A Selection Hyper-heuristic based on Q-learning for School Bus Routing Problem

Yan-e Hou, Wenbo Gu, Chunxiao Wang, Kang Yang and Yujing Wang

Abstract—**School bus routing problem (SBRP) has been studied for decades. Many successful approaches based on heuristics or metaheuristics have been developed for various SBRP problems. However, developing an effective algorithm for SBRP is still a very challenging task. This paper developed a Q-learning-based selection hyper-heuristic to solve basic and open single-school SBRP problems, which both aim to minimize the total travel distance. The proposed algorithm took a Q-learning algorithm as the high-level strategy to select a low-level heuristic from a set of low-level heuristics, which are dependent on the problem domain. The selected low-level heuristic was regarded as an action and then executed to improve the current solution. In each stage of the optimization process, the best action with the best cumulative rewards will be chosen to get better results. The presented algorithm was implemented and some experiments were carried out on some Capacitated vehicle routing problem (CVRP) instances and SBRP benchmark instances. Experiment results on two types of instances demonstrate that our proposed hyper-heuristic algorithm is more competition than existing approaches.**

Index Terms—**Q-learning, hyper-heuristic, school bus routing problem, vehicle routing, heuristic.**

I. Introduction

PROVIDING a safe, reliable, and low-cost school bus transportation system for the students in compulsory education is the main problem faced by school administrators and school bus service providers. However, planning a school bus routing system is very complex and challengeable. On one hand, school bus routes planning involves many factors such as schools, bus stations, students, school buses, and road networks. On the other hand, the differences in problem characteristics and planning objectives in the practical application will also lead to more complex problems. Therefore, the development of an efficient route generation algorithm is still the focus of current research [1].

School bus routing problem (SBRP) is closely related to vehicle routing problem (VRP), which also can be considered as an application branch of VRP. SBRP aims to find an optimal scheduling plan, which uses a fleet of school buses to pick up the students from the bus stations and send them to the designated school or send the students from their school to the bus stations while satisfying the bus capacity, school time windows and other constraints. SBRP is a well-known NP-hard combinatorial optimization problem. Since provided by [2], SBRP has been widely studied. The recent review of SBRP could be seen in [3] and [4].

The commonly used solution approaches for SBRP consist of exact algorithm, heuristic algorithm, and metaheuristic algorithm [3]. Exact algorithms such as dynamic programming, column generation and cutting planes, can solve small-scale instances optimally, but they are very time-consuming. For middle or large problems, heuristic and metaheuristic approaches are utilized. Traditional heuristic methods are commonly used to obtain an initial solution or improve the solution by using some neighborhood operators. Metaheuristics have a special scheme to avoid trapping local optima, and they can find a near-optimal solution in a logical time. The metaheuristics include iterated local search (ILS), hill-climbing, simulated annealing, genetic algorithm, ant colony optimization, and so on. Metaheuristics or hybrid metaheuristics have been widely used in SBRP [4],[5],[6],[7],[8].

The successful algorithms mentioned above have greatly promoted the development of research on SBRP problems. However, metaheuristics trend to be problem-specific based on the No-free-lunch theorem [9]. The well-designed algorithm for a certain SBRP problem is difficult to solve another SBRP problem, because it needs to be redesigned to get better performance. Furthermore, the algorithm inevitably requires complex parameter settings and adjustments. Thus, it is very important and meaningful to develop a unified effective methods for SBRP problems in various application scenarios.

Hyper-heuristic is a general-purpose heuristic algorithm, which uses a high-level strategy (HLS) to conduct a set of low-level heuristics (LLH). The hyper-heuristic manipulates a set of pre-designed low-level heuristics instead of operating directly on solutions [10]. It has the advantages of simple parameter tuning, easy design and implementation. For hyper-heuristic, LLH is dependent on the problem domain and HLS is responsible for managing or manipulating intelligently existing low-level heuristics to solve problems. Hyper-heuristic can solve cross-domain problems and various variants of the same problem [10],[11]. Recently, some hyper-heuristic algorithms have been applied to solve VRP problems [12],[13],[14]. It shows the hyper-heuristic has the potential to solve SBRP, because SBRP is regarded as a special variant of VRP.

This paper tries to design a selection hyper-heuristic algorithm (denoted as HHQL), which can solve the route planning problem for a single-school SBRP. The selection method based on Q-learning [15] is used to choose the

Manuscript received April 20, 2022; revised August 30, 2022. This research has been supported by National Natural Science Foundation of China (Grant No.41801310).

Yan-e Hou is an associate professor of Henan Key Laboratory of Big Data Analysis and Processing, Henan University, Kaifeng, 475004, China (e-mail: houyane@henu.edu.cn)

Wenbo Gu is a graduate student of College of Computer and Information Engineering, Henan University, Kaifeng, 475004, China (e-mail:guwenbo@henu.edu.cn)

Chunxiao Wang is a graduate student of College of Computer and Information Engineering, Henan University, Kaifeng, 475004, China (e-mail:henuwcx@henu.edu.cn)

Kang Yang is a graduate student of Henan Key Laboratory of Big Data Analysis and Processing, Henan University, Kaifeng, 475004, China (e-mail:yangkang@henu.edu.cn)

Yujing Wang is a lecturer of College of Computer and Information Engineering, Henan University, Kaifeng, 475004, China (corresponding author,e-mail:yjwang@henu.edu.cn)

low-level heuristics, and a Montel Carlo acceptance rule is used to accept the neighborhood solution obtained by the selected low-level heuristic. The proposed algorithm is tested on benchmark instances, and the experimental results reveal that our proposed algorithm can find a good solution in a limited time and has good stability.

The rest of the paper is organized as follows. Section II describes the research problem in this paper. Section III introduces the Q-learning algorithm. The designed hyper-heuristic algorithm is shown in section IV. Computational experiments and results analysis are given in Section V. Finally, Section VI summarizes this paper and draws a conclusion.

II. Problem Description

In this paper, we focus on a homogeneous single-school SBRP in different application scenarios. There are two kinds of single-school SBRP problems. For a basic single-school SBRP, all the school buses return to the depart station after they serve all the students. It is a variant of classical capacitated VRP (CVRP). When the school buses do not return to the depart station, the single-school SBRP is a case of open VRP. In this paper, we take the basic single-school SBRP and open single-school SBRP into account.

We assume that the school node is the depart station. There are a set of buses located at the school, and the fleet types of them are the same. For anyone school bus, it has a certain capacity. There are some students at every student station. When the school bus serves the student station, it needs some service time related to the number of students. For any two nodes, there is a travel cost between them. The constraints are defined as follows. Every school bus starts from the school and ends at the school node or other position. The student station must be served only once by a school bus. At any time, the total number of students on the bus must not exceed the capacity of the school bus. For every student, his riding time on the bus must be less than or equal to the constraint of maximum student's riding time. For a school bus transportation system, the cost is the first and main objective considered by buses service providers. Because the purchase cost of school buses is given in advance, the daily operation cost that is the total travel distance is mainly considered. Therefore, the optimization objective is to minimize the total travel distance. The formulation of this problem is similar to the model that was built in [4].

III. Q-learning Algorithm

Q-learning is one of the most efficient methods of reinforcement learning techniques [15]. Q-learning is made up of five components, which are agent, environment, state, action, and reward. The agent interacts with the environment, where the agent can select actions, and the environment gives a response to the agent. The interaction process will execute iteratively until it reaches a final state. Q-learning aims to learn the value of state-action pair called Q-value. For every state-action pair, $Q(s,a)$ represents the expected reward for state s and action a respectively. Assume that $S = \{s_1, s_2, ..., s_n\}$ is a set of possible states, $A = \{a_1, a_2, ..., a_m\}$ denotes a set of actions that can be selected. At state s_t, the agent performs the action a_t, and then the immediate reinforcement reward r_t will be obtained. The learning rate α is a number between 0 and 1, which is used to balance exploration and exploitation. γ is a discount factor, which represents the influence of future rewards. The Q-table stores cumulative rewards that are used to evaluate the best state-action pair. The Q-value at time t denotes $Q_t(s_t, a_t)$, and the value of $Q_{t+1}(s_t, a_t)$ is computed by a Q-functions as in the equation (1).

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha\{r_t + \gamma argmax Q_{t+1}(s_{t+1}, a) - Q_t(s_t, a_t)\} \quad (1)$$

As mentioned above, the function structure of the Q-learning can be descried in the following.

(1) Initialize the Q-value randomly or zero for each state-action pair.

(2) Denote the current state as s_t.

(3) Choose an action a_t using the action strategy.

(4) Execute the selected action a_t, receive the reward r_t and determine the next state s_{t+1}.

(5) Update the value of $Q_{t+1}(s_t, a_t)$ using the equation(1).

(6) Set $S_t = S_{t+1}$, and then go to step(3) until the termination condition is satisfied.

IV. Proposed Hyper-heuristic Algorithm

In this section, we will describe the Q-learning-based hyper-heuristic for solving the single-school SBRP. First, the overall framework of our proposed hyper-heuristic algorithm is described. Then, a set of problem-specific low-level heuristics are designed which are used as the actions in the Q-learning algorithm. Finally, we introduce the implementation of the high-level strategy, including the action selection method, reward evaluation function, and neighborhood acceptance rule.

A. Overall Description of HHQL

The proposed hyper-heuristic HHQL is a single-solution-based hyper-heuristic. The HHQL algorithm starts with an initial solution, and iteratively chooses a low-level heuristic to execute, and then decides to accept or reject the solution found by the selected low-level heuristic during the optimization process. Thus, the key to HHQL is the design of low-level heuristics, selection strategy and acceptance rules. Algorithm 1 describes the overall framework of HHQL.

In Algorithm 1, step (1) and (2) initialize the values of parameters. The initial solution is obtained by the cheapest insertion method in step (3). The main loop of the proposed algorithm is step (4) $\sim$ step (28), which consists of three phases. In the first phase, an action will be chosen by the ε-greedy selection strategy in step (5) $\sim$ step (9). The selected action, that is, low-level heuristic is executed during an episode. Then the global best solution will also be updated in the second phase, which is described in step (10) $\sim$ step (21). Additionally, when the low-level heuristic based on ruin-and-recreate principle is chosen, the execution procedure of it must be take the destruction factor and the maximum trail number into account. At the last phase, the state-action pair $Q_{t+1}(s_t, a_t)$ are calculated and the current state will be updated from step (22) to step (26). These three phases will be literately executed until the algorithm meets the termination condition.

Algorithm 1 HHQL

Input: the maximum iteration *maxiter*, the neighbor list size *nb*, learning rate α, the discount factor γ, the length of episode *ep*, the destruction factor ρ and maximum trail number *iter*.

Output: the best solution S^*.

```
S = S* = NULL; Initialize the set of state and action list A; Set iteration number t = 1; ε = 0.9;
Initialize the value of every state-action pair in Q_table to zero;
Get an initialization solution S by cheapest insertion method;
while t≤maxiter do
    if random number rand < ε then
        select an action a_t randomly from A;
    else
        select an action a_t with the max Q-value at current state s_t in the Q-table;
    end if
    for int j = 0; j < ep; j++ do
        if a_t is ruin-and-recreate low-level heuristic then
            S_n =AppliedLLH(S, a_t, nb, ρ, iter);
        else
            S_n =AppliedLLH(S, a_t, nb);
        end if
        if S_n meets the acceptance rule then
            accept S_n, at the same time update the global best solution S*;
        else
            reject S_n;
        end if
    end for
    calculate the reward r and get the next state s_{t+1};
    find the maximum value of Q_t(s_{t+1}, a) of all the actions, ∀a ∈ A;
    update the learning rate α;
    update the Q_{t+1}(s_t, a_t) value by the equation(1) and set s_t = s_{t+1};
    S = S_n;
    t++;
end while
```

B. Actions

The low-level heuristic is dependent on the solving problem, with affects the efficiency of the hyper-heuristic algorithm. For single-school SBRP, we design seven simple and easy-to-implement heuristics to improve the neighborhood solution. The set of low-level heuristics are made up of six regular neighborhood operators and one neighborhood operator based on the ruin-and-recreate principle. These low-level heuristics are regarded as different actions in the Q-learning algorithm. Additionally, every low-level heuristics must be used to produce a feasible solution without violating any constraints.

The seven low-level heuristics are described in the following. For some low-level heuristics, they include inter-route and intra-route two operations. The intra-route operation does not require checking the constraints, and the inter-route operation must be check the constraints and produce a feasible solution. Meanwhile, to illustrate the function of these low-level heuristics, we take two routes R_1 {0-1-2-3-4-5-s} and R_2 {0-6-7-8-9-s} as examples. In the route, 0 and s indicate the depot and the school respectively, the other numbers represent the student stations.

(1) **One-point-move** (LLH_1): Remove one student station from a route and then insert it into another position of the same route or another route. For example, if the student station 1 is removed and then inserted behind the student station 3 on the same route R_1, the route R_1 is changed into $R_1^{'}$ {0-2-3-1-4-5-s}. If the student station 1 is shifted from route R_1 to the behind student station 7 on the route R_2. The routes R_1 and R_2 will change into $R_1^{'}$ {0-2-3-4-5-s} and $R_2^{'}$ {0-6-7-1-8-9-s}.

(2) **Two-point-swap** (LLH_2): Select two different student stations from the same route or two different routes, and swap them. When we exchange two student stations 2 and 4 in the route R_1, the new route is $R_1^{'}$ { 0-1-4-3-2-5-s }. We select the station 3 from route R_1 and station 7 from R_2 and then swap them. The new routes $R_1^{'}$ and $R_2^{'}$ are denoted as { 0-1-2-7-4-5-s } and {0-6-3-8-9-s}.

(3) **2-opt** (LLH_3): From a route, select two non-adjacent edges, reverse the nodes between the two edges, and then generate a new route. Suppose two edges 0-1 and 4-5 on the route R_1 are broken, and the new route $R_1^{'}$ is {0-1-3-2-4-5-s} after the use of the 2-opt operation.

(4) **Cross** (LLH_4): For two different routes, select an edge from each route, and then cross them. For example, we cross the edge 2-3 on the route R_1 and edge 7-8 on the route R_2, and the new routes are changed into $R_1^{'}$ {0-1-2-8-9-s} and $R_2^{'}$ {0-6-7-3-4-5-s}.

(5) **Or-opt** (LLH_5): Remove some student stations from one route and then insert them into another route. The number of student stations is usually an integer number between 2 and 4. Suppose the number of shifted stations is 2. When we shift stations 1 and 2 behind station 4 on the route R_1, the new route $R_1^{'}$ is {0-3-4-1-2-5-s}. When these two stations are shifted into behind station 8 on the route R_2, the two routes are changed into $R_1^{'}$ {0-3-4-5-s} and $R_2^{'}$ {0-6-7-8-1-2-9-s}.

(6) **Two-edge-swap** (LLH_6): Select two edges from two different routes, and swap them to generate two new routes. For example, we swap the edge 2-3 on the route R_1 and edge 7-8 on the route R_2, the new routes are changed into $R_1^{'}$ {0-1-7-8-4-5-s} and $R_2^{'}$ {0-6-2-3-9-s}.

(7) **Ruin-and-recreate** (LLH_7): For a solution, the ruin procedure is first executed, which selects some stations and then removes them from the current solution to get a partial solution. In the repair procedure, it tries to insert the removed stations into the partial solution to get a whole solution. The example of ruin-and-recreate low-level heuristic is shown in Fig. 1. As shown in Fig. 1(a), the original solution is made up of three routes, which includes 9 student stations. When the stations 1, 4, 5, 7 are removed from the solution, the partial solution is generated shown in Fig. 1(b). Then the removed stations are reinserted into the partial solution to procedure a whole solution (Fig. 1(c)).

The parameters of the ruin-and-recreate heuristic consist of the maximization iteration number and the degree of destruction. In general, the degree of destruction is a number between 0 and 1, which determines the number of removed stations. The destroy procedure and repair procedure will be

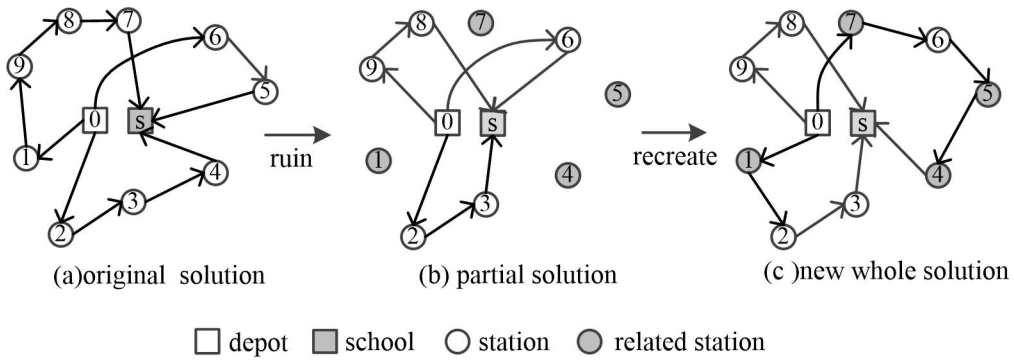


Fig. 1. an example of ruin-and-recreate low-level heuristic

iteratively executed many times until it meets the termination condition. This low-level heuristic can enlarge the solution space, which helps to find better neighborhood solutions.

C. Action Selection Strategy

In the learning process, a suitable action selection strategy can effectively balance exploration and exploitation. We use the ε-greedy policy to choose the action, which is defined in equation (2), Where k is a random number with [0, 1]. When k is smaller than ε, the action will be randomly selected from the action list. Otherwise, the action which makes the Q-value $Q(s_t, a_t)$ having the largest value, is selected. The value of ε is dynamically changed by equation (3), where t_c is the current evaluation, and t_{max} is the total evaluation time.

$$a_t = \begin{cases} random, & k < \varepsilon \\ argmax Q(s_t, a), & \forall a \in A, k \geq \varepsilon \end{cases} \tag{2}$$

$$\varepsilon = 1 - \frac{t_c}{t_{max}} \tag{3}$$

D. Reward Function

In Q-learning, the environment will respond a reinforcement signal, after an action was executed. The reinforcement signal r can be used to evaluate the performance of an action. For SBRP studied in this paper, minimizing the total travel distance is the main optimization objective. Thus, the improved degree of objective value can be a measure of evaluation. We use the improvement percentage defined in equation (4) as the reward or penalty signal.

$$r = \frac{f_{new} - f_{cur}}{f_{cur}} \times 100 \tag{4}$$

Where f_{cur} is the objective value before executing the action, f_{new} is the fitness of the new obtained solution by the low-level heuristic.

E. Q-function Parameters

There are two parameters in the Q-function defined in equation (1), and they are learning rate α and discount factor γ. For learning rate α, it represents the probability of accepting new information or maintaining the existing information. If α is a high value, it means that the new information tends to replace the existing information; Otherwise, it encourages more exploitation on the existing information. Inspired by [16], we also dynamically control the value of α by the equation (5). The value of the learning rate α will be decreased to encourage more exploitation during the optimization process.

$$\alpha = 1 - (0.9 \times \frac{t_c}{t_{max}}) \tag{5}$$

The discount factor γ denotes the influence of the long-term reward. A high γ value shows that the future rewards are more important than the current reward. If γ is low, the learning process will be more concerned about the current reward. In our proposed HHQL, the value γ is set to 0.8 after some experiments.

F. Acceptance Mechanism

The acceptance mechanism determines to accept or refuse the new solution found by the selected low-level heuristic, which is an action. To keep the diversity of solutions, we adopt Montel Carlo acceptance rules [17] to evaluate the new solution, which is a kind of non-deterministic acceptance rule. If the new solution outperforms the current solution, it is always accepted. The worse solutions that will be accepted must meet the equation (6).

$$k < e^{-\delta}, k \in [0, 1] \tag{6}$$

Where k is a random number between 0 and 1, δ is the change of objective value between the current solution and the new solution. Equation (6) indicates the worse solution will be accepted with a certain probability. As the objective value of the solution reduces, the probability of accepting a worse solution will decrease.

V. Computational Experiments

In this section, we conduct some experiments to evaluate the effectiveness of our proposed algorithm. First, we take the traditional CVRP with the standard benchmarks to verify the performance of the HHQL algorithm, because the problem issued in this paper is the same as the CVRP problem by updating the constraints condition. Then, we solve the SBRP by HHQL and compare it with some SBRP algorithms. Finally, we evaluate the performance of learning mechanism. The proposed algorithm was implemented based on the SBRP framework proposed by [22] and executed on a computer with an i7-6700 central processing unit at 3.4GHz and 8GB RAM.

A. Parameter Settings

In the proposed algorithm, many parameters need to be determined to get the best result. The maximization iteration number *maxiter* is set to 1000, the size of neighborhood list *nb* is set to 30. When the problem scale is smaller than 30, *nb* is the number of stations. The destruction factor ρ in the ruin-and-recreate low-level heuristic is set to 0.2 and the max trail number *iter* is 50. For Q-learning, there are four parameters including the learning factor α, discount factor γ , episode *ep* , and ε-greedy ε. The initial value of learning factor α and parameter ε are set to 1 and 0.9 respectively. The value of learning factor α and ε both are updated dynamically during the optimization process. After some experiments, the discount factor γ and the length of episode *ep* are set to 0.8 and 5 separately.

B. Experiment Results on CVRP

In this part, we modified the proposed algorithm to fit the CVRP by setting the service time of every student station to zero and ignoring the maximum ridding time constraint. We selected two groups of standard benchmark instances to investigate the effectiveness of the proposed HHQL algorithm. The first group of instances is the same as that used in [18], which is composed of three instances from Set A [19], two instances from Set B [19] and seven CMT instances [20]. The second group of instances have twelve instances from Set E [19]. These benchmark instances can be downloaded from the website (https://neo.lcc.uma.es). Each instance is executed 10 times.

TABLE I shows the result of instances in the first group obtained by our proposed algorithm and the other three algorithms, which are ACO&DE [18], LNS-ACO [21], and ILS [22] respectively. The columns *Stops* and *BKS* are the problem scale and the best-known solution. The column *Best* denotes the best solution obtained by the corresponding algorithm. Column *Gap* is the deviation percentage of the best solution and the best-known solution. The results shown in bold represent the best-known solution.

As shown in TABLE I, the HHQL outperforms other three algorithms, and the average deviation is just 0.43%. Among these algorithms, the performance of the ACO&DE algorithm is the worst. The LNS-ACO and ILS algorithms have the same average deviation of 0.47%. For the five small instances, all the algorithms can find the best-known solutions. While for the seven CMT instances, although the LNS-ACO can get 5 best-known solutions, the average deviation of all the instances is higher than the HHQL algorithm. The results reveal that the HHQL algorithm has good adaptability.

Next, we executed the HHQL algorithm to solve the Set E instances. Besides, we also compared it with the state-of-art CVRP algorithms, such as SC-ESA [23], LNS-ACO [21], CVRP-FA [24], ILS [22], SA [25], and ISA-CO [26]. The results of instances in set E are shown in TABLE II. In TABLE II, column *BKS* represents the best-known solutions, and other columns denote the deviation percentage relative to the best-known solution by the algorithms. If the value of the deviation percentage is null, it means that the value does not be provided by the corresponding literature. Meanwhile, the results shown in bold indicate the algorithm find the best-known solution of the corresponding instance.

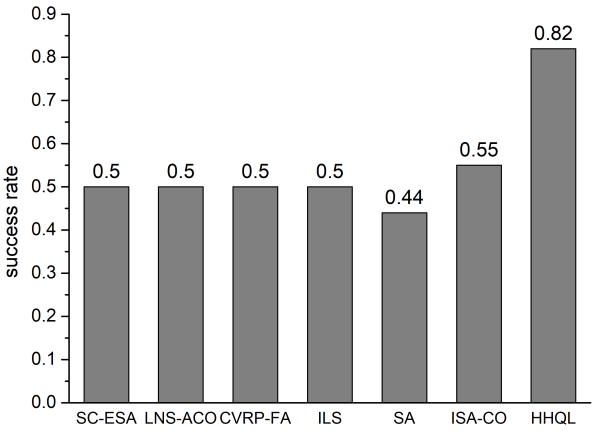


Fig. 2. the success rate of all the algorithms on set E

We can find from the TABLE II and Fig. 2 that the HHQL obtain the lowest average deviation value, just only 0.08%. Among these algorithms, the algorithms such as SC-ESA, LNS-ACO, CVRP-FA and ILS obtain 4 best-known solutions of 8 instances, and their success rate is 0.5. The SA algorithm finds 3 best-known solutions of 9 instances, whose success rate is 0.44. The ISA-CO and HHQL both solve 11 instances, and their success rates are 0.55 and 0.82 respectively. From these results, we can conclude that our proposed algorithm is effective.

C. Experiment Results on SBRP

In this section, we evaluate the performance of the proposed algorithm solving for SBRP. The difference between the SBRP issued in this paper and CVRP is that each stop station has a service time estimated by the number of students, expect for the students maximum ridding time constraints. This maximum ridding time constraint requires that the ridding time on the school bus of each student cannot exceed a predefined value. We use 12 benchmark instances proposed by [4] to test HHQL. In simple terms, the capacity of school bus is 66 and the speed of bus is 20 mile per hour. The maximum riding time is 2700 s, and the total distance are represented in miles.

To evaluate the performance of the HHQL algorithm, we compared it with two state-of-the-art single-solution metaheuristics: iterated local search (ILS) [22] and variable neighborhood search (VNS) [27]. For a fair comparison, we re-implemented these two metaheuristics algorithms based on the same SBRP framework [22]. They had the same parameters settings as the HHQL algorithm, and were also executed on the same computer. In additional, they used the same neighborhood operators as the low-level heuristics designed in HHQL. Because the computation time of three algorithms were basically the same, the computation time did not compare each other no longer.

First, we employed the HHQL algorithm to solve the basic SBRP, in which each school bus starts from the depot and ends to the school after it serves students. TABLE III shows the results obtained by the proposed algorithm and two single-solution metaheuristics including ILS [22] and VNS [27]. Columns *Stops* denotes the problem scale of the instance. Columns *TD* and *AvgTD* are the total travel distance and the average total travel distance. Columns G_1 and G_2

TABLE I
RESULTS ON THE FIRST GROUP OF INSTANCES

Instance	Stops	BKS	ACO&DE		LNS-ACO		ILS		HHQL	
			Best	Gap(%)	Best	Gap(%)	Best	Gap(%)	Best	Gap(%)
A-n32-k5	32	784	**784**	0.00	**784**	0.00	**784**	0.00	**784**	0.00
A-n33-k5	33	661	**661**	0.00	**661**	0.00	**661**	0.00	**661**	0.00
A-n33-k6	33	742	**742**	0.00	**742**	0.00	**742**	0.00	**742**	0.00
B-n31-k5	31	672	**672**	0.00	**672**	0.00	**672**	0.00	**672**	0.00
B-n34-k5	34	788	**788**	0.00	**788**	0.00	**788**	0.00	**788**	0.00
CMT1	50	524.61	**524.61**	0.00	**524.61**	0.00	**524.61**	0.00	**524.61**	0.00
CMT2	75	835.26	841.38	0.73	**835.26**	0.00	**835.26**	0.00	**835.26**	0.00
CMT3	100	826.14	832.62	0.78	**826.14**	0.00	827.39	0.15	827.39	0.15
CMT4	150	1028.42	1048.33	1.94	1046.90	1.80	1038.51	0.98	1035.71	0.71
CMT5	200	1291.45	1314.24	1.76	1341.40	3.87	1343.51	4.03	1341.71	3.89
CMT11	120	1042.11	1056.26	1.36	**1042.11**	0.00	1047.08	0.48	1046.45	0.42
CMT12	100	819.56	835.25	1.91	**819.56**	0.00	**819.56**	0.00	**819.56**	0.00
Average		834.55	841.64	0.71	840.25	0.47	840.25	0.47	839.81	0.43

TABLE II
RESULTS ON THE SECOND GROUP OF INSTANCES

Instance	Stops	BKS	SC-ESA	LNS-ACO	CVRP-FA	ILS	SA	ISA-CO	HHQL
E-n22-k4	22	375	**0.00**	**0.00**	**0.00**	**0.00**	1.60	**0.00**	**0.00**
E-n23-k3	23	569	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
E-n30-k4	30	503	**0.00**	**0.00**	0.59	**0.00**		**0.00**	**0.00**
E-n33-k4	33	835	0.48	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
E-n51-k5	51	521			**0.00**		**0.00**	**0.00**	**0.00**
E-n76-k7	76	682	2.05	1.91	0.15	0.44	2.20	0.15	**0.00**
E-n76-k8	76	735	1.09	1.22		0.27	2.40	**0.00**	**0.00**
E-n76-k10	76	830			0.60		**0.00**	0.36	0.12
E-n76-k14	76	1021	**0.00**	0.88	0.78	1.76	2.45	0.29	**0.00**
E-n101-k8	101	815					2.21	0.25	**0.00**
E-n101-k14	101	1067	1.41	1.41		1.03		1.31	0.75
Average			0.63	0.68	0.27	0.44	1.21	0.21	0.08

TABLE III
RESULTS OF THE BASIC SBRP

Instance	Stops	ILS		VNS		HHQL		Gap(%)	
		TD	AvgTD	TD	AvgTD	TD	AvgTD	G_1	G_2
C01	70	363.00	370.11	363.17	370.33	362.71	363.93	0.08	0.13
C02	35	245.58	245.60	245.58	249.15	245.58	245.58	0.00	0.00
C03	30	202.98	203.45	202.98	204.29	202.19	202.19	0.39	0.39
C04	23	143.22	143.33	143.22	143.31	143.22	143.22	0.00	0.00
C05	75	385.68	392.79	386.49	391.25	385.67	386.17	0.00	0.21
C06	17	101.83	102.00	101.83	101.98	101.83	101.83	0.00	0.00
R01	38	185.73	190.88	185.77	186.22	185.73	185.73	0.00	0.02
R02	40	193.18	194.62	193.33	195.44	193.18	193.18	0.00	0.08
R03	51	214.92	217.22	214.91	219.01	214.91	215.48	0.00	0.00
R04	35	215.22	218.82	215.22	219.03	215.22	215.22	0.00	0.00
R05	42	189.67	192.35	189.67	190.55	189.29	189.29	0.20	0.20
R06	44	186.30	194.34	186.99	190.98	185.77	185.77	0.29	0.65
Average	41.67	218.94	222.13	219.10	221.79	218.78	218.97	0.08	0.14

TABLE IV
RESULTS OF OPEN BI-OBJECTIVE SBRP

Instance	ILS		VNS		HHQL	
	N	TD	N	TD	N	TD
C01	16	194.52	16	194.90	16	194.27
C02	12	126.66	12	126.66	12	126.66
C03	9	104.82	10	104.22	9	104.82
C04	7	74.04	7	74.04	7	74.04
C05	18	203.65	18	203.65	18	203.65
C06	6	53.38	6	53.49	6	53.38
R01	9	103.19	10	97.19	9	103.19
R02	9	104.81	9	107.02	9	104.81
R03	13	119.78	13	118.31	13	118.31
R04	10	115.59	10	115.59	10	115.59
R05	9	102.24	9	102.24	9	102.24
R06	9	101.42	9	101.81	9	100.42
Average	10.58	117.01	10.75	116.59	10.58	116.78

TABLE V
RESULTS OF OPEN SBRP WITH ONE OPTIMIZATION OBJECTIVE

Instance	ILS	VNS	HHQL	$G_1(\%)$	$G_2(\%)$
C01	194.47	198.34	194.27	0.10	2.05
C02	126.66	126.74	126.66	0.00	0.07
C03	104.22	104.31	104.22	0.00	0.08
C04	74.04	74.04	74.04	0.00	0.00
C05	203.65	204.23	203.65	0.00	0.29
C06	53.54	53.44	52.29	2.33	2.14
R01	96.23	97.71	96.23	0.00	1.51
R02	103.71	103.71	103.71	0.00	0.00
R03	118.31	120.24	117.96	0.30	1.90
R04	115.59	115.59	115.59	0.00	0.00
R05	102.24	101.13	101.13	1.09	0.00
R06	101.81	99.83	99.83	1.95	0.00
Average	116.21	116.61	115.80	0.48	0.67

indicate the improve percentage of HHQ compared with ILS and VNS algorithms respectively.

As reported in TABLE III, the HHQL algorithm has the lowest average total travel distance among of three algorithms. Compared with ILS and VNS, the HHQL algorithm decreased the total travel distance by 0.08% and 0.14% on average. For 12 instances, the HHQL algorithm finds 12 best solutions, while the ILS algorithm and VNS algorithm get 6 and 4 best solutions respectively. Furthermore, for the HHQL algorithm, there are 8 instances whose best solution is equals to the average solution. It can be seen that HHQL is very stable.

Further, we used our algorithm to solve the open SBRP, in which each bus need not return to the school. There are two kinds of solutions for open SBRP. One is to first minimize the number of routes and then reduce the total travel distance in the process of optimization. It means that the routes number objective has higher priority than the total travel distance objective. The other is to directly obtain the solution with the lowest total travel distance. Thus, we carried out experiments on these two cases respectively.

TABLE IV and TABLE V give the results of three algorithms on two open SBRP problems. Columns *N* and *TD* represent the total route number and total travel distance respectively. Columns G_1 and G_2 denote the improvement percentage of the HHQL algorithm compared with the ILS algorithm and the VNS algorithm respectively.

We can have some findings from the TABLE IV and TABLE V. For an open bi-objective SBRP, the HHQL algorithm and ILS algorithm can both find best total route number when the total route number objective is the first optimization objective. The average total route number of them are both 10.58, which is better than that of the VNS algorithm. The HHQL algorithm finds lower total travel distance on average than the ILS algorithm, when they have the same route numbers. Thus, the HHQL algorithm outperforms the ILS algorithm and VNS algorithms. For an open SBRP just considering the total travel distance optimization objective, the HHQL algorithm still has the lowest average total travel distance. Compared with the ILS algorithm and the VNS algorithm, the HHQL algorithm improves by 0.48% and 0.67% respectively. For instance C06, the maximum improvement percentages are separately 2.33% and 2.14%. All in all, these results on two tables demonstrate the effectiveness and performance of our proposed algorithm.

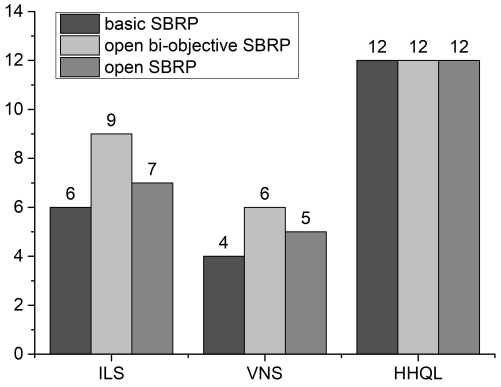
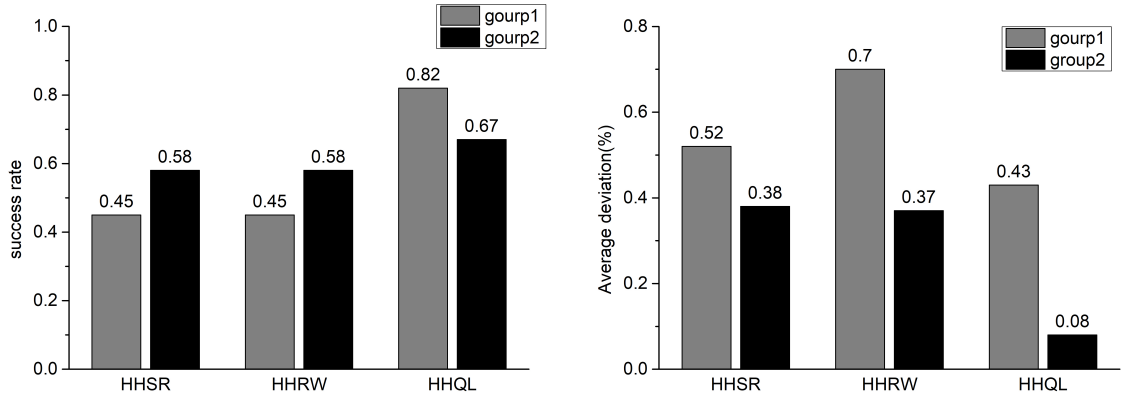


Fig. 3. the number of best solutions found by three algorithms

Further, we calculated the number of best solutions obtained by three algorithms on three types of SBRP problems. The results are shown in Fig. 3. Seen from the Fig. 3, the HHQL finds all the best solutions of three SBRP problems and the success rates are 1. While for ILS and VNS, the ILS algorithm is better than the VNS algorithm, whose maximal success rate is 0.75. In total, the HHQL algorithm is very effective.

D. Performance Analysis of Q-learning Mechanism

In this section, we evaluate the performance of Q-learning mechanism used in our proposed algorithm. First, we constructed two algorithms named as HHSR and HHRW, which employed random selection strategy and roulette wheel selection strategy to select low-level heuristics respectively. The two algorithms differed from the HHQL algroithm only in the selection strategy. Second, we conducted an experiments on two CVRP instance sets,including the instances (denoted as group1) in TABLE I and those (denoted as group2) in TABLE II. Finally, we used these two algorithms to solve



(a) success rate

(b) average deviation

Fig. 4. success rate and average deviation of three algorithms on two groups

TABLE VI
COMPARISON OF THREE ALGORITHMS ON DIFFERENT INDICATORS

Problem Type	Indicator	HHSR	HHRW	HHQL
basic SBRP	ATD	219.24	219.19	218.78
bi-objective open SBRP	AN	10.83	10.83	10.58
	ATD	116.37	116.43	116.78
open SBRP	ATD	115.93	11.5.94	115.8

three types of SBRP problems and recorded the experiment results.

Fig. 4 shows the success rate and average deviation of three algorithms on two instances sets. As shown in Fig. 4, the HHQL algorithm has the highest success rate, which are 0.82 and 0.67 respectively. The HHSR algorithm and HHRW algorithm have the same success rate, which represents that they have the same ability to find the best-known solutions. While for the average deviation, the HHQL algorithm also outperforms other two algorithms.

Furthermore, we adapted average route number (AN) and average total distance (ATD) two indicators to show the performance of these three algorithms on three types of SBRP problems. The results are shown in TABLE VI. As reported in TABLE VI, the HHQL still outperforms the HHSR algorithm and HHRW algorithm. For basic SBRP problem and open SBRP problem, the HHQL algorithm finds the best average total distance. While for bi-objective open SBRP problem, the HHQL algorithm finds the best average route number. For three types of SBRP problems, the HHQL algorithm is more competitive than other algorithms.

In general, the HHQL algorithm uses the Q-learning algorithm to select the best suitable low-level heuristic at each iteration, which can take advantage of the historical performance of them. Meanwhile, the Q-learning algorithm also directs the hyper-heuristic algorithm toward the direction of the best solution avoiding uncertainty and randomness in the search of the algorithm.

VI. CONCLUSION

This paper proposed a selection hyper-heuristic (HHQL) for single-school SBRP with two different problem characteristics. The proposed algorithm adapted a Q-learning algorithm, which is a kind of reinforcement learning algorithm, as high-level strategy to manage a set of problem-dependent low-level heuristics. The low-level heuristic, which is considered as an action, with maximum reward will be chosen to improve the solution at the stage of optimization process. The HHQL is first used to solve the CVRP, and the experiment results show that HHQL outperforms other state-of-the-art CVRP algorithms. For two single-school SBRP problems, the HHQL algorithm also has better performance compared with existing SBRP metaheuristics. Additionally, the experiment results also reveal that the Q-learning-based selection strategy used in the proposed algorithm is more competitive than random selection and roulette wheel selection strategies.

In future, we will do more works to improve the effectiveness of our proposed algorithm. At the same time, we will extend the proposed algorithm to other complex SBRP problems.

REFERENCES

[1] J. Park and B. I. Kim, "The school bus routing problem: A review," *European Journal of Operational Research*, vol. 202, no. 2, pp. 311-319, 2010.

[2] R. M. Newton and W. H. Thomas, "Design of school bus routes by computer," *Socio-Economic Planning Sciences*, vol. 3, no. 1, pp. 75-85, 1969.

[3] W. A. Ellegood, S. Solomon, J. North and J. F. Campbell, "School bus routing problem: Contemporary trends and research directions," *Omega*, vol. 95, 102056, 2020.

[4] L. X. Dang, Y. E. Hou, Q. S. Liu and Y. F. Kong, "A Hybrid Metaheuristic Algorithm for the Bi-objective School Bus Routing Problem ," *IAENG International Journal of Computer Science*, vol. 46, no. 3, pp. 409-416, 2019.

[5] Y. E. Hou, L. X. Dang, Z. Y. Wang and Y. F. Kong, "A metaheuristic algorithm for routing school buses with mixed load," *IEEE Access*, vol. 8, pp. 158293-158305, 2020.

[6] F. M. Lima, D. S. Pereira, S. V. Conceição and N. T. Nunes, "A mixed load capacitated rural school bus routing problem with heterogeneous fleet: Algorithms for the Brazilian context," *Expert Systems with Applications*, vol. 56, pp. 320-334, 2016.

[7] O. Ünsal and T. Yigit, "Using the genetic algorithm for the optimization of dynamic school bus routing problem," *BRAIN Broad Research in Artificial Intelligence and Neuroscience*, vol. 9, no. 2, pp. 6-21, 2018.

[8] Y. E. Hou, N. Zhao, L. X. Dang, and B. B. Liu, "A Hybrid Metaheuristic Algorithm for the School Bus Routing Problem with Multi-School Planning Scenarios," *Engineering Letters*, vol. 29, no. 4, pp. 1397-1406, 2021.

[9] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, 1997.

[10] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan and R. Qu, "Hyper-heuristics: a survey of the state of the art," *Journal of the Operational Research Society*, vol. 64, no. 12, pp. 1695-1724, 2013.

[11] J. H. Drake, A. Kheiri, E. Ö zcan and E. K. Burke, "Recent advances in selection hyper-heuristics," *European Journal of Operational Research*, vol. 285, pp. 405-428, 2020.
[12] M. Misir, K. Verbeeck, P. De Causmaecker and G. Vanden Berghe, "A new hyper-heuristic as a general problem solver: an implementation in hyflex," *Journal of Scheduling*, vol. 16, no. 3, pp. 291-311, 2013.
[13] W. Qin, Z. L. Zhuang, Z. Z. Huang and H. Z. Huang, "A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem," *Computers & Industrial Engineering*, vol. 156, 107252, 2021.
[14] Y. W. Zhao, L. L. Leng and C. M. Zhang, "A novel framework of hyper-heuristic approach and its application in location-routing problem with simultaneous pickup and delivery," *Operational Research*, vol. 21, no. 2, pp. 1299-1332, 2021.
[15] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279-292, 1992.
[16] S. S. Choong, L. P. Wong and C. P. Lim, "Automatic design of hyper-heuristic based on reinforcement learning," *Information Sciences*, no. 436-437, pp. 89-107, 2018.
[17] S. Abdullah, N. R. Sabar, M. Z. A. Narzri and M. Ayob,"An exponential monte-carlo algorithm for feature selection problems," *Computers & Industrial Engineering*, no. 67, pp. 160-167, 2014.
[18] H. B. Li, X. X. Zhang, S. Fu and Y. Y. Hu," A hybrid algorithm based on ant colony optimization and differential evolution for vehicle routing problem," *Engineering Letters*, vol. 29, no. 3, pp. 1201-1211, 2021.
[19] P. Augerat, J. M. Belenguer, E. Benavent, A. Corberán, D. Naddef and G. Rinaldi," Computational results with a branch and cut code for the capacitated vehicle routing problem," *IMAG*, vol. 34, 1995.
[20] N. Christofides and S. Eilon, "An algorithm for the vehicle-dispatching problem," *Journal of the Operational Research Society*, vol. 20, no. 3, pp. 309-318, 1969.
[21] S. Akpinar, "Hybrid large neighborhood search algorithm for capacitated vehicle routing problem," *Expert Systems with Applications*, vol. 61, pp. 28-38, 2016.
[22] Y. E. Hou, B. B. Liu, L. X. Dang, W. W. He and W. B. Gu, "A Local Search-based Metaheuristic Algorithm Framework for the School Bus Routing Problem," *Engineering Letters*, vol. 30, no. 1, pp. 17-28, 2022.
[23] M. Stanojević, B. Stanojević and M. Vujošević, "Enhanced savings calculation and its applications for solving capacitated vehicle routing problem," *Applied Mathematics and Computation*, vol. 29, no. 20, pp. 10302-10312, 2013.
[24] M. A. Asma, M. M. Abdulqader and G. Abdullatif, "An improved hybrid firefly algorithm for capacitated vehicle routing problem," *Applied Soft Computing*, vol. 84, 105728, 2019.
[25] İ. Ïlhan, "A population based simulated annealing algorithm for capacitated vehicle routing problem," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 28, no. 3, pp. 1217-1235, 2020.
[26] İ. Ïlhan, "An improved simulated annealing algorithm with crossover operator for capacitated vehicle routing problem," *Swarm and Evolutionary Computation*, vol. 64, 100911, 2021.
[27] P. Hansen, N. Mladenovic, "Variable neighborhood search: principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449-467, 2001.