# A Two-stage Selection Hyper-heuristic Algorithm for the Capacitated Vehicle Routing Problem

Yan-e Hou, Wenwen He, Congran Wang and Xinhui Ren

***Abstract*—As a classical combinatorial optimization problem, capacitated vehicle routing problem (CVRP) has been continuously studied. However, developing effective approaches for CVRP remains very challengeable. This article presented a two-stage selection hyper-heuristic algorithm for CVRP. The proposed algorithm used the multi-armed bandit algorithm as the high-level selection strategy, which selected a low-level heuristic at each step in the iterative local search framework according to the rewards that it had obtained. For the new neighborhood solution that was found by the selected low-level heuristic, a hybrid acceptance strategy was employed to decide whether to accept it. In addition, we also designed a low-level heuristic based on the ruin-and-recreate principle to expand the search space of neighborhood solutions except for the regular neighborhood operators widely used in CVRP. Moreover, the routes of better solutions were also put into a pool of routes. Then a set partitioning procedure model was built based on these routes and then solved by CPLEX to obtain better solution. Computational results on 82 instances demonstrate that the presented algorithm is a satisfactory and competitive approach for CVRP.**

***Index Terms*—hyper-heuristic; vehicle routing problem; multi-armed bandit; set partitioning; heuristics.**

## I. Introduction

VEHICLE routing problem (VRP) is a classical combinatorial optimization problem, which has a very extremely wide range of applications. Since it was put forward by [1], scholars have been continuously studying the vehicle routing problems. Capacitated vehicle routing problem (CVRP) is the basic problem of VRP. The goal of CVRP is to find a set of routes with lowest travel cost to meet the demands of the customers while satisfying with the vehicle capacity constraints.

CVRP is an extremely difficult problem, which means that the optimal solution of it cannot be found in polynomial time. Thus, different kinds of algorithms are proposed to solve CVRP. Exact algorithms can solve relatively small problems, and the commonly used exact algorithms include column generation, dynamic programming, branch and cutting and so on. The recent review of exact algorithms for CVRP can found in some literatures [2],[3],[4]. Exact algorithms can find optimal solution, but they spend more computation time. It is the reason that many researchers prefer to use heuristics, metaheuristics or hybrid algorithms. These algorithms can find an approximate optimal solution in a limited time. There are a huge number of researches about these algorithms for CVRP [5],[6],[7]. The recent review of CVRP on these algorithms could be found in the literatures [8],[9],[10],[11].

Although there are a lot of approaches for CVRP, developing effective approaches for CVRP is still a very challenging job. As a basic problem of VRP, the developed methods for the CVRP may be extended to solve its other variants. Meanwhile, the methods with the less troublesome procedure of parameters setting and tuning will be more attractive.

Hyper-heuristic is a kind of heuristic or metaheuristic algorithm that guides or generates heuristic algorithms to tackle difficult optimization problems [12],[13]. It uses a high-level strategy (HLS) to guide a set of low-level heuristics (LLH), which are related to the specific problem-solving domain. The HLS is responsible for managing or manipulating several LLHs to solve the problem. Therefore, the hyper-heuristic algorithms cannot only solve different problems across domains [12],[14],[15], but also solve many variants of a class of problems [16],[17],[18]. Hyper-heuristic algorithm has the advantages of having fewer parameters, simple parameter setting, easy design and implementation.

In this study, we try to develop a two-stage hyper-heuristic algorithm for CVRP, due to the advantages of the hyper-heuristic algorithm. In the first stage, we used the multi-armed bandit (MAB) algorithm based on Upper Confidence Bound (UCB) [19] to select the proper LLH. For each new obtained neighborhood solution by the chosen LLH, the record-to-record travel method [20] or naive acceptance rule will be chosen randomly to accept or refuse it. Moreover, the routes of neighborhood solutions were also recorded into a route pool. When the global best solution found in the first stage was not the best-known solution, the routes in the route pool was used to build a set partitioning procedure (SP) model. Then the SP model was solved by CPLEX 12.6 optimization software to get a better solution. The presented algorithm was tested on 82 standard CVRP instances. The results prove that our proposed algorithm has a good performance. Meanwhile, the proposed algorithm is also effective when compared with the state-of-the-art algorithms.

The structure of this article is organized as follows. Section II describes the definition of CVRP. The related works about CVRP and hyper-heuristic are given in Section III. The designed algorithm in this paper is described in Section IV. In section V, the experimental study is carried out, and the comparison results are given. Section VI gives the conclusion and indicates the future research direction.

Manuscript received February 19, 2022; revised August 29, 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)

Wenwen He is a graduate student of College of Computer and Information Engineering, Henan University, Kaifeng, 475004, China (e-mail: ww82323@henu.edu.cn)

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

Xinhui Ren is a lecturer of College of Computer and Information Engineering, Henan University, Kaifeng, 475004, China (corresponding author, e-mail: henu_rxh@126.com)

## II. Problem Definition

The CVRP can be described in the following. There is a single depot and some vehicles. There are several customers with a known number of demands, which will be served only once by a vehicle. Every vehicle starts from the depot, serves the customers and then returns to the depot. The objective of CVRP is to find a set of routes that minimize the total traveled distance by all the vehicles.

The CVRP is usually defined as a complete graph $G = (V, A)$. $V = \{0, 1, 2, ..., n\}$ is the node set, and $A = \{(i, j)|i, j \in V\}$ is the arc set. Node 0 indicates the depot. A set of $K$ homogeneous vehicles are available at the depot, each of them has the same capacity $Q$. The customers nodes to be served are denoted as $U = \{1, 2, 3, ..., n\}$. For every customer node $i$ (i$\in U$), it has a certain demand $q_i$. At the same time, $q_0 = 0$. The travel distance between node $i$ and node $j$ is represented by $c_{ij}$. Each customer node must be served only once by the vehicle. Whenever the total demand of customer nodes served by the same vehicle must not be larger than the vehicle capacity $Q$. We use binary decision variable ($x_{ijk}$) to indicate if a vehicle $k$ passes through directly node $i$ and then node $j$. When $x_{ijk} = 1$, it means that the vehicle $k$ directly visits node $j$ after visiting node $i$, otherwise $x_{ijk} = 0$. The mathematical problem of CVRP is defined as follows.

Minimize

$$Z = \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=1}^{K} c_{ij}x_{ijk} \tag{1}$$

Subject to:

$$\sum_{i=0}^{n}\sum_{k=1}^{K} x_{ijk} = 1, \forall j \in U, k \in \{1, 2, ...K\}, i \neq j \tag{2}$$

$$\sum_{j=0}^{n}\sum_{k=1}^{K} x_{ijk} = 1, \forall i \in U, k \in \{1, 2, ...K\}, i \neq j \tag{3}$$

$$\sum_{i=0}^{n}\sum_{j=0}^{n} x_{ijk}q_i \leq Q, \forall i, j \in U, k \in \{1, 2, ...K\} \tag{4}$$

$$\sum_{j=0}^{n} x_{0jk} = \sum_{i=0}^{n} x_{i0k} = 1, \forall k \in \{1, 2, ...K\} \tag{5}$$

$$x_{ijk} \in \{0, 1\} \tag{6}$$

The objective function (1) minimizes the total distances of all routes. Constraints (2) and (3) guarantee that every customer can be visited only once by one vehicle. Equation (4) ensures that the total demands of all the visited nodes by the same vehicle does not exceed the vehicle capacity. Constraint (5) indicates that every vehicle must start from the depot and end the depot. Equation (6) gives the value of the decision variable $x_{ijk}$.

## III. Related Works

This section will present the related works to the proposed study. First, the algorithms that combine set partitioning procedure for the CVRP and its variants are briefly described. Then, the overview of hyper-heuristic algorithm used in CVRP is shown.

### A. Set Partitioning-based Approaches

Among the solving approaches of CVRP, exact algorithm can find the optimal solution in theory. Although it has limitations in computation time and problem scale, it still attracts researchers to exploit the successful exact approaches for CVRP. Set partitioning is one of the mathematical formulations of the CVRP, which was originally proposed by [21]. The set partitioning-based exact approaches have been summarized in [2] and [3].

The set partitioning procedure could be applied to recombine routes that are generated by a heuristic. There also emerge some successful set partitioning-based hybrid metaheuristics for CVRP and its variants. As early as in 1995, the set partitioning procedure was considered as a post-optimization technique to improve the quality of the tabu algorithm [22]. After that, a genetic algorithm hybrid with SP was proposed by [23] for the CVRP with time window. The genetic algorithm was firstly used to generate multiple routes, and then SP model was accurately solved. And then the iterated local search (ILS) metaheuristic mixed with SP was developed for the heterogeneous VRP [24], which only the routes of the local best solutions were used to construct the SP model. Then the authors extended this hybrid approach to solve a class of VRP problems [5]. Several experiments proved that their hybrid approach had better performance than the state-of-the-art algorithms for CVRP. Beyond that, some works took the set partitioning procedure as inside hybrid heuristics [25],[26],[27].

The set partitioning-based approaches can be applied to the special variants of VRP. A special variant of heterogeneous fleet VRP in the context of hazardous materials transportation addressed was addressed, and a variable neighborhood search (VNS) was developed [28]. Then SP problem was taken as a post-improvement procedure, and then was solved on the routes generated in the local search procedure of VNS. Additionally, the hybrid algorithm combing ILS and SP was proposed to solve school bus routing problem with different problem characteristics and application scenarios in our previous works [29],[30],[31].

The successfully experience of mixing SP with the metaheuristic algorithm reveals that it can take advantages of exact algorithms and enhance effectively the quality of the algorithm.

### B. Hyper-heuristics for VRP

In this section, we focus on the hyper-heuristics that are specially designed for CVRP and its variants, excluding those designed for cross-domain problems including VRP [13].

Garrido and Castro [32] proposed a hyper-heuristic to solve CVRP, which used some LLHs to construct and improve the partial solutions in the framework of a hill climbing method. Further, the authors designed an evolutionary hyper-heuristic for the dynamic VRP [33]. Marshall et al. [34] constructed forty-eight combinations by six selection methods and eight acceptance criteria to compare their performance over randomly generated instances of CVRP. Experiment results showed that the combination of simulated annealing and naive acceptance had the best performance. Sabar et al. [35] proposed a math-hyper-heuristic for CVRP with time windows, which adopted column generation to construct

**Algorithm 1** HHMAB-SP

**Input:** maximum iteration $maxiter$, neighborhood list size $nb$, scaling factor $C$, deviation factor $dev$, destruction factor $p$ and maximum trail number $iter$
**Output:** the best solution $S$

1: $S_c$ = $S$ = Null; $RoutePool$ = Null;
2: $S_c$ = GenerateInitialSolution();
3: AddRoutesToRoutePool($S_c$, $RoutePool$);
4: $LLH\_List$ = InitialLLH();
5: **while** Not meet the stop condition **do**
6: **if** there exists a LLH that has not been used **then**
7: $LLHs$ = SelectByRandom($LLH\_List$);
8: **else**
9: $LLHs$ = SelectByUCBMAB($LLH\_List$,$C$);
10: **if** $LLHs$ is ruin-and-recreate operator **then**
11: $S_n$ = AppliedLLH($S_c$,$LLHs$,$p$,$iter$);
12: **else**
13: $S_n$ = AppliedLLH($S_c$,$LLHs$,$nb$);
14: AddRoutesToRoutePool($S_n$, $RoutePool$);
15: **if** AcceptanceRule($S_c$,$S_n$,$dev$) **then**
16: **if** $S_n$ is better than $S$ **then**
17: $S$ = $S_n$;
18: $S_c$ = $S_n$;
19: **if** $S$ is not the best-known solution **then**
20: $SP\_M$ = BuildSPModel($RoutePool$);
21: $S_p$ = SolveModelByCPLEX($SP\_M$);
22: **if** $S_p$ is better than $S$ **then**
23: $S$ = $S_p$;
24: **return** $S$

an initial solution and then used several LLHs improve the solution. The algorithm employed a MAB selection approaches and exponential Monte Carlo move acceptance rules. The hybrid math-hyper-heuristic was proved to be effective for large-scale VRPTW instances. Qin et al. [36] developed a hyper-heuristic for heterogeneous VRP that is based on reinforcement learning method, and the algorithm was first evaluated on some well-known CVRP instances and then used to solve the heterogeneous VRP.

## IV. Methodology

### A. Overall Procedure of Proposed Hyper-heuristic

In the framework of traditional selection hyper-heuristic, two consecutive processes will be executed iteratively until meeting the terminal condition. These two processes are the selection of LLHs and move acceptance rules, which decides whether accept a new solution found by the selected LLH. Therefore, the selection strategy of LLHs and acceptance mechanism are the key parts of the design of hyper-heuristic. In our hyper-heuristic algorithm (named HHMAB-SP), we used multi-armed bandit based on UCB as selection strategy and a nondeterministic acceptance rule as acceptance rule. In the second stage, a SP model was established and then solved by CPLEX to find a better solution. Algorithm 1 describes the outline of the HHMAB-SP algorithm.

The next is to describe Algorithm1 in detail. Step (1)∼(4) do some initialization works, which include initialization of the variables, solution and the list of LLHs. And then the routes of initial solution are put into a pool of routes. The main body of Algorithm in the first stage is step (5)∼(18). When the loop number does not reach *maxiter* or the immediate solution is not the best-known solution, the main procedure of the algorithm will be executed iteratively until it meets the stop condition. The selection of low-level heuristic is defined in Step (6)∼(9). When there exists one unused LLH, a LLH is chosen randomly. If all the LLHs have been used, the LLH in the next will be selected by multi-armed bandit method in Step (9). Step (10)∼(13) apply the chosen LLH to find a new neighborhood solution $S_n$. The routes of the new solution $S_n$ are put into the routes pool in Step (14), and $S_n$ is evaluated by the acceptance rules in Step (15). When $S_n$ is accepted, the global best solution $S$ will be modified in Step (16)∼(17). Then, the new solution $S_n$ participates in the next loop as the current solution $S_c$ in Step (18). When the first stage finishes, the global best solution $S$ is gotten. The second stage of the proposed algorithm is described in Step (19)∼(23). If $S$ is not the best-known solution, a SP model will be built and solved to get a new solution $S_p$. If $S_p$ is better than $S$, $S$ will be replaced by $S_p$. When the algorithm terminates, the global best solution $S$ returns.

### B. Multi-armed Bandit Selection Strategy

For the proposed selection hyper-heuristic, the high-level selection strategy selects one low-level heuristic from a set of LLHs. The HLS needs to decide which LLH is to be selected. The selection chance of each LLH is measured by its performance under assessment. The selection of LLH can be regarded as a problem of adaptive operator selection. For this problem, there are some promising approaches such as probability matching method [37], adaptive pursuit strategy [38], and multi-armed bandit [19],[35]. The issued problem can also be considered as a problem of the balance between exploitation and exploration. On one hand, it should exploit the operators that are often used. On the other hand, it should give chance to the poor operators that may be better in the future. Several MAB related approaches have been developed to deal with this problem [39],[40].

In our proposed hyper-heuristic algorithm, we use an UCB-based MAB as the selection mechanism. Each LLH can be regarded as an arm in the MAB problem. At each time $t$, the low-level heuristic, which makes the function defined in Equation (7) have maximal value, will be selected.

$$h(t) = w_{i,t} + C \times \sqrt{\frac{2 \times ln \sum_{j=1}^{N} n_{j,t}}{n_{i,t}}} \tag{7}$$

Where the first component $w_{i,t}$ is an empirical reward of $i^{th}$ low-level heuristic obtained from the beginning to the time $t$. The second one is an upper confidence bound that depends on the number of times $n_{i,t}$, which the $i^{th}$ low-level heuristic has been used so far. These two components are related exploitation and exploration respectively. The number of low-level heuristics is $N$. The parameter $C$ is a scaling factor, which controls the tradeoff between exploitation and exploration. When the parameter $C$ is smaller, it prefers to select the LLH that has the best reward of all the LLHs. Otherwise, it will tend to choose the LLH that is applied

infrequently. The empirical rewards $w_{i,t}$ is calculated by Equation (8).

$$w_{i,t+1} = \frac{n_{i,t-1} \times w_{i,t} + r_{i,t}}{n_{i,t}} \tag{8}$$

Where $r_{i,t}$ is the score of the $i^{th}$ low-level heuristic up to time $t$.

### C. Credit Score Assignment

The goal of high-level selection strategy is to select the best appropriate low-level heuristic for the current solution in the process of search. Thus, every low-level heuristic must be evaluated whether it is suitable for the current solution.

For every low-level heuristic, the credit score assignment defines the reward based on its recent performance in the search process. The most common approach is to directly use the improvements values of the objective, which is obtained by the recently used low-level heuristic. However, it is not suitable to evaluate the low-level heuristic in the overall search process. At the beginning of the algorithm, the improvement value of objective is relatively high. As the algorithm executes, it will gradually decrease. Therefore, the improvement target value does not accurately reflect the performance of the low-level heuristic.

To this end, we use the improvement rate of objective value to evaluate the low-level heuristic. At time $t$, $S_c$ is the current solution and its objective value is denoted as $f(S_c)$. After a low-level heuristic $i$ is applied to $S_c$, a new solution $S_n$ is gotten and its objective value changes to $f(S_n)$. The score of the low-level heuristic $i$ at time $t$ is calculated by Equation (9).

$$r_{i,t} = \frac{f(S_c) - f(S_n)}{f(S_c)} \times 100 \tag{9}$$

### D. Acceptance Criterion

For a new neighborhood solution found by the selected low-level heuristic, acceptance rule must be used to accept or refuse it. We use a hybrid acceptance rule that consists of record-to-record travel and naive acceptance to make a decision. It has the advantage of bringing about more variety of solutions by allowing accepting less worsening solutions. By this acceptance mechanism, a new solution that is superior to the original solution is always accepted, because the new solution is improved by the selected LLH. For the worsening solution, these two acceptance rules will be randomly selected to decide to whether accept it or not. For naive acceptance, the worse solution will be accepted by 50% probability. While for the record-to-record travel acceptance, the worsening solution $S$ will be accepted when Equation (10) is true:

$$f(S) < (1 + dev) \times record, dev \in [0, 1] \tag{10}$$

Where $S$ is the new solution and its objective value is $f(S)$, variable $record$ indicates the objective value of current best solution, and parameter $dev$ is the deviation factor between 0 and 1. The value of $record$ will change as the procedure of local search.

### E. Low-Level Heuristics

The low-level heuristics used in the proposed algorithm consist of six regular neighborhood operators, which are often used in VRP algorithm, and one operator based on the ruin-and-recreate principle [41]. When each LLH is executed, the obtained new solution must be a feasible solution. That is to say, every moved must not violate all the constraints.

The low-level heuristics are described in the following.

(1) $\text{LLH}_1$: Choose a customer node at random and move it to another position in a route. It may occur at the same route or between two different routes.

(2) $\text{LLH}_2$: Exchange two customer nodes from two different routes. It is also regarded as the swapping of two points.

(3) $\text{LLH}_3$: Select two non-adjacent edges in the same route and the nodes between two edges reverse to form a new route. It is known as 2-opt.

(4) $\text{LLH}_4$: Select two edges from two different routes and cross the nodes between the edges.

(5) $\text{LLH}_5$: Select two or four consecutive nodes from one route and then shift them to the other route.It is also called or-opt.

(6) $\text{LLH}_6$: Exchange two edges from two different routes.

(7) $\text{LLH}_7$: It is a neighborhood operator based on the ruin-and-recreate principle. Several nodes from the current solution are first chosen and removed. Then, they are tried to inserted into the partial solution to create a new solution. It should be pointed out that the number of destroyed nodes is controlled by the destruction factor $p$. The value of destruction factor must be appropriate to keep the balance between quality and performance. The operations of this LLH include destruction and recreate two procedures. The maximum number of destroy-and-repair procedures is denoted as $iter$. First, the nodes that need to be deleted are chosen randomly from the current solution, and then these nodes are removed to generate a partial solution. Finally, each removed node will be inserted in the position with the cheapest insertion cost. If it cannot find the insertion position, a new route will be recreated by the removed nodes. From the point of view, the ruin-and-recreate operator may be find a better solution or perturb the current solution.

### F. Set Partitioning Procedure

In the second stage of the proposed algorithm, we will use a set partitioning procedure to enhance the performance of algorithm. We build a set partitioning procedure based on the historical routes that were put into the routes pool. The set partitioning model of CVRP can be defined as follows.

Minimize

$$\sum_{r \in P} d_r x_r \tag{11}$$

Subject to:

$$\sum_{r \in P_i} x_r = 1, \forall i \in U \tag{12}$$

$$x_r \in \{0, 1\}, \forall r \in P \tag{13}$$

Where $P$ is all the routes in the route pool, and $d_r$ is the total distance of route $r$. $U$ is the set of all customer nodes. For a customer node $i$, all the routes that owner node $i$ is denoted as $P_i$ ($P_i \subseteq P$). When decision variable $x_r = 1$, it indicates the route $r$ is in the final solution. When $x_r = 0$,

the route $r$ does not included by the final solution. Equation (11) defines the objective function that minimizes the total distances. Constraint (12) ensures the every customer node must locate in only one route. Constraint (13) indicates that the value of decision variable must be 0 or 1.

As a weak NP-hard problem, the SP model could be solved by CPLEX optimization software. To avoid consuming too much computation time, the maximum execution time of CPLEX is limited to 60 seconds. The solution obtained by the CPLEX returns and compares with the global solution found by the first stage of proposed algorithm.

## V. Experiments and Analysis

To evaluate the performance of the HHMAB-SP algorithm, we introduce four experimental studies. Firstly, the obtained results on CVRP standard instances by the HHMAB-SP algorithm are given in some tables. Then the HHMAB-SP is compared with other state-of-the-art algorithms for CVRP. Secondly, we test the performance of the proposed algorithm with different high-level selection strategies in order to prove the advantage of MAB selection strategy. Thirdly, we perform some experiments to evaluate the advantage of set partitioning procedure. Finally, we analysis the influence of the ruin-and-recreate operator in proposed algorithm.

The proposed algorithm was implemented by C# and run on the personal computer of Intel i5-9500 3.0GHz CPU and 16GB RAM. The HHMAB-SP algorithm was used to solve 82 instances from four CVRP standard instances including Set A, B and P proposed by [42] and Set E proposed by [43]. Every instance was executed independently by the HHMAB-SP algorithm 10 times.

### A. Parameters Setting

The parameters of the HHMAB-SP consist of the parameters that are used in the first stage of it and the parameters of CPLEX solver. The parameters in the first stage include maximum iteration number $maxiter$, size of neighborhood list $nb$, learning factor $C$ in MAB with upper confidence bound, deviation factor $dev$, destruction factor $p$ in ruin-and-recreate operator and the iteration numbers $iter$ of ruin and recreate procedure. In the second stage of the HHMAB-SP algorithm, a SP model may be built and solved by CPLEX 12.6. For CPLEX solver, most of parameters are set to default values, we just set the maximum execution time $max\_execute\_time$ and the tolerances of MIP $tolerances$. The parameters setting of the HHMAB-SP algorithm are shown in TABLE I.

TABLE I
PARAMETER VALUES OF THE HHMAB-SP ALGORITHM

| Parameters | values |
|---|---|
| maxiter | 1000 |
| nb | 30 |
| C | 6 |
| dev | $10^{-4}$ |
| p | 0.2 |
| iter | 20 |
| max_execute_time | 60 |
| tolerances | $10^{-6}$ |

### B. Comparison HHMAB-SP with Other Algorithms

To verify the effectiveness of the HHMAP-SP algorithm, we used it to solve four CVRP benchmark instances sets including 82 instances and then compared it with five existing successful approaches. The comparison algorithms include a heuristic named SC-ESA [44], a hybrid metaheuristic named LNS-ACO [45], a hybrid firefly algorithm(CVRP-FA) [46], a population-based simulated annealing(PSA) proposed by [47] and an improved simulated annealing with crossover operator(ISA-CO) [48]. The test results are shown in four tables, which are TABLE II, TABLE III, TABLE IV, and TABLE V respectively. In all tables, columns $Instance$ and $BKS$ denote the instance name and the best-known solution of instance. Column $Time$ is the average execution time in seconds. For the HHMAB-SP algorithm, column $Best$ is the best solution among 10 solutions. Column $Gap$ represents the deviation percentage between the best solution and the best-known solution, which is calculated by the Equation (14). For other CVRP algorithms, the deviation percentage values come from the corresponding literatures. If the deviation percentage value is null, it means that it does not be provided in the corresponding literatures.

$$Gap = \frac{Best - BKS}{BKS} \times 100 \tag{14}$$

Additionally, we also calculated the success rate, which indicates the ability of the algorithm getting the best-known solutions out of all instances. The success rate of these algorithms on four sets is shown in TABLE VI.

As shown in five tables, the proposed HHMAB-SP algorithm achieves the lowest average deviation values on sets A, P and E. Among these algorithms, the HHMAB-SP algorithm finds 26 best-known solutions in the set A and the success rate is 0.96. For PSA, it solves 14 instances and the success rate is 0.57. ISA-CO has the highest success rate among of other algorithms, which is 0.85. The set B consists of 22 instances and the results are shown in TABLE III. The HHMAB-SP algorithm gains 20 best-known solutions and the success rate is 0.91. The ISA-CO algorithm solves 21 instances and its success rate is 0.95, which is the highest success rate of other algorithms. For set E, there are 11 instances in TABLE IV. The HHMAB-SP algorithm finds 8 best-known solutions and has the highest success rate of 0.73. For other algorithms, the highest success rate is 0.55 that belongs to the ISA-CO algorithm. In TABLE V, there are 22 instances in set P and the HHMAB-SP algorithm gets the lowest deviation percentage value. The HHMAB-SP algorithm finds 20 best-known solutions and 1 new best-known solution that was provided by [46], and the success rate is 0.95.

In general, the HHMAB-SP algorithm is very effective. For instance set A, P and E, the HHMAB-SP algorithm is competitive than other approaches. For set B, the performance of HHMAB-SP algorithm almost equals to the ISA-CO algorithm. From the view of computation time, the HHMAB-SP algorithm consumes relatively less computation time. It can obtain a relative better solution in reasonable time.

TABLE II
RESULTS OF THE CVRP SET A

| Instance | BKS | SC-ESA | LNS-ACO | | CVRP-FA | PSA | | ISA-CO | | HHMAB-SP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Gap | Gap | Time | Gap | Gap | Time | Gap | Time | Best | Gap | Time |
| A-n32-k5 | 784 | 0.00 | 0.00 | 856.21 | 1.53 | 0.00 | 25 | 0.00 | 1449 | 784 | 0.00 | 3.08 |
| A-n33-k5 | 661 | 0.00 | 0.00 | 900.06 | 0.00 | 0.00 | 26 | 0.00 | 1464 | 661 | 0.00 | 0.57 |
| A-n33-k6 | 742 | 0.00 | 0.00 | 947.86 | 0.00 | 1.08 | 25 | 0.00 | 1564 | 742 | 0.00 | 4.69 |
| A-n34-k5 | 778 | 0.00 | 0.00 | 909.04 | 0.00 | | | 0.00 | 1499 | 778 | 0.00 | 4.19 |
| A-n36-k5 | 799 | 0.00 | 0.00 | 1055.60 | 0.00 | | | 0.00 | 1583 | 799 | 0.00 | 4.85 |
| A-n37-k5 | 669 | 0.00 | 0.00 | 1103.70 | 0.00 | 0.00 | 24 | 0.00 | 1609 | 669 | 0.00 | 2.92 |
| A-n37-k6 | 949 | 0.00 | 0.00 | 1113.40 | 0.00 | 2.42 | 25 | 0.00 | 1638 | 949 | 0.00 | 3.61 |
| A-n38-k5 | 730 | 0.00 | 0.00 | 1139.20 | 0.00 | | | 0.00 | 1584 | 730 | 0.00 | 2.78 |
| A-n39-k5 | 822 | 0.00 | 0.00 | 1202.80 | 0.00 | | | 0.00 | 1665 | 822 | 0.00 | 5.11 |
| A-n39-k6 | 831 | 0.00 | 0.00 | 1266.00 | 0.00 | 0.00 | 28 | 0.00 | 1696 | 831 | 0.00 | 3.51 |
| A-n44-k6 | 937 | 0.00 | 0.00 | 1567.80 | 0.00 | | | 0.00 | 1753 | 937 | 0.00 | 6.31 |
| A-n45-k6 | 944 | 0.00 | 1.48 | 1728.10 | 0.95 | 1.48 | 26 | 0.00 | 1784 | 944 | 0.00 | 7.76 |
| A-n45-k7 | 1146 | 0.00 | 0.00 | 1741.70 | 0.09 | 0.00 | 29 | 0.00 | 1775 | 1146 | 0.00 | 5.31 |
| A-n46-k7 | 914 | 0.00 | 0.00 | 1804.50 | 0.00 | 2.74 | 28 | 0.00 | 1801 | 914 | 0.00 | 2.33 |
| A-n48-k7 | 1073 | 1.03 | 1.03 | 1978.60 | 0.00 | 0.00 | 29 | 0.00 | 1845 | 1073 | 0.00 | 3.83 |
| A-n53-k7 | 1010 | 0.10 | 0.00 | 2323.70 | 0.10 | | | 0.00 | 1973 | 1010 | 0.00 | 8.36 |
| A-n54-k7 | 1167 | 0.09 | 0.00 | 2497.40 | 0.43 | | | 0.00 | 1999 | 1167 | 0.00 | 8.03 |
| A-n55-k9 | 1073 | 0.00 | 0.00 | 2771.00 | 0.09 | 0.00 | 32 | 0.00 | 2080 | 1073 | 0.00 | 7.39 |
| A-n60-k9 | 1354 | 0.07 | 0.00 | 3345.40 | 0.07 | 1.92 | 33 | 0.00 | 2128 | 1354 | 0.00 | 14.12 |
| A-n61-k9 | 1034 | 0.00 | 3.19 | 3355.70 | 0.48 | | | 0.00 | 2238 | 1034 | 0.00 | 8.27 |
| A-n62-k8 | 1288 | 0.78 | 1.55 | 3363.30 | 0.78 | | | 0.08 | 2207 | 1288 | 0.00 | 13.67 |
| A-n63-k9 | 1616 | 0.50 | 2.04 | 3651.20 | 0.87 | | | 0.62 | 2230 | 1616 | 0.00 | 18.46 |
| A-n63-k10 | 1314 | 0.08 | 1.14 | 3800.10 | 0.00 | | | 0.00 | 2255 | 1314 | 0.00 | 12.54 |
| A-n64-k9 | 1401 | 0.57 | 1.00 | 3831.80 | 1.36 | | | 0.57 | 2277 | 1401 | 0.00 | 14.52 |
| A-n65-k9 | 1174 | 0.34 | 0.94 | 3854.20 | 0.34 | 0.00 | 30 | 0.00 | 2277 | 1174 | 0.00 | 11.58 |
| A-n69-k9 | 1159 | 0.00 | 0.95 | 4460.90 | 0.26 | | | 0.00 | 2436 | 1159 | 0.00 | 14.97 |
| A-n80-k10 | 1763 | 0.74 | 2.94 | 6493.60 | 0.57 | 4.20 | 31 | 0.91 | 2704 | 1769 | 0.34 | 39.74 |
| Average | 1041.9 | 0.16 | 0.60 | 2335.66 | 0.29 | 0.99 | 27.93 | 0.08 | 1908 | 1042.15 | 0.01 | 8.61 |

### *C. Performance Analysis of MAB-based High-level Selection Strategy*

This section test the influence of different selection strategy in the proposed algorithm. We select some selection methods commonly used in the hyper-heuristic to evaluate the performance of MAB selection strategy. These selection methods include random selection (RS), roulette wheel selection (RW), probability matching (PM) and adaptive pursuit strategy (AP).

First, we built other four algorithms based on the proposed algorithm with different selection strategies. These algorithms all have the same parameters values and the same components of the HHMAB-SP algorithms. They are just only different in the selection methods. For example, the algorithm HHRS-SP represents the hyper-heuristic uses random selection strategy as high selection rule. While for probability matching and adaptive pursuit strategy, they have more parameters, such as $p_{min}$, adaptive factor $\alpha$, learning rate $\beta$. They are set to 0.1, 0.8 and 0.6 respectively.

Next, we used these algorithms to solve all the instances sets and calculated their deviation values. Fig.1 gives the average deviation value of all instance sets of these algorithms. For all instances sets, the HHMAB-SP all have the lower deviation value. The results confirm that the multi-armed bandit high-level selection strategy used in HHMAB-SP is very resultful and it enhances the algorithms ability to find better solutions.

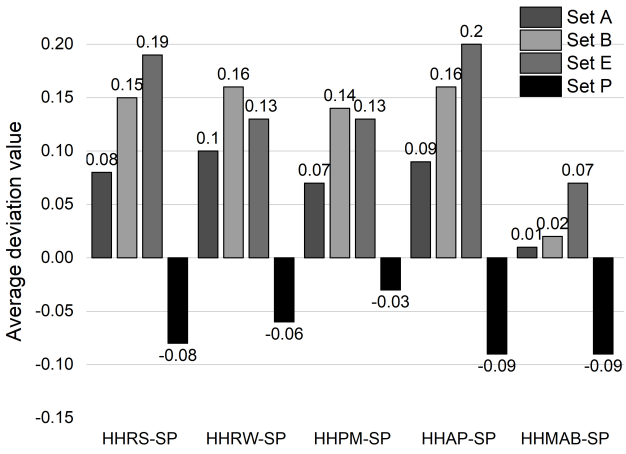


Fig. 1. Average deviation values obtained by five algorithms on four sets

Therefore, we can come to a conclusion that the proposed hyper-heuristic algorithm with MAB selection strategy is more competitive than that with other selection strategy.

### *D. Performance Analysis of Set Partitioning Procedure*

To evaluate the performance of set partitioning procedure, we designed one experiment. On one hand, the experiment was conducted to verify the influence of SP on quality of the

TABLE III
RESULTS OF THE CVRP SET B

| Instance | BKS | SC-ESA | LNS-ACO | | CVRP-FA | PSA | | ISA-CO | | HHMAB-SP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Gap | Gap | Time | Gap | Gap | Time | Gap | Time | Best | Gap | Time |
| B-n31-k5 | 672 | 0.00 | 0.00 | 828.20 | 0.00 | 0.00 | 26 | 0.00 | 1441 | 672 | 0.00 | 7.83 |
| B-n34-k5 | 788 | 0.00 | 0.00 | 908.51 | 0.00 | 0.00 | 26 | 0.00 | 1569 | 788 | 0.00 | 2.02 |
| B-n35-k5 | 955 | 0.84 | 0.00 | 998.94 | 0.00 | | | 0.00 | 1527 | 955 | 0.00 | 2.66 |
| B-n38-k6 | 805 | 1.24 | 0.00 | 1219.90 | 0.12 | 1.86 | 27 | 0.00 | 1632 | 805 | 0.00 | 5.93 |
| B-n39-k5 | 549 | 0.18 | 0.00 | 1241.00 | 0.18 | 0.00 | 27 | 0.00 | 1662 | 549 | 0.00 | 0.57 |
| B-n41-k6 | 829 | 4.46 | 0.00 | 1392.10 | 0.00 | 0.24 | 26 | 0.00 | 1683 | 829 | 0.00 | 3.02 |
| B-n43-k6 | 742 | 0.54 | 0.00 | 1502.00 | 0.00 | 0.00 | 27 | 0.00 | 1757 | 742 | 0.00 | 4.38 |
| B-n44-k7 | 909 | 1.32 | 0.00 | 1623.30 | 0.00 | 3.08 | 28 | 0.00 | 1780 | 909 | 0.00 | 3.19 |
| B-n45-k5 | 751 | 0.00 | 0.00 | 1621.70 | 0.00 | 0.00 | 27 | 0.00 | 1829 | 751 | 0.00 | 4.07 |
| B-n45-k6 | 678 | 1.18 | 0.00 | 1657.70 | 1.18 | 0.00 | 27 | 0.00 | 1771 | 678 | 0.00 | 7.84 |
| B-n50-k7 | 741 | 0.00 | 0.00 | 2174.80 | 0.00 | 1.21 | 28 | 0.00 | 1969 | 741 | 0.00 | 0.21 |
| B-n50-k8 | 1312 | 1.30 | 0.53 | 2169.20 | 0.46 | 3.51 | 28 | 0.00 | 1989 | 1312 | 0.00 | 67.79 |
| B-n51-k8 | 1016 | 0.00 | 0.00 | 2228.50 | 0.00 | | | | | 1016 | 0.00 | 8.87 |
| B-n52-k7 | 747 | 0.67 | 0.00 | 2289.30 | 0.00 | | | 0.00 | 2090 | 747 | 0.00 | 1.59 |
| B-n56-k7 | 707 | 0.00 | 0.00 | 2709.80 | 0.28 | 0.00 | 30 | 0.00 | 2181 | 707 | 0.00 | 9.92 |
| B-n57-k9 | 1598 | 0.13 | 0.00 | 3055.80 | 0.75 | | | 0.00 | 2076 | 1598 | 0.00 | 69.15 |
| B-n63-k10 | 1496 | 2.81 | 1.20 | 3750.90 | 0.47 | | | 0.00 | 2269 | 1496 | 0.00 | 10.94 |
| B-n64-k9 | 861 | 0.00 | 1.51 | 3834.60 | 0.12 | | | 0.00 | 2322 | 863 | 0.23 | 13.52 |
| B-n66-k9 | 1316 | 1.90 | 1.06 | 4242.60 | 0.23 | 0.00 | 35 | 0.00 | 2327 | 1316 | 0.00 | 68.26 |
| B-n67-k10 | 1032 | 1.74 | 1.74 | 4523.00 | 0.97 | 2.91 | 33 | 0.00 | 2340 | 1032 | 0.00 | 16.74 |
| B-n68-k9 | 1272 | 1.57 | 1.42 | 4395.20 | 0.47 | 0.00 | 33 | 0.16 | 2349 | 1276 | 0.31 | 47.09 |
| B-n78-k10 | 1221 | 2.05 | 0.57 | 6049.20 | 0.25 | 2.38 | 32 | 0.00 | 2630 | 1221 | 0.00 | 61.55 |
| Average | 954.4 | 1.00 | 0.37 | 2485.59 | 0.25 | 0.95 | 28.75 | 0.01 | 1962 | 954.68 | 0.02 | 18.96 |

TABLE IV
RESULTS OF THE CVRP SET E

| Instance | BKS | SC-ESA | LNS-ACO | | CVRP-FA | PSA | | ISA-CO | | HHMAB-SP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Gap | Gap | Time | Gap | Gap | Time | Gap | Time | Best | Gap | Time |
| E-n22-k4 | 375 | 0.00 | 0.00 | 447.39 | 0.00 | 1.60 | 22 | 0.00 | 1223 | 375 | 0.00 | 1.72 |
| E-n23-k3 | 569 | 0.00 | 0.00 | 393.08 | 0.00 | 0.00 | 21 | 0.00 | 1363 | 569 | 0.00 | 0.04 |
| E-n30-k4 | 503 | 0.00 | 0.00 | 698.98 | 0.59 | | | 0.00 | 1593 | 503 | 0.00 | 4.3 |
| E-n33-k4 | 835 | 0.48 | 0.00 | 818.03 | 0.00 | 0.00 | 24 | 0.00 | 1548 | 835 | 0.00 | 0.67 |
| E-n51-k5 | 521 | | | | 0.00 | 0.00 | 26 | 0.00 | 2034 | 521 | 0.00 | 6.75 |
| E-n76-k7 | 682 | 2.05 | 1.91 | 5186.30 | 0.15 | 2.20 | 29 | 0.15 | 2837 | 683 | 0.15 | 15.66 |
| E-n76-k8 | 735 | 1.09 | 1.22 | 5289.70 | | 2.40 | 30 | 0.00 | 2666 | 737 | 0.27 | 14.86 |
| E-n76-k10 | 830 | | | | 0.60 | 0.00 | 34 | 0.36 | 2503 | 830 | 0.00 | 17.58 |
| E-n76-k14 | 1021 | 0.00 | 0.88 | 6190.70 | 0.78 | 2.45 | 36 | 0.29 | 2642 | 1021 | 0.00 | 17.51 |
| E-n101-k8 | 815 | | | | | 2.21 | 31 | 0.25 | 3954 | 817 | 0.25 | 7.67 |
| E-n101-k14 | 1067 | 1.41 | 1.41 | 12039.00 | | | | 1.31 | 3233 | 1068 | 0.09 | 72.62 |
| Average | 723 | 0.63 | 0.68 | 3882.90 | 0.27 | 1.21 | 28.11 | 0.21 | 2327 | 723.55 | 0.07 | 14.49 |

proposed algorithm. On the other hand, it was used to test the universality and effectiveness of SP procedure, which could reduce the difference in solution quality caused by different algorithm strategies.

We prepared 10 algorithms to solve all the instances. The first five algorithms consist of HHMAB-SP and other 4 algorithms with different high-level strategy mentioned above including RS, RW, PM and AP. The other five algorithms are the former five algorithms without set partitioning procedure. We used these 10 algorithms to solve all the instances, and the average deviation percentage of instance sets are shown in Fig. 2.

According to the result of Fig. 2, we have some findings. For five algorithms without a SP procedure, HHAMB has better performance than other four algorithms. In the meantime, we also find that these hyper-heuristics with a SP procedure are easier to have lower deviation values. Furthermore, the hyper-heuristic algorithms with a SP procedure can outperform those without a SP procedure. The adding of SP procedure to the hyper-heuristic algorithm can reduce the difference of these algorithms.

Taken together, we can draw a conclusion that combing set partitioning procedure with the algorithms can enhance the performance of it. The adding set partitioning procedure to the algorithms with different selection strategies, it also can reduce the difference between the algorithms with different

TABLE V
RESULTS OF THE CVRP SET E

| Instance | BKS | SC-ESA | LNS-ACO | | CVRP-FA | PSA | | ISA-CO | | HHMAB-SP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Gap | Gap | Time | Gap | Gap | Time | Gap | Time | Best | Gap | Time |
| P-n16-k8 | 450 | 0.00 | 0.00 | 737.30 | 0.00 | 0.00 | 39 | 0.00 | 1379 | 450 | 0.00 | 1.39 |
| P-n19-k2 | 212 | 3.30 | 0.00 | 363.90 | 0.00 | 2.83 | 22 | 0.00 | 1139 | 212 | 0.00 | 0.51 |
| P-n20-k2 | 216 | 0.93 | 0.00 | 353.04 | 0.00 | 1.39 | 33 | 0.00 | 1165 | 216 | 0.00 | 1.91 |
| P-n21-k2 | 211 | 0.47 | 0.00 | 399.88 | 0.00 | 2.84 | 35 | 0.00 | 1212 | 211 | 0.00 | 0.47 |
| P-n22-k2 | 216 | 0.00 | 0.00 | 413.12 | 0.00 | 0.00 | 32 | 0.00 | 1244 | 216 | 0.00 | 0.36 |
| P-n22-k9 | 590 | 0.00 | 0.00 | 562.84 | 0.00 | | | | | 590 | 0.00 | 2.09 |
| P-n23-k8 | 529 | 0.00 | 0.00 | 615.53 | 0.00 | | | 0.00 | 1514 | 529 | 0.00 | 2.46 |
| P-n40-k5 | 458 | 0.22 | 0.00 | 1227.50 | 0.00 | 2.18 | 39 | 0.00 | 1694 | 458 | 0.00 | 2.03 |
| P-n45-k5 | 510 | 0.20 | 0.00 | 1569.40 | 0.00 | 1.37 | 41 | 0.00 | 1811 | 510 | 0.00 | 4.56 |
| P-n50-k7 | 554 | 0.00 | 0.00 | 2025.60 | 0.00 | 1.26 | 42 | 0.00 | 1868 | 554 | 0.00 | 4.73 |
| P-n50-k10 | 696 | 0.14 | 0.00 | 2347.80 | 0.14 | 2.87 | 32 | 0.00 | 1933 | 696 | 0.00 | 8.29 |
| P-n51-k10 | 741 | 0.00 | 0.81 | 2449.10 | 0.13 | 3.78 | 34 | 0.00 | 1990 | 741 | 0.00 | 8.14 |
| P-n55-k7 | 568 | 1.06 | 0.00 | 2532.90 | 0.00 | 2.29 | 42 | 0.00 | 2003 | 568 | 0.00 | 9.56 |
| P-n55-k8 | 588 | 0.00 | 0.26 | 2648.85 | -2.04 | | | -2.04 | 2032 | 576 | -2.04 | 7.45 |
| P-n55-k10 | 694 | 0.14 | 0.00 | 2876.60 | 0.58 | 3.89 | 42 | 0.14 | 2047 | 694 | 0.00 | 12.69 |
| P-n60-k10 | 744 | 0.13 | 1.48 | 3992.70 | 0.67 | 5.38 | 43 | 0.00 | 2143 | 744 | 0.00 | 12.33 |
| P-n60-k15 | 968 | 0.00 | 0.93 | 3997.20 | 0.00 | 2.17 | 30 | 0.00 | 2475 | 968 | 0.00 | 12.89 |
| P-n65-k10 | 792 | 0.51 | 1.01 | 3883.00 | 0.00 | 1.81 | 35 | 0.00 | 2216 | 792 | 0.00 | 13.16 |
| P-n70-k10 | 827 | 0.00 | 1.21 | 4640.60 | 0.00 | 2.15 | 37 | 0.24 | 2335 | 827 | 0.00 | 15.28 |
| P-n76-k4 | 593 | 2.36 | 0.84 | 5054.70 | 0.00 | 1.52 | 36 | 0.00 | 4449 | 594 | 0.17 | 15.58 |
| P-n76-k5 | 627 | 3.03 | 2.87 | 4940.00 | 0.16 | 1.75 | 37 | 0.00 | 3564 | 627 | 0.00 | 15.3 |
| P-n101-k4 | 681 | | | | 0.00 | 3.52 | 40 | 0.00 | 6897 | 681 | 0.00 | 23.26 |
| Average | 566.59 | 0.64 | 0.51 | 2262.45 | -0.02 | 2.26 | 36.37 | -0.08 | 2243 | 566.09 | -0.09 | 7.93 |

TABLE VI
SUCCESS RATES OF ALL THE ALGORITHMS

| Instances | SC-ESA | LNS-ACO | CVRP-FA | PSA | ISA-CO | HHMAB-SP |
|---|---|---|---|---|---|---|
| set A | 0.63 | 0.63 | 0.48 | 0.57 | 0.85 | 0.96 |
| set B | 0.32 | 0.68 | 0.41 | 0.56 | 0.95 | 0.91 |
| set E | 0.50 | 0.50 | 0.50 | 0.44 | 0.55 | 0.73 |
| set P | 0.43 | 0.62 | 0.73 | 0.11 | 0.90 | 0.95 |

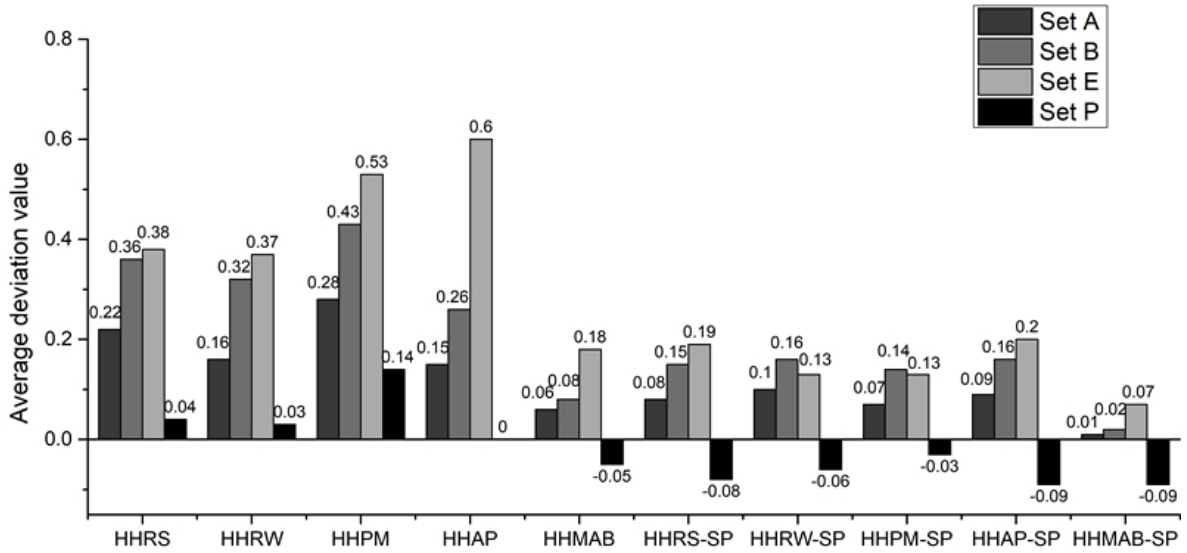


Fig. 2. Average deviation values obtained ten algorithms on four sets

high-level selection strategies. The set partitioning procedure can find a better solution from the sight of global view and it can take advantage of exact algorithm. From the view of computation time, it consumes more CPU time because it

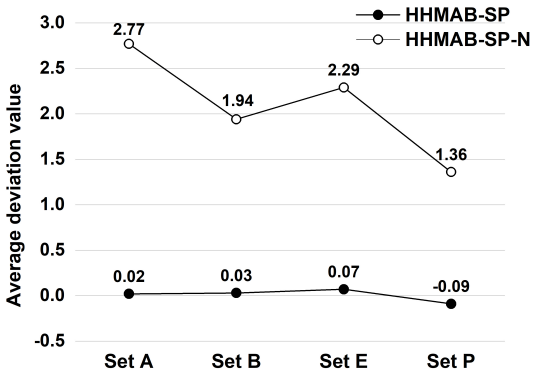


Fig. 3. Average deviation values obtained two algorithms on four instance sets

needs to be solved by CPLEX optimization software.

### *E. Performance Analysis of Ruin-and-Recreate Low-Level Heuristic*

The low-level heuristics in the proposed algorithm are made up of six regular neighborhood operators and one neighborhood operator based on ruin-and-recreate. It is generally regarded that the neighborhood operator based on ruin-and-recreate can explore larger solution space [41]. Therefore, we test the performance of ruin-and-recreate low-level heuristic in our proposed algorithm.

We used the algorithm without ruin-and-recreate low-level heuristic (denoted as HHMAB-SP-N) and the HHMAB-SP algorithm solve four instance sets. The results of all instances sets are shown in Fig. 3. We can see that the HHMAB-SP algorithm outperforms HHMAB-SP-N. For each instance set, the HHMAB-SP algorithm has lower average deviation value. The average deviation values of all sets have greatly decreased because of the ruin-and-recreate operator, and the improvement percentages of them are all larger than 98%. The results also verify that the low-level heuristic based on ruin-and-recreate is very effective in our proposed algorithm.

## VI. Conclusion

In this article, we designed a selection hyper-heuristic algorithm solving CVRP. The proposed hyper-heuristic algorithm used a multi-armed bandit algorithm with upper confidence bound strategy as the high-level selection method. In the iterative local search framework, every low-level heuristic was selected by the accumulative reward. The low-level heuristics consisted of some regular neighborhood operators and a ruin-and-recreated based neighborhood operator. Meanwhile, the indeterminate acceptance rules were employed to decide to accept or refuse the neighborhood solution. These designed strategies made the proposed algorithm keep the balance between diversification and intensification. Further, a set partitioning model was built by the routes that recorded in the local search phase, when the algorithm could not obtain the best-known solution. Then the SP model was solved by CPLEX 12.6 to get a better solution. It takes advantage of exact algorithms to overcome the short-sight shortcoming of local search algorithms.

We tested our algorithm on four CVRP standard instance sets including 82 instances and did some comparison experiments. From the results, we reach some conclusions. First of all, the proposed algorithm is more competitive. For four instance sets, the proposed algorithm has lowest average deviations relative to the best-known values. Secondly, the multi-armed bandit selection strategy used in the proposed algorithm is more effective than other selection strategies, such as random selection, roulette wheel selection, probability matching and adaptive pursuit strategy. Thirdly, the SP procedure in the second stage of the proposed algorithm can find a better solution from a view of global. Finally, the low-level heuristic designed based on the ruin-and-recreate rules can not only enlarge the search space of neighborhood solution in the local search of process, but also help to avoid trapping local optima. The experimental results show that the ruin-and-recreate low-level heuristic can enhance the quality of our proposed algorithm.

In the future, we will study other selection strategies based on reinforcement learning because of its excellent performance. Further, we will extend our proposed algorithm to solver other VRP variants.

## References

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80-91, 1959.
[2] R. Baladacci, P. Toth and D. Vigo, "Recent advances in vehicle rioting exact algorithms," *4OR*, vol. 5, pp. 269-298, 2007.
[3] R. Baladacci, P. Toth and D. Vigo, "Exact algorithms for routing problems under vehicle capacity constraints," *Annals of Operations Research*, vol. 175, no. 1, pp. 213-245, 2010.
[4] M. Poggi and E. Uchoa, "New exact algorithms for the capacitated vehicle routing problem," in *Vehicle routing: Problems, methods, and applications 2014*, pp. 59-86.
[5] A. Subramanian, E. Uchoa and L. S. Ochi, "A hybrid algorithm for a class of vehicle routing problems," *Computers & Operations Research*, vol. 40, no. 10, pp. 2519-2531, 2013.
[6] T. Vidal, T. G. Crainic, M. Gendreau and C. Prins, "A unified solution framework for multi-attribute vehicle routing problems," *European Journal of Operational Research*, vol. 234, no. 3, pp. 658-673, 2014.
[7] V. R. Máximo and M. C. V. Nascimento, "A hybrid adaptive iterated local search with diversification control to the capacitated vehicle routing problem," *European Journal of Operational Research*, vol. 294, pp. 1108-1119, 2021.
[8] T. Vidal, T. G. Crainic, M. Gendreau and C. Prins, "Heuristics for multi-attribute vehicle routing problems: A survey and synthesis," *European Journal of Operational Research*, vol. 231, pp. 1-21, 2013.
[9] G. Laporte, "Fifty years of vehicle routing," *Transportation Science*, vol. 43, no. 4, pp. 408-416, 2009.
[10] A. O. Adewumi and O. J. Adeleke, "A survey of recent advances in vehicle routing problems," *International Journal of System Assurance Engineering & Management*, vol. 9, no. 1, pp: 155-172, 2018.
[11] R. Elshaer and H. Awad, "A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants," *Computers & Industrial Engineering*, vol. 140, 106242, 2020.
[12] 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.
[13] J. H. Drake, A. Kheiri, E. Özcan and E. K. Burke, "Recent advances in selection hyper-heuristics," *European Journal of Operational Research*, vol. 285, pp. 405-428, 2020.
[14] 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.
[15] S. Asta and E. Özcan, "A tensor based selection hyper-heuristic for cross-domain heuristic search," *Information Science*, vol. 299, pp. 412-432, 2015.
[16] P. Ross, "Hyper-heuristics," In *Search methodologies: Introductory tutorials in optimization and decision support techniques 2014*, pp. 611-638.
[17] S. Asta, E. Özcan and T. Curtois, "A tensor based hyper-heuristic for nurse rostering," *Knowledge-Based System*, vol.98, pp. 185-199, 2016.
[18] A. Kheiri, A. Gretsista, E. Keedwell, G. Lulli, M. G. Epitropakis and E. K. Burke, "A hyper-heuristic approach based upon a hidden Markov model for the multi-stage nurse rostering problem," *Computers & Operations Research*, vol. 130, 105221, 2021.

[19] P. Auer, N. Cesa-Bianchi and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235-256, 2002.
[20] G. Dueck, "New optimization heuristics: the great deluge algorithm and the record-to-record travel," *Journal of Computational Physics*, vol. 104, no. 1, pp. 86-92, 1993.
[21] M. Balinski and R. Quandt, "On an integer program for a delivery problem," *Operation Research*, vol. 12, pp. 300-304, 1964.
[22] Y. Rochat and E. D. Taillard, "Probabilistic diversification and intensification in local search for vehicle routing," *Journal of Heuristics*, vol. 1, pp. 147-167, 1995.
[23] G. Alvarenga, G. Mateus and G. De Tomi, "A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows," *Computers and Operations Research*, vol. 34, no. 6, pp. 1561-1584, 2007.
[24] A. Subramanian, P. H. V. Penna, E. Uchoa and L. S. Ochi, "A hybrid algorithm for the Heterogeneous Fleet Vehicle Routing Problem," *European Journal of Operational Research*, vol. 221, pp. 285-295, 2012.
[25] P. Grangier, M. Gendreau, F. Lehuédé and L. M. Rousseau, "A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking," *Computers & Operations Research*, vol. 84, pp. 116-126, 2017.
[26] O. Tellez, S. Vercranene, F. Lehuédé, O. Péton and T. Monteiro, "The fleet size and mix dial-a-ride problem with reconfigurable vehicle capacity," *Transportation Research Part C: Emerging Technologies*, vol. 91, pp. 99-123, 2018.
[27] D. Dumez, F. Lehuédé and O. Péton, "A large neighborhood search approach to the vehicle routing problem with delivery options," *Transportation Research Part B: Methodological*, vol. 144, pp. 103-132, 2021.
[28] G. A. Bula, C. Prodhon, F. A. Gonzalez, H. M. Afsar and N. Velasco, "Variable neighborhood search to solve the vehicle routing problem for hazardous materials transportation," *Journal of Hazardous Materials*, vol. 324, pp. 472-480, 2016.
[29] L. X. Dang, Y. E. Hou, Q. S. Liu, 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.
[30] Y. E. Hou, L. X. Dang, Y. F. Kong, Z. Y. Wang, and Q. J. Zhao, "A Hybrid Metaheuristic Algorithm for the Heterogeneous School Bus Routing Problem and a Real Case Study," *IAENG International Journal of Computer Science*, vol. 47, no. 4, pp. 775-785, 2020.
[31] Y. E. Hou, N. Zhao, L. X. Dang, 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.
[32] P. Garrido and C. Castro, "Stable solving of cvrps using hyperheuristics," In *Genetic and Evolutionary Computation Conference 2009*, pp. 255-262.
[33] P. Garrido and M. C. Riff, "Dvrp: A hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic," *Journal of Heuristics*, vol. 16, no. 6, pp. 795-834, 2010.
[34] R. J. Marshall, M. Johnston and M. Zhang, "Hyper-heuristic operator selection and acceptance criteria," In *Lecture Notes in Computer Science:the European Conference on evolutionary Computation in combinatorial optimization 2015*, pp. 99-113.
[35] N. R. Sabar, X. J. Zhang and A. Song, "A math-hyper-heuristic approach for large-scale vehicle routing problems with time windows," In *proceedings of the IEEE congress on evolutionary computation 2015*, pp. 830-837.
[36] W. Qin, Z. L. Zhuang, Z. H. Huang and H. Z. Huang, "A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem," *Computers & Industrial Engineering*, vol. 156, 107252, 2021.
[37] D. E. Goldberg, "Probability matching, the magnitude of reinforcement, and classifier system bidding," *Machine Learning*, vol. 5, pp. 407-425, 1990.
[38] M. A. L. Thathachar and P. S. Sastry, "A Class of Rapidly Converging Algorithms for Learning Automata," *IEEE Transactions on Systems Man and Cybernetics*, vol. 15, pp. 168-175, 1985.
[39] N. R. Sabar, M. Ayob, G. Kendall and R. Qu, "A Dynamic Multiarmed Bandit-Gene Expression Programming Hyper-Heuristic for Combinatorial Optimization Problems," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 217-228, 2015.
[40] L. Fialho, L. Da Costa, M. Schoenauer and M. Sebag, "Analyzing bandit-based adaptive operator selection mechanisms," *Annals of Mathematics and Artificial Intelligence*, vol. 60, no. 1-2, pp. 25-64, 2010.
[41] G. Schrimpf, J. Scheider, H. Stamm-Wilbrandt and G. Dueck, "Record breaking optimization results using the ruin and recreate principle," *Journal of Computational Physics*, vol. 159, no. 2, pp. 139-171, 2000.
[42] 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.
[43] 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.
[44] M. Stanojević, B. Stanojević and M. Vujôsević, "Enhanced savings calculation and its applications for solving capacitated vehicle routing problem," *Applied Mathematics and Computation*, vol. 29, no. 20, pp. 10302-10312, 2013.
[45] S. Akpinar, "Hybrid large neighbourhood search algorithm for capacitated vehicle routing problem," *Expert Systems with Applications*, vol. 61, pp. 28-38, 2016.
[46] 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.
[47] İ. İlhan, "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.
[48] İ. İlhan, "An improved simulated annealing algorithm with crossover operator for capacitated vehicle routing problem," *Swarm and Evolutionary Computation*, vol. 64, 100911, 2021.