A Hybrid Metaheuristic Algorithm for the Bi-objective School Bus Routing Problem

Lan-xue Dang, Yan-e Hou, Qing-song Liu and Yun-feng Kong

***Abstract*—This paper deals with the school bus routing problem (SBRP) while satisfying with bus capacity and student travel time constraints. The objective is to minimize the number of school buses as well as the total travel distance. A hybrid metaheuristic, which is combined iterated local search (ILS) with set partitioning procedure (SP), is proposed for this bi-objective problem. An SP model is built by the routes which are generated in the execution of ILS, and then the model is solved by the optimization software CPLEX. In the local search of ILS, four neighborhood operators are sequentially executed to improve the solution, and the routes of the improved solution and the best local optimization obtained every iteration are both put into the route pool. To keep the diversification of the local search, an effective perturbation method based on ruin and recreate is also adopted. The developed algorithm was tested on the benchmark instances. The results show that the proposed algorithm is effective.**

***Index terms*—school bus routing problem, bi-objective, iterated local search, set partitioning, hybrid metaheuristic**

I. Introduction

WITH the development of the compulsory education in China, providing the bus service for the students has become one of the main tasks for the education authorities. Providing the bus service not only can enhance the quality of school services, but also relieve the traffic pressure, especially for the relatively densely populated areas. Some areas in China, such as Shanghai, Shenzhen, Dalian, have begun the work of school bus services to explore the model of school bus operation management. Among the school bus operation management, planning school buses routes are the basic work. It is generally agreed that the reasonable school buses planning can reduce the purchase costs and the ordinary operating costs of school buses while ensuring student safety. Therefore, it is important to develop an efficient algorithm, which allows the transportation department in organizing routes easily and efficiently. School bus routing problem (SBRP) seeks to plan an efficient schedule for a fleet of school buses where each bus picks up some students from bus stops and delivers them to their designated school while satisfying with various constraints [1]. SBRP belongs to an application area of vehicle routing problem (VRP), and it is an NP-hard combination optimization problem. Although the model and algorithms for SBRP have been researched for many years, the problem is also continuously studied because of the complexity of constraints and planning objectives in real life [2-4].

The mainly operation mode of school bus service in China is the single-school SBRP, that is the fleet of buses are servicing for one school. We can reduce the single-school SBRP to the capacitated vehicle routing problem (CVRP) if we only consider the students capacity constraint. However, in practice, single-school SBRP is more complex than the CVRP. There are many other constraints such as maximum riding time for each student, school time window and the maximum bus travel distance which need to be considered. To satisfy school bus service regulations, the objectives of single-school SBRP need to balance the operation costs, services quality, and equality. As more objectives added to the problem, the process of organizing efficient routes becomes more time-consuming and complicated. In the current literature, the commonly used methods are including exact methods, heuristic methods, and metaheuristics. The exact methods suit for small size problem and can get the exact best routes [5, 6], e.g. integer linear programming (MIP), dynamic programming. The heuristics and metaheuristic methods are non-exact methods, which can only get the approximation of best routes, e.g. saving heuristic, simulated annealing, ant colony optimization (ACO). However, these methods can solve practical large dataset problems within reasonable computational time. Therefore, the heuristics and metaheuristics are the better to solve the large dataset for single-school SBRP [7, 8]. From the objective perspective, the common objectives of single-school SBRP are aiming to minimize the bus travel distance or the number of school bus [6, 7]. In some cases, researchers combined the two objectives, which called the bi-objective problem. One of the bi-objective is to minimize the number of buses to reduce the bus purchase budget, and the other is to minimize the total travel distance to reduce the bus travel cost in practical. By combining the two objectives, it has a significant influence on practical application for SBRP. However, this bi-objective problem is only solved by exact methods [6] or heuristic algorithm [9]. Seldom paper tries to combine the exact method and the metaheuristic method to solve the bi-objective problem. This paper aims to fill this gap to improve the efficiency of the current algorithm. Based on the current metaheuristics techniques, we design a

The manuscript is received April 18, 2018. This research has been partially supported by the National Natural Science Foundation of China (Grant No. 41801310, No.41401461).

L. X. Dang is with Henan Key Laboratory of Big Data Analysis and Processing, College of Computer and Information Engineering, Henan University, Kaifeng, Henan 475004, China (e-mail: danglx@foxmail.com)

Y. E. Hou is with College of Computer and Information Engineering, Henan University, Kaifeng, Henan 475004, China (corresponding author, phone:+86-13592123785, e-mail: houyane@henu.edu.cn)

Q. S. Liu is with the Department of Geography, Kent State University, Kent, OH 44240, USA (e-mail: qliu20@kent.edu)

Y. F. Kong is with College of Environment and Planning, Henan University, Kaifeng, Henan 475004, China (e-mail: yfkong@henu.edu.cn)

new metaheuristics algorithm, which combines the two categories of methods to tackle bi-objective SBRP.

On the one hand, the Iterated local search (ILS) is one kind of metaheuristic algorithm, which has been successfully applied to combination optimal problems [10]. The advantage of ILS is that it is easy to implement and it can integrate with other algorithms effectively. On the other hand, Local search-based set-partitioning (SP) method can effectively improve the VRP/SBRP solutions produced by local search methods (e.g., ILS) [11-13]. Therefore, in this study, we proposed a hybrid metaheuristic, which integrates the ILS with SP to solve the bi-objective single-school SBRP. In our proposed algorithm, ILS used a perturbation mechanism based on ruin and recreated principles [14] aiming to add the diversity of algorithm. The SP model was built by the search history of ILS to find the global best solution. The experiments on a set of benchmark instances prove that our ILS algorithm outperforms other methods in some scenarios and has advantages in solution quality solving for bi-objective SBRP problems.

The paper is organized as follows. The bi-objective SBRP and an MIP formulation are defined in Section II. Section III describes the design of our algorithm. Section IV shows the results on a set of SBRP instances. The algorithm performance is also compared with the MIP solutions obtained from CPLEX optimizer and other algorithms. Section V presents the concluding remarks of this work.

II. Problem Definitions

The school bus routing problem can be defined as the following graph problem. Let $G=(V,A)$ be a complete graph where $V=\{0,1,...,n\}$ is the node set, and $A=\{(i,j),i,j\in V\mid i\neq j\}$ is the arc set. Nodes $\{0,1,...,n\}$ correspond to the student stops, each with a known number of students q_i, to be served, whereas node 0 corresponds to the school as a depot (q_0=0). The costs c_{ij} and t_{ij} are associated with each arc $(i,j)\in A$, and represent the travel distance and travel time from node i to node j. A set of school buses are available at the depot, each of them has the same capacity Q. Each bus leaves from school depot, visits several student stops and returns to school. Every stop must be visited only once. Also, some schools usually manage the length of time that the students ride on the bus, and the maximum riding time for any one student shall not exceed $T_{\max}$.

We assume cost matrix is symmetric, where $c_{ij}=c_{ji}$ and $t_{ij}=t_{ji}$. Subject to school bus capacity and maximum riding time constraints, both the number of school buses and the total travel distance have to be minimized. The service time for picking up students at stop i, t_i (t_0=0), can be estimated by the number of students at that stop. The bi-objective SBRP can be expressed as an MIP model. The model is proposed based on the two-index vehicle flow formulation, which uses binary decision variables x_{ij} ($i,j\in V\mid i\neq j$) to indicate if a bus traverses or not an arc in the optimal solution. Additional integer decision variables y_i (y_0=0) and z_i (z_0=0) donate the cumulative number of students in a bus and the cumulative travel time of the bus when it leaves from stop i. The bi-objective MIP formulation for SBRP is shown as follows:

$$\min \quad Z=M_0\sum_{i\in V|i\neq 0}x_{0i}+\sum_{i\in V}\sum_{j\in V|i\neq j,i\neq 0}c_{ij}x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{j\in V|j\neq i}x_{ij}=\sum_{j\in V|j\neq i}x_{ji}=1,\forall i\in V\mid i\neq 0 \tag{2}$$

$$y_i\leq Q,\forall i\in V|i\neq 0 \tag{3}$$

$$y_i+q_j-y_j\leq M_1(1-x_{ij}),\forall i,j\in V\mid i\neq j,j\neq 0 \tag{4}$$

$$z_i\leq T_{\max},\forall i\in V \tag{5}$$

$$z_i+t_{ij}+t_j-z_j\leq M_2\left(1-x_{ij}\right),\forall i,j\in V\mid i\neq j,j\neq 0 \tag{6}$$

$$x_{ij}\in\{0,1\},\forall i,j\in V\mid i\neq j \tag{7}$$

$$y_i,z_i\in\{0,1,2...\},\forall i\in V \tag{8}$$

The objective function (1) is to minimize the number of routes and the total travel distance in a lexicographic manner. The parameter M_0 is a big enough positive number which represents the fixed cost of a bus. Constraints (2) ensure that every stop is served by one bus and visited exactly once. Constraints (3) guarantee that the bus capacity is never exceeded. Constraints (4) state the accumulation of students in a bus: $(y_i+q_j)x_{ij}\leq y_j$ .If the arc (i,j) is traveled, $x_{ij}=1$, then $y_i+q_j=y_j$, otherwise $(y_i+q_j)x_{ij}=0$; so the nonlinear inequality can be expressed to linear inequality by introducing a positive integer M_1 ($M_1>2Q$). Constraint (5) ensures that the riding time of every student is never exceeded the maximum riding time T. Constraints (6) state the accumulation of bus travel time: $(z_i+t_{ij}+t_j)x_{ij}\leq z_j$ by introducing a big positive integer M_2 ($M_2>2T$). Constraints (7) and (8) are the constraints on all decision variables.

III. Proposed Algorithm

A. Algorithm Framework

The proposed hybrid algorithm (ILS-SP) includes an iterated local search (ILS) heuristic and a set partitioning (SP) procedure. ILS has been successfully applied to various combination optimization problems, especially for the VRPs. Its performance depends mainly on the choice of the local search, the perturbations and the acceptance criterion [10]. There are four components in the ILS algorithm i.e. a method for generating an initial solution, neighborhood structures, perturbation scheme, and an acceptance rule. Also, the intermediate routes in the locally optimal solution identified by ILS are recorded in a route pool. After iterations of the local search, an SP model described in Section C will be built based on the routes in the route pool. The SP model is then solved by an MIP solver. The algorithm framework of ILS-SP is described in Algorithm 1.

Algorithm 1: ILS-SP (*Maxiter, Inneriter,p*)
(1) Generate an initial feasible solution S_0;
(2) *RoutePool=Null*, S_{best}=*S*=S_0;
(3) For (*i*=0; *i*<*Maxiter*; *i*++)
(4) For each local search operator *op*
(5) *S*=Localsearch(*op*,*S*,S_{best});
(6) Update_route_pool(*RoutePool*, *S*);
(7) *S*=Perturbation(*Inneriter* , *p*,*S*, S_{best});
(8) Update_route_pool(*RoutePool*, *S*);
(9) *sp*=Build_sp_model(*RoutePool*);
(10) *S**=MIPSolver(*sp*);

(11) S_{best}=GetBetter(S_{best}, S^*);
(12) Output S_{best}.

As described in Algorithm1, the initial feasible solution S_0 is first generated. In our algorithm, S_0 is obtained by the standard saving method in [15]. The procedure *Localsearch*(*op*, S , S_{best}) uses operator op to explore the neighborhood space of the current solution S . A neighborhood solution is accepted or rejected according to the acceptance rules. Once a neighborhood solution is better than the best solution S_{best} , the best solution will be updated by this solution. Then the perturbation method (shown in section C) is adapted to the solution S . The procedure *Update_route_pool*(*RoutePool*, *S*) records the routes of the solution S in *RoutePool*. The procedure *Build_sp_model* (*RoutePool*) builds an SP model based on the routes in *RoutePool*. The procedure *MIPSolver* (sp) solves the set partitioning model *sp*. At the end of the algorithm, the better solution of S_{best} and S is produced.

B. Neighborhood Structures

We use four neighborhood structures: one-point move, two-point move, 2-opt move, and or-opt move, to explore the solution space in the local search phase sequentially. All the four types of moves are performed within a route or between routes. The neighborhood structures are described as follows.

(1) *One-point move*. A student stop is removed from a route and then inserted to a different position of the same route or into a different route in the solution (Fig. 1). In Fig.1 (a), student stop 4 is removed and inserted to a new position of the same route. In Fig.1 (b), student stop 4 is removed from the right route and inserted into the left route.

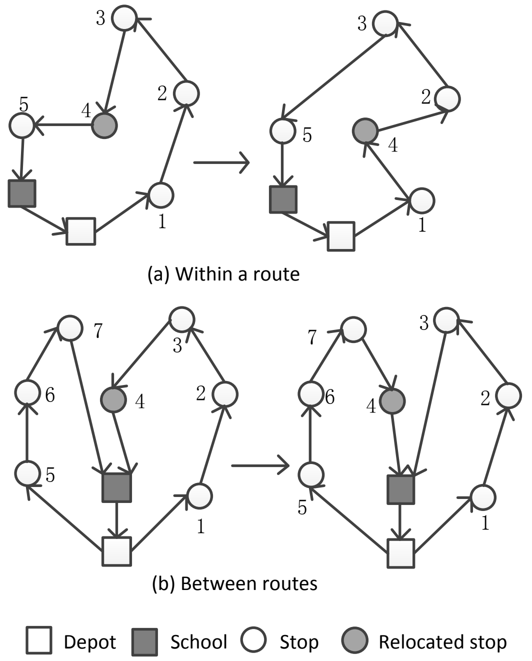

Fig. 1. Examples of One-point Move

(2) *Two-point move*. A pair of student stops are swapped. The procedure chooses a random student stop and then tries to swap it with another student stop in the same route or in different route (Fig. 2). Example in Fig.2 (a) swapped a pair of student stops (stops 2 and 5) in the same route. Example in Fig.2 (b) swapped a pair of student stops (stops 4 and 8) in different routes.

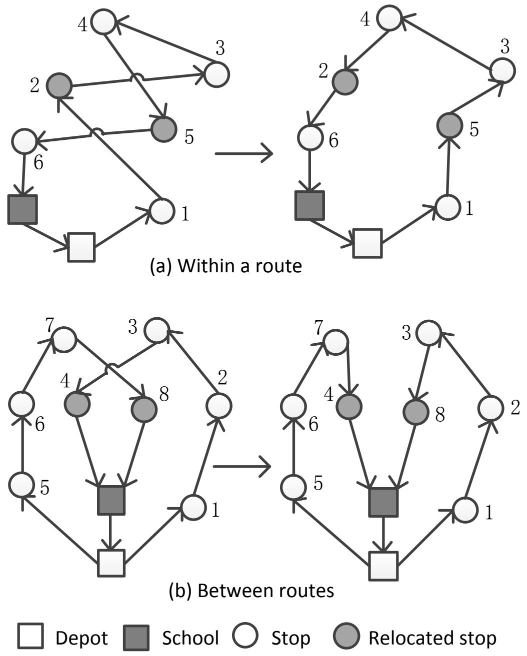

Fig. 2. Examples of Two-point Move

(3) *2-opt move*. 2-opt is a simple and effective improvement procedure. It is often used in local search of VRP. It removes two non-adjacent arcs and adds two new arcs while maintaining the route structure. When performing 2-opt in the same route, the nodes between two arcs are reversed. For example, in Fig. 3(a) after reversing the order of student stops 2, 3 and 4 a new route is generated (i.e. removing arc e_1 and e_2 from the original route and adding new arcs e_3 and e_4 to it). When performing 2-opt between different routes, we randomly select two arcs from different routes, remove them and then switch their end points. After 2-opt operation, we can generate a new feasible solution and get new routes. For example, in Fig. 3(b), arc e_1 and e_2 are located in different routes. After 2-opt, two new routes are generated (i.e. removing arc e_1 and e_2 and inserting new arc e_3 and e_4).

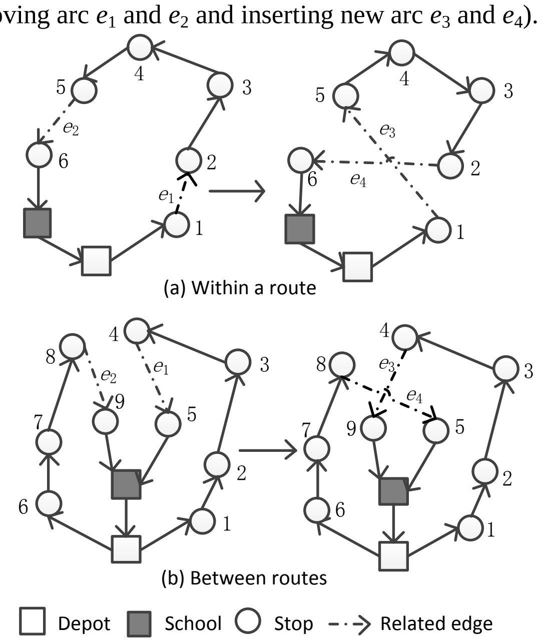

Fig. 3. 2-OPT

(4) *Or-opt move*. This move is aimed at shifting a sequence of consecutive student stops from a randomly chosen route to the same route or another route. The number of student stops is randomly generated between two and four. After determining the length of consecutive student stops, the neighborhood operator attempts to shift these student stops to other positions as long as the shifting does not violate problem constraints. Fig.4 (a) and Fig.4 (b) show examples of performing or-opt within a route and between routes respectively.

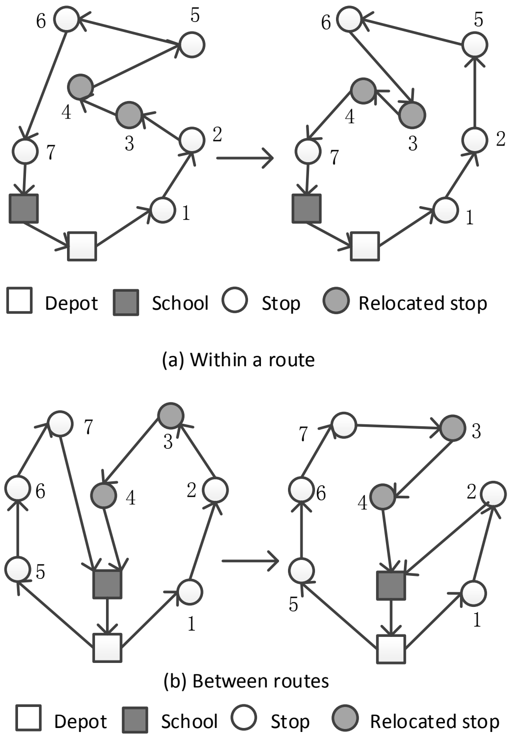

Fig. 4. Or-opt

C. Acceptance Rules

The optimization objective defined in this paper is to minimize the number of school buses and total travel distance. However, the object of the number of school buses has higher priority than the objective of total travel distance. The new local optimal solution found in local search need to evaluate whether is accepted or not according by optimization objective. We use a lexicographic bi-objective function to evaluate the neighborhood solutions:

$$Cost(S) = \alpha |S| + \beta(-\sum_{r \in S} |r|^2) + \lambda \sum_{r \in S} d(r) \qquad (9)$$

In the function, S is a solution, $|S|$ indicates the number of routes in the solution S, $|r|$ is the number of nodes in route r, and $d(r)$ indicates the distance of route r, and $\alpha >> \beta >> \lambda$. The second component maximizes $\sum_{r \in S} |r|^2$ which encourages the search operators to move stops from shorter routes to longer routes, and guide the algorithm toward the direction of reducing the number of routes more and more easily. When the number of routes decreases, the solution must be accepted; otherwise, the solution is accepted or not by the record-to-record [16] acceptance criterion. The accepted rule is defined in the following:

$$S^* = \begin{cases} S', f(S') < L \\ S', f(S') < (1+dev) \times L, dev \in [0,1] \end{cases} \qquad (10)$$

In formula (10), S' is the new neighborhood solution, S^* is the current best solution, L is the total travel distance of solution S^*, and dev is the relative coefficient. If the total travel distance of S' is better than L or less than $(1+dev) \times L$, S' is selected as the new best solution and L will be updated. Otherwise S' is rejected.

D. Perturbation Schemes

ILS is based on building a sequence of local optimal solutions by perturbing the current local optimum and applying local search to the modified solution. The perturbation strength has to be sufficient to lead the trajectory to a different attraction basin leading to a different local optimum. In our algorithm, we adopt a ruin and recreate perturbation method, which is different from the perturbation method that is usually used in ILS (e.g., random customer insertion, the exchange route segment), to perturb the current solution. The perturbation method is described in Algorithm 2.

```
Algorithm 2: Perturbation (Inneriter,p,S,Sb)
(1) S*=S;
(2) for(int i=0;i<Inneriter;i++)
(3)   R*= Destroy(S*, p);
(4)   Sr = Recreate(R*);
(5)   if Sr is better than S*
(6)       S*= Sr
(7) Sb=GetBetter(S*, Sb);
(8) S=Sb;
(9) return S;
```

The perturbation procedure executes *Inneriter* times to destroy and recreate the solution. In every iteration, the solution is firstly partially destroyed by rejected some nodes, and then the solution is recreated by reinserting the rejected nodes. Some nodes are selected from the current solution and then remove them in the *Destroy* procedure, which can explore the bigger solution space. The lager solution space could cause higher computation complexity, so the perturbation factor p is introduced to control the strength of perturbation. The value of perturbation factor is a decimal number between zero and one, i.e., when p is 0.1, it means that the 10% of nodes are removed from the solution. Thus, the current solution S^* is destroyed by randomly removing p *100% nodes to obtain a partial solution R^*. The procedure *Recreate* performs a task of recreating solution. A new solution S_r is obtained by inserting nodes in the guidance of insertion rules. The insertion rule is designed to find the position to insert with the cheapest feasible insertion cost for every insertion. The solution S_r that is better than S^* will become the next solution to perturbation.

E. Set Partitioning Procedure

After the execution of ILS, there are many routes in the route pool. A set partition model is built by these routes in the route pool. Let C be the set of stops. Let R be the set of all possible routes for SBRP and R_i be the subset of the routes covering stop i ($i \in C, R_i \subseteq R$). Each route r ($r \in R_i$) has an associated cost c_r and a binary variable x_r. The cost c_r consists of the purchase costs (denoted by M_0) and the total travel distance of the route r. A set partitioning formulation for the SBRP is given as follows:

$$\min \quad \sum_{r \in R} c_r x_r \qquad (11)$$

$$\text{s.t.} \quad \sum_{r \in R_i'} x_r = 1, \forall i \in C \qquad (12)$$

$$x_r \in \{0,1\}, \forall r \in R \qquad (13)$$

This SP model tries to select an optimal solution from the possible routes. The objective function (11) minimizes the sum of the route costs. Constraints (12) guarantee that each stop must be covered exactly once. Constraints (13) define the binary decision variables. The new solution is obtained by solving the SP model by CPLEX software.

IV. Computational Experiments

Our algorithm was programmed in C# language and executed in an AMD Phenom™ II X4 B97 3.20GHz with 4GB of RAM running 64-bit Windows 7 operating system. The parameters values were selected after some preliminary experiments. The parameters *Maxiter* and *Inneriter* were both set to 50. The perturbation factor p is set to a number within the range of [0.1, 0.4]. The parameter dev in acceptance rule is set to 10^{-4}. The SP model was solved by IBM ILOG CPLEX 12.6. The parameters for CPLEX solver were set to their default values, except that the maximum computation time was set to 60 seconds and the *MIPGap* parameter was set to 10^{-10}. The ILS-SP algorithm was executed 10 times over each instance.

A. Test Instances

We consider two sets of instances for SBRP designed by [17] to evaluate the performance of our algorithm. The instances are classified into two groups: random spatial distribution of schools and bus stops (RSRB) and clustered distribution (CSCB). Based on the data, we prepared 12 single school instances for this research. R01~R06 and C01~C06 come from the RSRB01 and CSCB01 benchmark instances. The number of student stops is 17~75. We assume the capacity of each school bus is 66, and its average speed is 20 miles per hour. The service time at student stop is an integer number estimated by the formula 19+2.6*q, where q is the number of students at the stop. The maximum riding time (MRT) of a student in a bus is set to 45 minutes. The distance between any two nodes is calculated by Manhattan distance.

B. Comparison of Exact Algorithm and ILS-SP

In order to make comparison our results with the current exact methods, we run both the ILS-SP algorithm and the exact algorithm (i.e. the MIP model) on 12 instances. According to Section II, we build the MIP models for every instance. The model parameters M_0, M_1, M_2 are set to 10^8, 10^3 and 10^5 respectively. The models were solved by CPLEX solver which we set the number of threads to 8, the maximum computation time to 1200s, the *MIPGap* parameter to 10^{-10} and other parameters to default values.

The ILS-SP and MIP solutions are shown in Table I. Num indicate the number of bus stops, which is the size of the instance. LB is the lower bound of a number of buses, which is the smallest positive integer that is greater than the total demand of all stops divided by the capacity of the school bus. N and D represent the number of routes and the total travel distance in seconds. The computing time in seconds on the computer is indicated by T. The optimal solutions given by CPLEX are labeled with a star (*), and the optima and better solutions found by our algorithm are recorded in bold.

As shown in Table I, ILS-SP is more competitive than the MIP model. Compared with MIP, ILS-SP uses less the numbers of buses, and it has the shorter total travel distance on average. ILS-SP finds all the solutions with the optimal numbers of buses and total travel distance, which are obtained by MIP. For some solutions, ILS-SP can obtain better solution than MIP, such as for instance C01, C02, and C05. Compared with MIP, ILS-SP improves the number of buses and the total travel distance by 1.32% and 0.87% on average respectively. Further, ILS-SP algorithm uses less computation time.

C. Effective of Set Partitioning Procedure

We first run a heuristic method without SP (donated as ILS) on the same datasets as well as the proposed algorithm. For the heuristic method, the algorithm ILS has the same parameters values as our proposed ILS-SP. The results obtained by both two algorithms are shown in Table II. The columns, N_{avg} ,$D_{avg,}$ and T_{avg} represent the number of routes, the total travel distance and execution time in average among the 10 solutions, respectively. The columns, N ,D, and T is the same as that are defined in TABLE I.

There are some findings from TABLE II. (1) Compared with ILS, ILS-SP finds better solutions with the numbers of buses. Because of the first objective is the number of buses, ILS-SP can obtain a better solution for R01, and it improves the numbers of buses by 0.78% on average. For some instances, such as C02~C03, C04, C06, R02~R04 and R06, ILS-SP can keep the same performance with ILS. But for R03, R05, C01, and C05, ILS-SP can decrease the total travel distance when the number of buses is not improved. (2) The ILS-SP algorithm is more robust than ILS in all. For some instances, such as R01, R02, R05, and C02, ILS-SP has a more stable average route number. The number of optimal average route number obtained by ILS-SP and ILS are 4 and 1 respectively. Meanwhile, ILS-SP also finds the optimal average total travel distance on instances C04 and C06. (3) The ILS-SP algorithm solved the problem instances efficiently. The execution time of ILS-SP is a bit longer than ILS, because of solving the set partitioning model.

Based on these above results, we can find the effect of set partitioning procedure. This could be explained by the fact that locally optimal routes explored by the ILS local search are globally recombined and selected by set partitioning. The set partitioning procedure may find the best solution from a global perspective, and it also can make full use of the advantages of the SP model in solving accurately.

D. Influence of the mode of route construction on ILS-SP

We make further efforts to test the influence of mode of route construction on ILS-SP algorithm. The SP model is built by the routes recorded in the route pool in the local search procedure. The mode of route construction in the route pool could affect the solution of the SP model. We run two algorithms on the same instances, which is just different with our ILS-SP algorithm in the route construction mode. One algorithm is donated as ILS-SP-a, and the route pool of which just include the routes of local best solutions found in each iteration. The other algorithm is donated as ILS-SP-b. The routes that are improved by the local search procedure are put into the route pool of ILS-SP-b. These two algorithms and ILS-SP have the same parameters. The results of these three algorithms are shown in TABLE III. The columns in TABLE III have the same meaning with TABLE I.

As shown in TABLE II, three algorithms all find the best route number for the first optimization objective. For the total travel distance objective, ILS-SP is a bit better than the other two algorithms. For ILS-SP algorithm, the route pool consists of the routes of local search and the best local solution. The routes in the route pool are more varied, so ILS-SP could find better solutions in the global. At the same time, the size of route pool of ILS-SP is bigger than the other two algorithms. The execution time of ILS-SP is a bit longer.

E. Influence of the iterations on ILS-SP

In this section, we test the influence of the iterations on ILS-SP algorithm. The parameter *Maxiter* of ILS-SP is first set to 10 and then adding 10 every time until 90. The other parameters of ILS-SP are not changed. The results of the test are shown in TABLE IV. We use three measurement indicators, that is SNum, SDistance, and STime, to calculate the results.

SNum and SDistance indicate the sum of the best route number and best travel distance respectively. STime is the sum of the execution of time in seconds. For RSRB and CSCB instances, the results are given respectively.

The results in TABLE IV show that ILS-SP algorithm is very stable for optimizing the route number. It almost is irrelative with the number of iteration. For the total travel distance objective, the ILS-SP algorithm can find less total travel distance with the increasing of iterations at the beginning. When the number of iteration is added to 50, it is hard to find a better solution. For CSCB instances, the SDistance keeps the same value, when the number of iteration is larger than 50. For RSRB instances, the SDistance is not changed until the iteration is equal to or larger than 80. The SDistance decrease insignificantly when the number of iteration is 50. There is a little difference between CSCB and RSRB, because the random of stops and school for RSRB instances may cause the more random combination routes to be put into the route pool in the local search. For STime indicators, the execution time of ILS-SP increase with the adding of iterations. Therefore, the number of iterations should be set to an appropriate value to keep the balance between quality and efficiency of the algorithm.

F. Comparison ILS-SP with ACO algorithm

To further evaluate the performance of ILS-SP algorithm, we also run a metaheuristic method (i.e., ant colony optimization algorithm, ACO) on the same datasets as well as the proposed algorithm. For the metaheuristic method, we implemented the basic ACO algorithm [18] and had the following parameters: the number of iterates is 300, the number of ants is set to 40, a=2, b=0.7, p=0.9. The results obtained by these three algorithms are shown in TABLE V. The columns in TABLE V have the same meaning with TABLE I.

As shown in TABLE V, ILS-SP outperforms ACO. Comparing with ACO, ILS-SP finds the same numbers of buses with ACO, but it has the shorter total travel distance on average. For all the instances except C04 and C06, ILS-SP improves the total travel distance. ILS-SP decreases the total

TABLE I
THE RESULTS OF ILS-SP AND CPLEX

Instance	LB	Num	CPLEX			ILS-SP		
			N	D	T	N	D	T
R01	9	38	9*	18536	1200	**9***	18536	6.62
R02	9	40	9*	18866*	259	**9***	**18866***	8.58
R03	13	51	13*	21296	1200	**13***	21296	14.51
R04	7	35	10	20807	1200	10	20807	5.99
R05	9	42	9*	18393	594	**9***	18393	7.49
R06	8	44	9*	18076	1200	**9***	18076	10.79
C01	14	70	17	35836	1200	**16**	**34969**	19.59
C02	11	35	12	22814	1200	12	**22798**	6.76
C03	8	30	9*	18867	1200	**9***	18867	6.46
C04	7	23	7*	13327*	232	**7***	**13327***	3.88
C05	17	75	20	39830	1200	**18**	**36657**	22.92
C06	6	17	6*	9609*	3.34	**6***	**9609***	2.71
Average	9.83	41.67	10.83	21354.75	890.7	10.58	21016.75	9.69

TABLE II
COMPARISON OF ILS AND ILS-SP

Instance	ILS						ILS-SP					
	N_{avg}	D_{avg}	T_{avg}	N	D	T	N_{avg}	D_{avg}	T_{avg}	N	D	T
R01	10	17495	6.81	10	17515	6.54	9.7	17631	8.49	9*	18536	6.62
R02	9.7	19279	5.88	9*	18866*	6.33	9*	18986	8.44	9*	18866*	8.58
R03	13	21602	10.46	13	21529	10.18	13	21436	12.7	13	21296	14.51
R04	10	20824	5.16	10	20807	5.40	10	20807	7.51	10	20807	5.99
R05	11.6	22638	7.97	9*	18469	7.50	9.1	18511	9.83	9*	18393	7.49
R06	9.6	19878	7.74	9*	18076	7.65	9.1	18269	9.71	9*	18076	10.79
C01	16	35084	16.78	16	35108	16.14	16	34994	19.58	16	34969	19.59
C02	12.5	24364	6.94	12	22798	6.33	12	22798	6.78	12	22798	6.76
C03	9.2	18845	4.41	9*	18867	4.90	9*	18898	5.87	9*	18867	6.46
C04	7*	13581	2.69	7*	13327*	2.96	7*	13327*	3.15	7*	13327*	3.88
C05	18	36704	19.29	18	36660	18.24	18	36689	21.78	18	36657	22.92
C06	6*	9620	1.88	6*	9609*	1.87	6*	9609*	2.36	6*	9609*	2.71
Average	11.05	21659.5	8.00	10.67	20969.25	7.84	10.66	20996.25	9.68	10.58	21016.75	9.69

TABLE III
RESULTS OF THE THREE ALGORITHMS

Instance	ILS-SP-a			ILS-SP-b			ILS-SP		
	N	*D*	*T*	*N*	*D*	*T*	*N*	*D*	*T*
R01	9*	18640	6.09	9*	18578	5.92	9*	18536	6.62
R02	9*	18593	9.15	9*	18969	8.28	9*	18866*	8.58
R03	13	21701	13.96	13	21661	13.73	13	21296	14.51
R04	10	21431	5.07	10	21094	6.99	10	20807	5.99
R05	9*	18484	8.16	9	18484	7.13	9*	18393	7.49
R06	9*	18314	10.67	9	18314	10.84	9*	18076	10.79
C01	16	35017	19.86	16	35059	19.95	16	34969	19.59
C02	12	22814	6.92	12	22829	6.47	12	22798	6.76
C03	9	18868	6.70	9	18868	6.33	9	18867	6.46
C04	7*	13327	4.49	7*	13327	2.79	7*	13327*	3.88
C05	18	36691	23.02	18	36691	22.87	18	36657	22.92
C06	6*	9609*	2.59	6*	9609*	2.47	6*	9609*	2.71
Average	10.58	21124.08	9.72	10.58	21123.58	9.48	10.58	21016.75	9.69

TABLE IV
RESULT OF THE ALGORITHMS HAVING DIFFERENT ITERATIONS

The number of iteration	10	20	30	40	50	60	70	80	90
SNum(RSRB)	59	59	59	59	59	59	59	59	59
SNum(CSCB)	68	68	68	68	68	68	68	68	68
SDistance(RSRB)	116508	116101	116014	116007	115985	115985	115985	116014	116007
SDistance(CSCB)	136409	136271	136260	136240	136227	136227	136227	136227	136227
STime(RSRB)	12.24	24.57	36.71	47.51	58.82	60.53	65.28	70.26	85.22
STime(CSCB)	12.92	25.86	39.33	52.22	59	60.08	70.05	73.88	81.75

travel distance by 1.24% on average. It is shown that the SP procedure can improve the quality of the algorithm. The average computation time of ILS-SP is longer than that of ACO and ILS because of adding the execution of the SP procedure.

TABLE V
COMPARISON OF THE THREE ALGORITHMS

Instance	ACO			ILS-SP		
	N	*D*	*T*	*N*	*D*	*T*
R01	9	18543	3.21	9*	18536	6.62
R02	9	18953	3.06	9*	18866*	8.58
R03	13	21743	4.82	13	21296	14.51
R04	10	21470	2.87	10	20807	5.99
R05	9	18629	2.89	9*	18393	7.49
R06	9	18114	3.45	9*	18076	10.79
C01	16	35663	7.47	16	34969	19.59
C02	12	22814	2.90	12	22798	6.76
C03	9	18874	2.03	9*	18867	6.46
C04	7	13327	1.37	7*	13327*	3.88
C05	18	37610	7.92	18	36657	22.92
C06	6	9609	1.53	6*	9609*	2.71
Average	10.58	21279.08	3.67	10.58	21016.75	9.69

V. CONCLUSIONS

In this paper, we have proposed an ILS-based hybrid metaheuristic algorithm to solve bi-objective SBRP, which minimizes the number of buses and the total travel distance together. The proposed algorithm combines the ILS with SP procedure. In the execution of ILS, four neighborhood structures and a new ruin and recreate perturbation method to explore solution space to get better solutions. The routes of the solution in local search are recorded to build SP model. The SP procedure can find a better solution from a global perspective. The solutions presented to demonstrate the good performance of the ILS-SP algorithm. When compared with other heuristic algorithms, our proposed algorithm is competitive and able to find high-quality solutions.

REFERENCES

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

[2] Ellegood, W.A., J.F. Campbell, J. North, "Continuous approximation models for mixed load school bus routing," *Transportation Research Part B: Methodological*, vol.77,no.7,pp.182–198, July.2015.

[3] de Souza Lima, F.M., Pereira, D.S.D., da Conceição, S.V. et al. "A multi-objective capacitated rural school bus routing problem with heterogeneous fleet and mixed loads," *A Quarterly Journal of Operations Research*, vol.15,no.4,pp.359-386,Dec.2017.

[4] Mokhtari N and Ghezavati V. "Integration of efficient multi-objective ant-colony and a heuristic method to solve a novel multi-objective mixed load school bus routing model," *Applied Soft Computing*, vol.68,pp.92-109,Apr.2018.

[5] Bektaş T and Elmastaş S, "Solving school bus routing problems through integer programming," *Journal of the operational Research Society*,vol.58,no.12,pp.1599-1604, Dec.2007

[6] Riera-Ledesma J and Jose Salazar-Gonzalez J, "A column generation approach for a school bus routing problem with resource constraints," Computers & Operations Research, vol.40, no.2, pp.566-583, Feb.2013.

[7] Euchi J and Mraihi R, "The urban bus routing problem in the Tunisian case by the hybrid artificial ant colony algorithm," Swarm and Evolutionary Computation, Vol.2, pp.15-24, Feb.2012.

[8] Arias-rojas J, Jiménez J and Montoya-torres J R, "Solving of school bus routing problem by ant colony optimization," Revistaa, vol.17, pp.193-208, 2012.

[9] Caballero R and Laguna M, "Bi-Objective Bus Routing: An Application to School Buses in Rural Areas," Transportation Science, vol.47, no.3, pp.397-411, Mar.2013.

[10] Vidal T., Crainic T., Gendreau M. and Prins C, "Heuristics for multi-attribute vehicle routing problems: A survey and synthesis". European Journal of Operational Research, vol.231, no.1, pp.1-21, Jan.2013.

[11] Alvarenga G., Mateus G. and De Tomi G, "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, Jun.2007.

[12] Subramanian A, Penna P.H.V., Uchoa E and Ochi L.S, "A hybrid algorithm for the heterogeneous fleet vehicle routing problem," European Journal of Operational Research, vol.221, no.2, pp.285-295, Feb.2012.

[13] Subramanian A., Uchoa E. and Ochi L.S, “A hybrid algorithm for a class of vehicle routing problems,” Computers & Operations Research, vol.40, no.10, pp.2519-2531, Oct.2013.
[14] Schrimpf G, Scheider J, Stamm-Wilbrandt H and Dueck G, “Record breaking optimization results using the ruin and recreate principle,” Journal of Computational Physics, vol.159, no.2, pp.139-171, Feb.2000.
[15] Clarke G and Wright J, “Scheduling of vehicles from central depot to a number of delivery points,” Operations Research, vol.12, no.4, pp.568-581, Apr.1964.
[16] Dueck G., “New optimization heuristics: The great deluge algorithm and the record-to-record travel,” Journal of Computational Physics, vol.104, no.1, pp.86-92, Jan.1993.
[17] Park J, Tae H and Kim B I, “A post-improvement procedure for the mixed load school bus routing problem,” European Journal of Operational Research, vol.217, no.2, pp.204-213, Feb.2012.
[18] .Dorigo M. and Gambardella L.M., “Solving symmetric and asymmetric TSPs by ant colonies,” in Proc. the IEEE ICEC’96 Nagoya, Japan, 1996, pp.622-627.