups	GA	Hyperheuristic	GA	Hyperheuristic	GA	Hyperheuristic
1	258389	204103	121557	97828	39418	35527
2	267276	184474	119809	100364	41804	28754
3	253559	210053	116803	102049	39258	30514
4	271737	225726	116636	102628	40272	36450
5	266677	208323	120044	95586	38907	29451
6	283306	190228	120053	88542	38336	30042
7	282428	193575	122068	103520	38165	29310
8	283881	199872	119520	98754	38347	36415
9	264076	223142	124627	96882	38146	27461
10	256232	240859	1285 07	95662	38154	27598
Average value	268756.1	208035.5	120962.4	98181.5	39080.7	31152.2
Promotion rate	—	22.59%	—	18.83%	—	20.29%

passenger flow, the optimal capability of the hyperheuristic algorithm is much better than that of GA, and the target value can be reduced by 18% to 22%.

According to the experimental data obtained in Table 5, Figures 9(a)–9(c)show the solution amplitude curve of GA and hyperheuristic algorithm under the decreasing high, medium, and low passenger flow.

Table 6 is a stability analysis table under decreasing passenger flow.

The stability of the hyperheuristic algorithm under decreasing passenger flow model is also poor compared with GA. The stability indicators of low intensity of GA are lower about 8% than those of the hyperheuristic algorithm, and the stability of the latter is worse than that of GA in solving decreasing passenger flow problem.

5.4. Multisegment Convex Passenger Flow Case. To improve the results accuracy, ten experiments are completed for comparison. The experimental results data are listed in Table 7.

As shown in Table 7, the GA and the hyperheuristic column correspond to ten experimental results under the same parameters. The hyperheuristic algorithm under multisection convex passenger flow designed can obtain better results than GA, and the average value can be reduced by 28.62%. The promotion rate is the highest among smooth, increasing, decreasing, and multisegment convex passenger-flow modes.

The right column of Table 7 represents the high-level individuals corresponding to the optimal fit found by the current hyperheuristic algorithm. Each individual has seven digits of 1~7, corresponding to a different low-level heuristic operator. The optimal value can be found in which the locus of 0 is without a corresponding heuristic operator operation. The low-level heuristic operators constituted by [3, 3, 3, 3, 3, 5, 6] are a combination of heuristic operators with optimal performance in Table 7.

Figure 10 is the statistics of the various low-level heuristic operators in the results.

In Figure 10, the LLH3 operator accounts much more than the rest of the results, which can be considered as a high-quality low-level heuristic operator for the problem. And the LLH1 operator appears 12 times, while the LLH7 operator has the lowest proportion, which can be considered an operator design that does not meet the optimization requirements.

Because of the large number of LLH3 operators in the found optimal results, it is speculated that the LLH3 operator alone can find better results. High-level individuals shown in [3, 3, 3, 3, 3, 3, 3] are designed to perform experiments, with several optimization results in Table 8.

It is concluded from the data in Tables 7 and 8 that LLH3 operators alone do not perform as well as combination operators of [3, 3, 3, 3, 3, 5, 6]. The guess mentioned above is that the LLH3 operator alone can find better results, which is invalid.

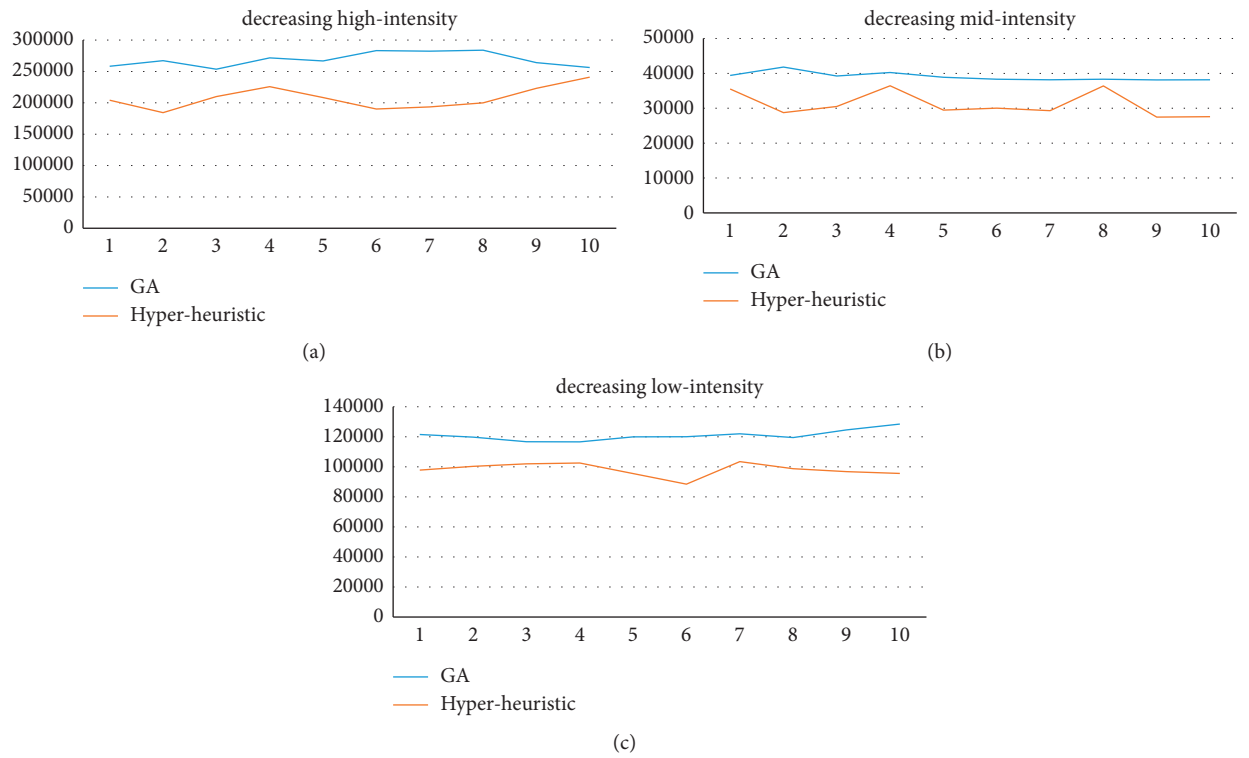

Figure 9: Result curve of GA and hyperheuristic algorithm of decreasing passenger flow. (a) High-intensity passenger flow. (b) Medium-intensity passenger flow. (c) Low-intensity passenger flow.

Table 6: Stability analysis table of GA and hyperheuristic algorithm under decreasing passenger flow.

Status	High intensity		Medium intensity		Low intensity	
Algorithm	Hyperheuristic	GA	Hyperheuristic	GA	Hyperheuristic	GA
Standard deviation	10770.18	16691.79	2351.84	4196.88	1125.82	3388.79
Average value	268756.1	208035.5	120962.4	98181.5	39080.7	31152.2
Stability indicator	4.01%	8.02%	1.94%	4.27%	2.88%	10.88%

Table 7: Results of GA and hyperheuristics algorithm under multisection convex passenger flow.

Goal value	Passengers' waiting time (min)		
Algorithm	GA	Hyperheuristic	High-level individuals
1	161743	122724	5323130
2	158880	105535	1233343
3	152598	125635	3133130
4	156932	115500	6651333
5	152687	119261	3113113
6	158178	97820	3333563
7	153301	114023	2433143
8	164859	117552	3323331
9	166144	113172	6337633
10	176650	112208	6315335
Average value	160197.2	114343	—
Promotion rate	—	28.62%	—

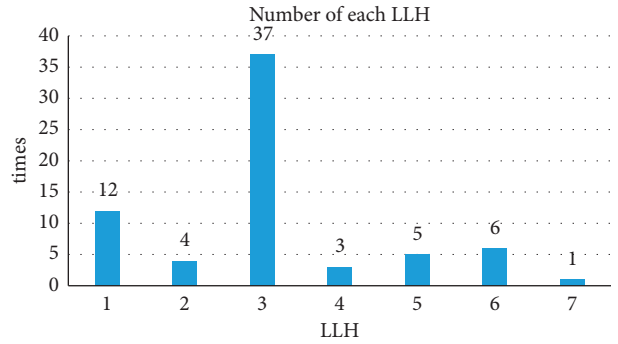

Figure 10: Number of each LLH.

Table 8: Optimization results of LLH3.

1	2	3	4	5	6	7	8	Average value
145676	121937	147752	151445	159972	149905	154113	145614	149389

The average value of Table 8 is 149389, which not only is far higher than the target value of [3, 3, 3, 3, 3, 5, 6] but is higher than the average value 114343 of hyperheuristic algorithm in Table 7.

6. Conclusion

A hyperheuristic algorithm based on GA is developed for solving the joint scheduling model of bus multiline departure time and vehicle speed. The cross, variation, selection operator, and various low-level heuristics in the high-level scheduling tactics of the hyperheuristic algorithm are designed in the study.

This study is based on the GA and hyperheuristic algorithm, performed ten sets of experiments using the hyperheuristic algorithm for the four-passenger flow modes, and compared the results solved by the GA and the hyperheuristic algorithm. The experimental results show that the optimization performance of the hyperheuristic algorithm is better than that of GA in the smooth, increasing, decreasing, and multisegment convex passenger flow modes. For smooth passenger flow mode, the promotion rate is about between 23% and 24%. For decreasing passenger flow mode, the average value of the medium and high passenger flow obtained by the hyperheuristic algorithm was improved by about 26 to 27%, by 21.84% for low passenger flow. It dramatically reduces the waiting time for passengers. For the decreasing passenger flow mode, the average value of the hyperheuristic algorithm was improved by 18 to 22%.

The stability of the three modes of GA and hyperheuristic algorithm is also compared in the study. For smooth passenger flow mode, the stability of the medium and low density of GA outperformed the metrics of the hyperheuristic algorithm. In contrast, the hyperheuristic high-density stability outperformed GA. The results also show that hyperheuristic algorithm shows lower stability than GA in solving increasing and decreasing passenger flow problems.

The hyperheuristic algorithm designed under multiple convex passenger flow works better than GA, and the average value can be reduced by 28.62%, greatly reducing passengers' waiting time. Through analysis of the designed multiple sets of low-level heuristic operators, it is concluded that LLH3 is more suitable for the studied multiple convex passenger flow.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was sponsored by the National Natural Science Foundation of China (nos. 71601126, 71831006, 71771070, and 71571037) and the Natural Science Foundation of Liaoning Province of China (no. 20180550423).

References

[1] Z. B. Chen, "China car ownership data report," Reports/specil/202118/36598, Trieworld, Karnataka, India, 2020.

[2] Y. Liu, X. Luo, S. Cheng, Y. Yu, and J. Tang, "Dynamic bus scheduling of multiple routes based on joint optimization of departure time and speed," *Discrete Dynamics in Nature and Society*, vol. 2021, pp. 1–20, 2021.

[3] P. G. Furth and N. Wilson, "Setting frequencies on bus routes: theory and practice," *Transportation Research Record*, vol. 818, pp. 1–7, 1981.

[4] A. Ceder, "Bus frequency determination using passenger count data," *Transportation Research Part A General*, vol. 18, no. 5-6, pp. 439–453, 1984.

[5] A. Ceder and H. I. Stern, "Optimal transit timetables for a fixed vehicle fleet," in *Proceedings of the 10th International*

Symposium on Transportation and Traffic, vol. 15, no. 1, pp. 331–355, Delft, The Netherlands, July 1984.

[6] A. Ceder and O. Tal, "Designing synchronization into bus timetables," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1760, no. 1, pp. 28–33, 2001.

[7] A. Ceder, "Bus timetables with even passenger loads as opposed to even headways," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1760, no. 1, pp. 3–9, 2001.

[8] A. Ceder, B. Golany, and O. Tal, "Creating bus timetables with maximal synchronization," *Transportation Research Part A: Policy and Practice*, vol. 35, no. 10, pp. 913–928, 2001.

[9] M. Dessouky, R. Hall, A. Nowroozi, and K. Mourikas, "Bus dispatching at timed transfer transit stations using bus tracking technology," *Transportation Research Part C: Emerging Technologies*, vol. 7, no. 4, pp. 187–208, 1999.

[10] R. Hall, M. Dessouky, and Q. Lu, "Optimal holding times at transfer stations," *Computers & Industrial Engineering*, vol. 40, no. 4, pp. 379–397, 2001.

[11] M. Dessouky, R. Hall, L. Zhang, and A. Singh, "Real-time control of buses for schedule coordination at a terminal," *Transportation Research Part A: Policy and Practice*, vol. 37, no. 2, pp. 145–164, 2003.

[12] M. S. Chowdhury and S. Chien, "Dynamic vehicle dispatching at the intermodal transfer station," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1753, no. 1, pp. 61–68, 2001.

[13] C. Liebchen and R. H. Möhring, "A case study in periodic timetabling," *Electronic Notes in Theoretical Computer Science*, vol. 66, no. 4, pp. 18–31, 2002.

[14] C. Fleurent, R. Lessard, and L. Séguin, "Transit timetable synchronization: evaluation and optimization," in *Proceedings of the 9th international conference on computer-aided scheduling of public transport*, vol. 1, pp. 9–11, San Diego, CA, USA, August 2004.

[15] M. Li and X. Li, "Study on the real-time scheduling optimization method of multi-line vehicles in the bus hub," *Highway traffic technology*, vol. 23, no. 10, pp. 108–112, 2006.

[16] Y. Y. Ulusoy, S. I.-J. Chien, and C.-H. Wei, "Optimal all-stop, short-turn, and express transit services under heterogeneous demand," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2197, no. 1, pp. 8–18, 2010.

[17] L. Sun, *Optimization Method of Dynamic Transmission of Urban Bus under the Internet of Things Environment*, Northeastern University, Boston, MA, USA, Article ID 084816, 2019.

[18] S. Song and W. S. Zhang, "Bus query system based on the minimum transfer algorithm," *Computer Knowledge and Technology*, vol. 14, no. 1, pp. 96–98, 2018.

[19] J. Chen, Z. Liu, S. Wang, and X. Chen, "Continuum approximation modeling of transit network design considering local route service and short-turn strategy," *Transportation Research Part E: Logistics and Transportation Review*, vol. 119, no. 6, pp. 165–188, 2018.

[20] X. M. Song, M. Y. Zhang, and J. L. Jing, "Consider multi-operator bus scheduling optimization with overlapping intervals," *Transportation System Engineering and Information*, vol. 20, no. 5, pp. 142–147, 2020.

[21] P.-L. Bourbonnais, C. Morency, M. Trépanier, and É. Martel-Poliquin, "Transit network design using a genetic algorithm with integrated road network and disaggregated O-D demand data," *Transportation*, vol. 48, no. 1, pp. 95–130, 2021.

[22] X. Luo, Y. Liu, Y. Yu, J. Tang, and W. Li, "Dynamic bus dispatching using multiple types of real-time information," *Transportation Business: Transport Dynamics*, vol. 7, no. 1, pp. 519–545, 2019.

[23] A. Sun and M. Hickman, "The holding problem at multiple holding stations," in *Proceedings of the The 9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT)*, San Diego, CA, USA, August 2004.

[24] F. Delgado, J. C. Munoz, and R. Giesen, "How much can holding and/or limiting boarding improve transit performance?" *Transportation Research Part B: Methodological*, vol. 46, no. 9, pp. 1202–1217, 2012.

[25] P. Chandrasekar, R. Long Cheu, and H. C. Chin, "Simulation evaluation of route-based control of bus operations," *Journal of Transportation Engineering*, vol. 128, no. 6, pp. 519–527, 2002.