**ORIGINAL RESEARCH**

# A School Bus Routing Heuristic Algorithm Allowing Heterogeneous Fleets and Bus Stop Selection

Monique Sciortino[1] · Rhyd Lewis[1] · Jonathan Thompson[1]

## Abstract

This paper addresses a school bus routing problem formulated as a capacitated and time-constrained open vehicle routing problem with a heterogeneous fleet and single loads. This problem incorporates several realistic features, such as student eligibility, maximum walking distances, bus stop selection, maximum riding times, different types of buses, multistops, and bus dwell times. A heuristic algorithm based on an iterated local search approach is proposed for this problem. It determines the selection of bus stops from a set of potential stops, the assignment of students to the selected bus stops, and the routes along the selected bus stops. The main objectives are minimizing the number of buses used, the total student walking distance, and the total route journey time. Other aims are balancing route journey times between buses and minimizing the total number of empty seats. A set of 20 real-world problem instances are used to evaluate the performance of the algorithm. Results indicate that the algorithm finds high-quality solutions in very short amounts of computational time.

**Keywords** School bus routing · Vehicle routing · Heterogeneous fleet · Bus stop selection · Iterated local search · Heuristics

## Introduction

The school bus routing problem (SBRP) is a combinatorial optimization problem that was first investigated over 4 decades ago [25] and has since received a lot of attention within the scientific community (see the section "Related Work"). The problem relates to the organisation of school bus transportation services for students between their home addresses and schools.

School bus operations constitute a challenging task from both a logistics and financial perspective. Typically, an SBRP entails compiling a list of addresses of all students

✉ Monique Sciortino
    sciortinom@cardiff.ac.uk

    Rhyd Lewis
    lewisr9@cardiff.ac.uk

    Jonathan Thompson
    thompsonjm1@cardiff.ac.uk

1   School of Mathematics, Cardiff University,
    Cardiff CF24 4AG, Wales

deemed eligible for school transportation, approving potential bus stops reachable by the students, determining the stops to be visited by the buses, assigning students to their respective bus stops, and designing routes that optimize operational efficiency without sacrificing school bus safety and service quality. These objectives are often conflicting in nature, since an improvement in the level of service quality can increase the cost of provision, and vice versa.

In various countries, school bus transportation forms part of the government's administrative mechanism and is funded through taxation. Students who live at least a certain distance from the school they attend are entitled to free or subsidized transport. In Malta, for example, school transport is provided free of charge to all kindergarten, primary, middle, and secondary state school students residing at least 1 km away from their school. Given the large amount of funds being invested in school bus transportation, it is critical for governments to minimize the total cost of these services.

An important objective is to minimize the number of buses, since this reduces their corresponding acquisition and driver employment costs. Another priority is to minimize operational costs by keeping route journey times as short as possible. This also promotes student well-being, particularly for younger children. In Wales, for example, a maximum 45 and 60 min journey time is recommended for primary

and secondary school pupils, respectively. Moreover, governments typically issue a policy concerning the maximum distance that students are expected to walk between their homes and designated bus stops. For instance, a walk of 1.6 km is deemed reasonable in Wales.

In this paper, we focus on the single-school SBRP in which routes are constructed for each school separately. Mixed loads, i.e., students from different schools travelling on the same bus at the same time, are not considered, since these are usually not permitted in the locations under study. We dedicate our research to the morning problem, whereby students are picked up from bus stops and dropped off at school. A solution to the afternoon problem, whereby students are picked up from school and dropped off at bus stops, is found by reversing the routes.

The remainder of the paper is organized as follows. The section "Related Work" presents a brief review of related work on the SBRP. The section "Problem Definition" then defines our SBRP, while the section "Algorithm Description" describes our heuristic algorithm for this problem. The section "Computational Experiments" describes the problem instances considered and our computational results. Finally, the section "Concluding Remarks and Suggestions" provides the concluding remarks and some suggestions for future work.

## Related Work

The SBRP falls into the class of NP-hard vehicle routing problems (VRPs). According to Laporte et al. [21], VRPs aim to design optimal delivery/collection routes from one or more depots to a finite set of geographically scattered customers under a variety of side constraints. A common objective in VRPs is to minimize the total operating costs of the fleet of vehicles, while typical constraints include maximum capacity restrictions on vehicles [the *capacitated* VRP (CVRP)] and maximum time/distance restrictions on routes. A taxonomic review of the VRP and its variants is presented by Braekers et al. [8].

The CVRP first appeared over 6 decades ago in the seminal paper of Dantzig and Ramser [14]. In their reviews on SBRPs, both Park and Kim [26] and Ellegood et al. [19] observe that almost all their reviewed publications consider capacity constraints. Here, we take the same assumption but choose to allow for a heterogeneous fleet of buses consisting of several types of buses, with buses of the same type having the same capacity (as in [3, 12]). In [19, 26], only around 20% and 25%, respectively, of SBRP publications are noted to model a heterogeneous fleet (e.g., [24, 29, 36, 37]). Here, we do the same to extend the homogeneous fleet SBRP that we previously studied in [35] and to shift our research towards more realistic

settings. The adaptation of our algorithm in [35] to heterogeneous fleets is the main contribution of this paper.

Another variant of the VRP proposed by Sariklis and Powell [31] is the *open* VRP (OVRP), in which routes do not start and end at a depot as in the classical VRP, but rather either start or end at a depot. Similar to [2], we model our SBRP as a capacitated and time-constrained OVRP. We do this since governments put out the designed routes to public tender, and thus, buses are kept at bus garages owned by private companies and not at the school. According to [26] and [19], approximately 66% and 35%, respectively, of SBRP publications include maximum riding time constraints (e.g., [11, 27]). They also observe that the majority of the publications on school bus routing also deal with the single-school SBRP (e.g., [13, 23, 33]).

Desrosiers et al. [15] decompose the SBRP into five subproblems. In the first subproblem, *data preparation*, a network containing the schools, student residences, potential bus stop locations, and bus depots is constructed. Information on the number of students at each residence, the school destination of each student, and the types and number of buses available and their capacities are also specified. The second subproblem, *bus stop selection*, seeks to select a subset of bus stops from a set of potential bus stops and assign students to these stops. Ellegood et al. [19] remark that less than one-quarter of SBRP research tackles this subproblem (e.g., [6, 28, 34]). The third subproblem, *route generation*, is the core of the SBRP. It deals with building initial routes through construction methods such as insertion-based or savings-based heuristics and enhancing them via improvement methods such as metaheuristics and integer programming. The last two subproblems, *school bell time adjustment* and *route scheduling*, adjust the schools' opening/closing times to allow buses to service multiple schools and establish chains of routes that can be executed by the same vehicle.

In this paper, we cover the first three subproblems stated above. For the bus stop selection and route generation subproblems, we employ the location-allocation-routing (LAR) strategy (as in [4, 5, 18]), in which bus stops are first selected, students are assigned to stops, and then, route generation follows. This strategy may lead to sub-optimal routes, since the bus stops selected in the former subproblem may not be best for the latter subproblem [7]. To address this issue, we therefore also include an operator that alters the selection of bus stops and the assignment of students, as discussed in the section "Generation of Alternative Solutions".

## Problem Definition

In our problem, we use parameter $m_w$ to indicate the maximum distance that students are expected to walk to a bus stop. We also use parameter $m_e$ to specify the minimum walking

distance students should live from school to be eligible for school transportation. As in [22, 35], our SBRP can be represented via two set of vertices, $V_1$ and $V_2$, and two sets of edges, $E_1$ and $E_2$. The vertex set $V_1$ consists of one school, $v_0$, and $n$ potential bus stops $v_1, v_2, \ldots, v_n$, and the edge set $E_1$ contains all $n(n+1)$ arcs $(u, v)$ where $u, v \in V_1$ and $u \neq v$. Each edge $(u, v)$ in the complete directed graph $(V_1, E_1)$ is weighted by the shortest driving time $t(u, v)$ from vertex $u$ to vertex $v$. Meanwhile, the vertex set $V_2$ consists of eligible student addresses, i.e., addresses that are within walking distance greater than $m_e$ from the school. Each address $w \in V_2$ has a corresponding number $s(w)$ of students residing at that address and requiring transportation to the school under consideration. The second edge set $E_2$ is the set $\{\{w, v\} : w \in V_2 \wedge v \in V_1 \backslash \{v_0\} \wedge d(w, v) \leq m_w\}$, where $d(w, v)$ represents the shortest walking distance from address $w$ to bus stop $v$.

It is assumed that the undirected bipartite graph $(V_2, V_1 \backslash \{v_0\}, E_2)$ has no isolated vertices. Otherwise, either an address has no bus stop within walking distance $m_w$ (and therefore a new bus stop must be added to $V_1$), or a bus stop has no address within walking distance $m_w$ and can be removed from $V_1$. Moreover, a bus stop $v \in V_1 \backslash \{v_0\}$ for which there exists an address $w \in V_2$ with the single incident edge $\{w, v\}$ shall be referred to as a *compulsory stop*. This is because such a stop $v$ is the only stop within walking distance $m_w$ for students living in address $w$. It must therefore be present in any solution.

A feasible solution to our SBRP is given by a set of routes $\mathcal{R} = \{R_1, R_2, \ldots\}$. An example is shown in Fig. 1. Each route $R \in \mathcal{R}$ uses one bus which needs to have adequate seating capacity for all the students boarding that route. (The issue of choosing a suitable capacity $C(R)$ for route $R$ will be considered at the end of this section.) This bus successively visits a subset of bus stops and terminates at the school $v_0$. The subset of bus stops traversed by at least one route is denoted by $V_1' \subseteq V_1 \backslash \{v_0\}$. This set should *cover* each address $w \in V_2$ at least once, meaning that students in each address $w$ will have at least one bus stop in $V_1'$ within walking distance $m_w$. Such a covering shall be referred to as a *complete covering* of $V_2$, whereas a covering that does not satisfy this property is an *incomplete covering* of $V_2$. In addition, the total number $s(R)$ of students boarding the bus on route $R$ should not exceed some pre-defined maximum bus capacity $C_{\max}$, and the journey time $t(R)$ of route $R$ should not exceed the maximum journey time $m_t$. These constraints can be expressed as follows:

$$\bigcup_{R \in \mathcal{R}} R = V_1', \tag{1}$$

$$\forall w \in V_2, \ \exists v \in V_1' \mid \{w, v\} \in E_2, \tag{2}$$

$$s(R) \leq C_{\max} \quad \forall R \in \mathcal{R}, \tag{3}$$
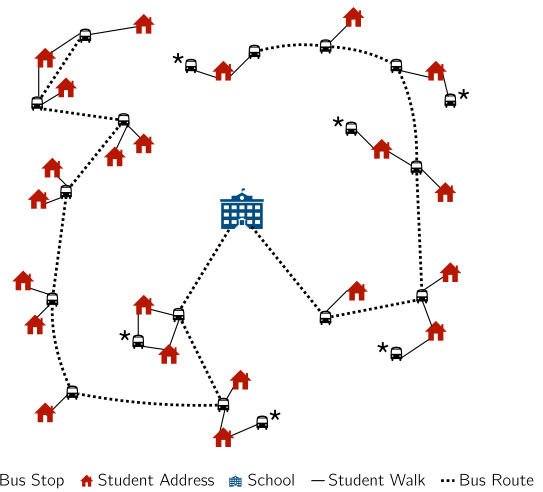


**Fig. 1** A feasible solution with $|\mathcal{R}| = 2$, $|V_1| = 21$, $|V_1'| = 14$, and $|V_2| = 22$. Bus stops marked by * are not used in the solution. All other bus stops are in the set $V_1'$

$$t(R) \leq m_t \quad \forall R \in \mathcal{R}. \tag{4}$$

It is important to note that bus stops in $V_1'$ can feature in more than one route in $\mathcal{R}$. For example, there may not be enough spare capacity in a bus to serve all students waiting at a bus stop $v \in V_1'$. In that case, bus stop $v$ must be visited by more than one bus. Here, a bus stop occurring on more than one route in a solution is called a *multistop*. In VRP literature, this characteristic is referred to as the allowance of split deliveries [16, 17]. It is assumed that each student boarding at a multistop is only permitted to board one specific route serving that stop, since, otherwise, a bus stopping at that multistop may be too full to serve a subsequent stop in its route. Generally, students living in the same address (in particular siblings) are preferably assigned to the same route; however, in practice, one may encounter unavoidable cases where students from the same address are assigned to different routes.

The calculation of the journey time $t(R)$ of route $R \in \mathcal{R}$ is composed of two components: the total *bus travel time* and the total *bus dwell time*. Each dwell time within a route captures the time spent servicing a designated bus stop; i.e., the time spent stopping the bus, opening the doors, boarding the students, and merging back into traffic. In our case, we estimate the dwell time at stop $v$ in route $R$ using the linear function $d(v, R) = d_1 + d_2 s(v, R)$, where $s(v, R)$ represents the number of boarding students at stop $v$ onto route $R$, $d_2$ represents the boarding time per student, and $d_1$ is a parameter which accounts for the remaining servicing time. Here, $d_1$ and $d_2$ are taken to be 15 and 5 s, respectively. Therefore, given a route $R = (v_1, v_2, \ldots, v_l, v_0)$, the route journey time $t(R)$ is given by

$$t(R) = \sum_{i=1}^{l-1} t(v_i, v_{i+1}) + t(v_l, v_0) + \sum_{i=1}^{l} \Big( d_1 + d_2 s(v_i, R) \Big), \quad (5)$$

where the first two terms give the total bus travel time and the last term gives the total bus dwell time.

The provision of school transportation involves high costs and is sometimes characterized by a shortage of buses. For instance, in September 2018, the Maltese Education Ministry was faced with the issue of having almost 1000 students without transport arrangements due to a lack of space on buses [30]. As previously mentioned, the primary objective of our SBRP is to identify an appropriate subset of bus stops $V_1'$ to minimize the number $|\mathcal{R}|$ of routes (buses) included in a solution. In our case, this is achieved by attempting to produce feasible solutions that use the lower bound of $\lceil \sum_{w \in V_2} s(w)/C_{max} \rceil$ routes needed to serve all students. Under the assumption of having enough buses of capacity $C_{max}$ to cater for all students, a solution satisfying constraints (1)–(3) and meeting this lower bound of $|\mathcal{R}|$ is always guaranteed, since multistops are allowed; however, any one of the routes could potentially violate the maximum riding time constraint (4). Thus, there may be cases where additional routes are needed.

In addition to minimizing the number of routes $|\mathcal{R}|$, in this paper, we also aim to target service efficiency by minimizing the total route journey time in a solution. Whenever multiple feasible solutions have the same minimum total route journey time, we also give preference to the one with the smallest discrepancy between the longest and shortest routes. This encourages equity of the service. Furthermore, we assign students to their closest stop in a selected subset of bus stops. In the future, we would like to incorporate the minimization of student walking distances into the cost function. This is not considered here, but is important for the effectiveness of the service.

At the end of the optimization process, one final task is to assign vehicles to the routes of a solution, such that the total number of empty seats is minimized. This is desirable, because larger buses typically consume higher amounts of fuel. Earlier, we mentioned that an adequate seating capacity is required for each route. In our case, each route is assigned to the smallest pre-defined bus capacity that is greater than or equal to the total number of students boarding that route. We follow this strategy, since, as we said before, governments usually construct the routes before putting them out to tender; hence, each bus type can be assumed to be available in an unlimited number. If buses of each type are limited in number, the problem of assigning buses to routes can be modelled as a minimum-weight maximum-cardinality bipartite matching problem. The bipartition consists of a set of $|\mathcal{R}|$ vertices representing the routes in the solution, and a set of $B$ vertices representing all available buses. A route vertex is then linked to a bus vertex if and only if its load is at most the capacity of the given bus. Such an edge is weighted by the difference between the bus capacity and the route load (i.e., the number of empty seats). The Hungarian algorithm, with worst-case complexity $\mathcal{O}(B^3)$, can then be used to find a matching that contains as many edges as possible and that has the minimum total weight.

## Algorithm Description

Our heuristic algorithm for this problem employs the following overall strategy. To begin, a subset of bus stops representing a complete covering is selected and a nearest-neighbour heuristic is applied to construct an initial solution using a fixed number $|\mathcal{R}|$ of routes. As mentioned, $|\mathcal{R}|$ is initially taken to be the lower bound stated in the previous section.

Note that the initial assignment of stops to routes allows the violation of the maximum riding time constraints (4). A local search routine involving six improvement heuristics is, therefore, invoked on the initial solution to try shortening the routes without changing the current subset of bus stops. After this routine has been completed, a procedure is performed whereby the current subset of selected bus stops is altered and the current solution is repaired. The local search routine is then re-applied. This entire process is repeated until a time limit is reached. If no solution satisfying (1)–(4) is achieved at this limit, then the number $|\mathcal{R}|$ of routes is increased by one and the algorithm is restarted.

## Construction of Initial Solution

In our approach, the initial subset of bus stops $V_1'$ is selected as follows. First, all compulsory stops are added to $V_1'$. The non-compulsory stops are then arranged in non-increasing order according to the number of currently uncovered addresses they serve. The stop with the largest such value is then added to $V_1'$, breaking ties randomly. This ordering and selection procedure is repeated until a complete covering of $V_2$ is obtained. Each address in $V_2$ is then assigned to the closest bus stop in $V_1'$. The assignment of addresses to stops determines the number $s_{V_1'}(v)$ of boarding students at each stop $v \in V_1'$. It may also be the case that some stops have no boarding students, in which case they are removed from $V_1'$.

Next, each bus stop in $V_1'$ is assigned to one of the $|\mathcal{R}|$ routes, such that each bus is not overloaded. In our case, this assignment follows a parallel backward implementation of the nearest-neighbour constructive heuristic. To start, $|\mathcal{R}|$ empty routes are defined and the remaining capacity $c_i$ of each route $R_i \in \mathcal{R}$ is set to $C_{max}$. The $|\mathcal{R}|$ closest stops to the school are then added at the front of the routes, one in each route. Closeness to school is measured by the dwell time at

the stop plus the shortest driving time from the stop to the school. To calculate the dwell time at stop $v \in V'_1$ in route $R_i$, the minimum of $c_i$ and $s_{V'_1}(v)$ is considered as there may be more than $c_i$ students boarding at stop $v$. In this case, a multistop is created, since the remaining $s_{V'_1}(v) - c_i$ students boarding at stop $v$ will need to be assigned to a different route $R_j$. The remaining capacities $c_i$ are then updated accordingly. This iterative procedure of determining the closest stop to the most recently added stop in route $R_i$, adding it to the front of the route and updating the remaining capacity $c_i$, is repeated until all stops in $V'_1$ are assigned to a route. On completion, an initial solution $\mathcal{R}$ will have been generated and can be evaluated according to the cost function described presently.

## Cost Function

Here, a candidate solution $\mathcal{R} = \{R_1, R_2, \dots\}$ is evaluated according to the cost function

$$f(\mathcal{R}) = \sum_{R \in \mathcal{R}} t'(R), \tag{6}$$

where

$$t'(R) = \begin{cases} t(R) & \text{if } t(R) \leq m_t, \\ m_t + m_t(1 + t(R) - m_t) & \text{otherwise.} \end{cases} \tag{7}$$

This means that if a route $R \in \mathcal{R}$ satisfies (4), then its journey time is unaltered. On the other hand, if the journey time $t(R)$ of route $R$ exceeds $m_t$, then this journey time is scaled up heavily via a penalty. The addition of the value 1 in the second case of (7) guarantees that two routes both with journey time of at most $m_t$ (and, therefore, a cost of at most $2m_t$) are always preferred over a single route with a journey time exceeding $m_t$.

## Local Search Routine

As mentioned, the intention of our local search routine is to shorten the journey times of routes in a solution $\mathcal{R}$ while maintaining the satisfaction of (1)–(3). Note that this local search acts on a solution using a fixed subset of bus stops $V'_1$. It uses a combination of three intra-route and three inter-route local search operators, with the former being applied to any single route $R_1 \in \mathcal{R}$ and the latter being applied to any pair of routes $R_1, R_2 \in \mathcal{R}$. Without loss of generality, assume that $R_1 = (v_1, v_2, \dots, v_{l_1}, v_0)$ and $R_2 = (u_1, u_2, \dots, u_{l_2}, v_0)$. The six operators considered are the following:

- *Exchange*: Choose two stops $v_i, v_j$ in $R_1$, where $1 \leq i < j \leq l_1$, and swap their position.
- *Two-Opt*: Choose two stops $v_i, v_j$ in $R_1$, where $1 \leq i < i + 3 \leq j \leq l_1$, and invert sub-route $v_i, \dots, v_j$.

Cases $j = i + 1$ and $j = i + 2$ are special cases of the exchange operator and are thus excluded.

- *Generalized Or-Opt*: Choose stops $v_i, v_j, v_k$ in $R_1$, where $1 \leq i \leq j \leq l_1$ and $(1 \leq k < i$ or $j + 1 < k \leq l_1 + 1)$. Remove sub-route $v_i, \dots, v_j$ and insert it before stop $v_k$, possibly also inverting the sub-route if this yields a better cost. If $k = l_1 + 1$, then the sub-route is inserted before school $v_0$.
- *Or-Exchange*: Choose stops $v_i, v_j$ in $R_1$, where $1 \leq i \leq j \leq l_1$, and stop $u_k$ in $R_2$, where $1 \leq k \leq l_2 + 1$. Remove sub-route $v_i, \dots, v_j$ from $R_1$ and insert it before stop $u_k$ in $R_2$, possibly also inverting the sub-route if this yields a better cost. If $k = l_2 + 1$, then the sub-route is inserted before school $v_0$.
- *Cross-Exchange*: Choose stops $v_{i_1}, v_{j_1}$ in $R_1$, where $1 \leq i_1 \leq j_1 \leq l_1$, and stops $u_{i_2}, u_{j_2}$ in $R_2$, where $1 \leq i_2 \leq j_2 \leq l_2$. Swap sub-routes $v_{i_1}, \dots, v_{j_1}$ and $u_{i_2}, \dots, u_{j_2}$, possibly inverting either sub-route if this yields a better cost.
- *Creating Multistops*: If routes $R_1$ and $R_2$ satisfy $t(R_1) > m_t$ and $s(R_2) < C_{max}$, then choose a stop $v_i$ in $R_1$, where $1 \leq i \leq l_1$, for which $s(v_i, R_1) \geq 2$. If $v_i$ is not already in $R_2$, then insert a copy of $v_i$ into $R_2$ before the stop $u_k$, where $1 \leq k \leq l_2$, (or school $v_0$) which causes the smallest increase in $t(R_2)$. Next transfer $z = \min\{s(v_i, R_1) - 1, C_{max} - s(R_2)\}$ students from the occurrence of stop $v_i$ in $R_1$ to the occurrence of stop $v_i$ in $R_2$. Here, the value $z$ gives the maximum number of students who can be transferred (hence, decreasing $t(R_1)$ as much as possible), such that both occurrences of $v_i$ have at least one boarding student and both routes $R_1$ and $R_2$ satisfy (3).

The neighbourhood sizes corresponding to the above operators are $\mathcal{O}(|V'_1|^2)$, $\mathcal{O}(|V'_1|^2)$, $\mathcal{O}(|V'_1|^3)$, $\mathcal{O}(|V'_1|^3)$, $\mathcal{O}(|V'_1|^4)$, and $\mathcal{O}(|V'_1|^3)$, respectively. These operators are the same as those used in [22, 35]. The exchange, two-opt and cross-exchange operators are also used in a similar context in [12], while the generalized Or-opt and Or-exchange are extensions (in cases where $i \neq j$) of operators used in [12, 33]. The creating multistops operator was first proposed in [22].

Note that the intra-route operators do not affect the total number of visited stops, the dwell time at each stop, or the total number of boarding students on a route. On the other hand, Or-exchange and cross-exchange moves can lead to a violation of the capacity constraints (3). Such moves are therefore not evaluated. Moreover, the two inter-route operators can result in duplicate stops in the same route, which are removed as follows. Without loss of generality, assume that sub-route $v_i, \dots, v_j$ is being transferred from route $R_1$ to route $R_2$ and that one stop $v_h, i \leq h \leq j$, is already present in $R_2$. Then, stop $v_h$ is removed from the sub-route and the

students boarding this occurrence of $v_h$ are all transferred to the occurrence of $v_h$ in $R_2$.

Our local search routine follows the direction of steepest descent. In each iteration, all moves in the union of the six neighbourhoods are evaluated and the move giving the largest reduction in cost is performed. If multiple moves give the largest reduction in cost, the one which yields the smallest discrepancy between the longest and shortest routes in the solution is performed. Such a breakage of ties aims at balancing the journey times between buses. The local search routine terminates when a solution whose neighbourhoods contain no improving moves is reached.

### Generation of Alternative Solutions

As noted, the subset of bus stops $V_1'$ is fixed during our local search routine. However, it may be the case that no solution $\mathcal{R}$ for a fixed subset $V_1'$ is feasible with respect to the maximum riding time constraints (4). In this case, altering $V_1'$ and re-running local search may lead to a better solution. For this reason, our algorithm also contains an operator that generates a new subset of bus stops $V_1''$, assigns students to these bus stops, and then creates a set of routes that use these stops. We designed four variants of the algorithm, which differ in the way they generate $V_1''$. These are the following:

   I   Generate $V_1''$ from scratch (with no reference to previous iterations);
   II  Generate $V_1''$ from the subset $V_1'$ used in the previous iteration;
   III Generate $V_1''$ from the most recent subset $V_1'$ that yielded a feasible solution with the lowest cost found so far;
   IV  Generate $V_1''$ via a trade-off between Variants II and III, whereby $V_1''$ has a 50% chance of being generated according to Variant II and a 50% chance of being generated according to Variant III.

Note that in Variant III, the subset of stops generated in the previous iteration is used if no subset has yet yielded a feasible solution.

In Variant I, the generation of $V_1''$ follows the same selection strategy as that discussed in the section "Construction of Initial Solution" and new routes are again produced via nearest-neighbour construction. For the remaining variants, the non-compulsory stops in $V_1'$ are identified and a random selection of these is removed. Assuming a total number $\alpha$ of non-compulsory stops, in our case, the number of removals is selected according to a Binomial distribution with parameters $\alpha$ and $3/\alpha$, so that three stops are removed on average. Upon removal, if we have an incomplete covering of $V_2$, then additional stops are added to $V_1'$. If all addresses not covered by the stops in $V_1'$ are covered by stops that were not originally in $V_1'$, then, at each stage, a stop from the latter set of stops that serves the largest number of uncovered addresses is added, breaking ties randomly. If, on the other hand, some address is uncovered by the stops which were not originally in $V_1'$, then at least one of the removed stops must be added back. The same selection strategy is applied in this case and the whole procedure is repeated until a new complete covering $V_1''$ of $V_2$ is achieved. Each address is then reassigned to the closest stop in $V_1''$. As before, stops with no addresses assigned to them are then removed from $V_1''$.

Having determined a new subset of bus stops, repairs now need to be made to $\mathcal{R}$, so that only bus stops in $V_1''$ feature in the solution. To do this, all occurrences of stops in $V_1' \setminus V_1''$ are first removed from $\mathcal{R}$. For stops $v \in V_1'' \cap V_1'$ for which $s_{V_1''}(v) < s_{V_1'}(v)$, $s_{V_1'}(v) - s_{V_1''}(v)$ students are removed from occurrences of $v$ in $\mathcal{R}$. If this results in an occurrence of $v$ with no boarding students, then this occurrence is removed from $\mathcal{R}$. For stops $v \in V_1'' \cap V_1'$ for which $s_{V_1''}(v) > s_{V_1'}(v)$, an attempt is made to add students to occurrences of $v$ in $\mathcal{R}$. If not all $s_{V_1''}(v) - s_{V_1'}(v)$ students can be added, then a new occurrence of $v$ must be added to $\mathcal{R}$. Stops $v \in V_1'' \setminus V_1'$ must also be added to the solution. A new stop is inserted in a route having the lowest load, at the position which causes the least increase in the route journey time. If this insertion does not cater for all students boarding that stop, then the procedure is repeated.

Having repaired solution $\mathcal{R}$ (or generated completely new routes in the case of Variant I), the local search routine is then re-invoked. This repair-and-improve process is repeated until the time limit is reached.

### Computational Experiments

A total of 20 real-world problem instances are considered here, as summarized in Table 1. The problem instances pertaining to the UK and Australia originate from [22] and can be downloaded at [9]. The remainder were generated by us and can be downloaded at [32]. Each problem instance was generated as follows. The location of a school was first identified and a number of random student addresses were selected within the school's catchment area, but more than $m_e$ km from the school. The number of students living at each address was generated randomly according to the following distribution: 1, 2, 3, and 4 with probabilities 0.45, 0.4, 0.14, and 0.01, respectively. This distribution approximates the relevant statistics in the locations considered. Potential bus stops were then identified through public records, such that each stop has at least one address within walking distance $m_w$ and each address has at least one stop within walking distance $m_w$. The shortest driving times between each bus stop pair and shortest walking distances between each bus

**Table 1** Summary statistics for the 20 real-world problem instances, listed in increasing order of $|V_1|$

| Location | Country/State | $|V_1|$ | $|V_2|$ | $S$ | $m_e$ | $m_w$ |
|---|---|---|---|---|---|---|
| Mġarr | Malta | 60 | 110 | 190 | 1.0 | 1.0 |
| Mellieħa | Malta | 86 | 98 | 171 | 1.0 | 1.0 |
| Porthcawl | Wales | 153 | 42 | 66 | 3.2 | 1.6 |
| Qrendi | Malta | 158 | 150 | 255 | 1.0 | 1.0 |
| Suffolk | England | 174 | 123 | 209 | 4.8 | 1.6 |
| Senglea | Malta | 186 | 158 | 266 | 1.0 | 1.0 |
| Victoria | Gozo | 316 | 99 | 171 | 1.0 | 1.0 |
| Pembroke | Malta | 322 | 200 | 335 | 1.0 | 1.0 |
| Canberra | ACT | 331 | 296 | 499 | 4.8 | 1.0 |
| Ħandaq | Malta | 393 | 170 | 285 | 1.0 | 1.0 |
| Valletta | Malta | 445 | 159 | 268 | 1.0 | 1.0 |
| Birkirkara | Malta | 469 | 181 | 306 | 1.0 | 1.0 |
| Ħamrun | Malta | 518 | 192 | 321 | 1.0 | 1.0 |
| Cardiff | Wales | 552 | 90 | 156 | 4.8 | 1.6 |
| Milton Keynes | England | 579 | 149 | 274 | 4.8 | 1.6 |
| Bridgend | Wales | 633 | 221 | 381 | 4.82 | 1.6 |
| Edinburgh-2 | Scotland | 917 | 190 | 320 | 1.6 | 1.6 |
| Edinburgh-1 | Scotland | 959 | 409 | 680 | 1.6 | 1.6 |
| Adelaide | South Australia | 1188 | 342 | 565 | 1.6 | 1.6 |
| Brisbane | Queensland | 1817 | 438 | 757 | 3.2 | 1.6 |

The number $S$ represents the total number of students, calculated as $\sum_{w \in V_2} s(w)$. Distances $m_e$ and $m_w$ are given in km

stop and address pair were then determined using the Bing Maps Routes API.

Here, we assume that the maximum journey time $m_t = 2700$ s (45 min) and that vehicles of different capacities are available, depending on the country of the problem instance under study.[1] These capacities were selected by inspecting the fleet of vehicles of several school transport providers in the said locations.

## Heuristic Algorithm Results

Our heuristic algorithm was coded in C++ and run on a 3.6 GhZ 8-Core Intel Core i9 processor with 8GB RAM. Variants I to IV were run 25 times on each instance using a time limit of 5 min per run. Overall, we found that feasible solutions using the lower bound of $\lceil \sum_{w \in V_2} s(w)/C_{\max} \rceil$ routes were achieved in nineteen of the 20 instances in all runs. The

only instance which required one additional route was the Bridgend instance. This is probably because the pairwise travel times have 25th, 50th, and 75th percentiles approximately equal to 9.4, 14.4, and 18.9 min, respectively. Since these percentiles are relatively high, there is a large chance of violating the maximum riding time constraints (4).

Table 2 displays summary statistics on the results achieved by our algorithm for heterogeneous fleets of buses. Columns 3 to 6 display the average number of iterations performed by each algorithm variant. For the Suffolk instance, approximately 99.9%, 99.9%, 69%, and 81.3% of the iterations performed in Variants I to IV, respectively, produced infeasible solutions, on average. For Cardiff, these percentages were 12.6%, 45.6%, 25.5%, and 33.3%, while those for Bridgend were 13.8%, 6%, 0.1%, and 3.7%. Some infeasible solutions were also produced in Variants II and IV for Porthcawl (0.04% and 0.02%, respectively) and Variant I for Brisbane (0.005%, 1 iteration only). Constraints (1)–(4) were satisfied in all runs for all remaining 15 instances.

According to Table 2, Variant II performs the highest average number of iterations for all instances except Ħamrun and Edinburgh-2. As expected, Variant I performs the lowest average number of iterations for all instances except Porthcawl. This is because this variant does not use information from previous iterations when altering the current subset of bus stops. Hence, local search takes longer in each iteration as it operates on a newly constructed set of

---

[1] For locations in Malta, the available capacities (excluding the driver) are taken to be 8, 14, 16, 18, 20, 36, 44, and 53. Capacities higher than 53 are excluded, since very long buses and double-decker buses are not currently used in Malta for school transport due to unsuitable road infrastructure. For locations in the UK, the assumed capacities are 8, 12, 16, 23, 25, 27, 29, 33, 37, 39, 43, 45, 49, 51, 53, 55, 57, 61, 63, 65, 70, 74, 78, and 80, while those for locations in Australia are 11, 13, 18, 21, 24, 28, 33, 35, 37, 39, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 65, 70, 78, and 80.

**Table 2** Number of iterations performed by our algorithm

| Location | $|\mathcal{R}|$ | Variant I | Variant II | Variant III | Variant IV |
|---|---|---|---|---|---|
| Mġarr | 4 | $29{,}073 \pm 69$ | $75{,}839 \pm 218$ | $66{,}166 \pm 233$ | $69{,}455 \pm 205$ |
| Mellieħa | 4 | $16{,}408 \pm 53$ | $40{,}295 \pm 152$ | $35{,}578 \pm 176$ | $37{,}257 \pm 176$ |
| Porthcawl | 1 | $94{,}639 \pm 346$ | $99{,}574 \pm 349$ | $87{,}701 \pm 2481$ | $94{,}631 \pm 3234$ |
| Qrendi | 5 | $16{,}118 \pm 23$ | $52{,}394 \pm 225$ | $47{,}665 \pm 1446$ | $49{,}906 \pm 178$ |
| Suffolk | 3 | $4524 \pm 24$ | $11{,}086 \pm 70$ | $7610 \pm 329$ | $8180 \pm 317$ |
| Senglea | 6 | $16{,}421 \pm 64$ | $43{,}684 \pm 248$ | $40{,}566 \pm 1292$ | $42{,}074 \pm 1268$ |
| Victoria | 4 | $2211 \pm 16$ | $5066 \pm 75$ | $4383 \pm 301$ | $4616 \pm 225$ |
| Pembroke | 7 | $6594 \pm 10$ | $22{,}950 \pm 69$ | $20{,}075 \pm 828$ | $20{,}876 \pm 757$ |
| Canberra | 7 | $2452 \pm 7$ | $8851 \pm 66$ | $7552 \pm 281$ | $7670 \pm 362$ |
| Ħandaq | 6 | $4566 \pm 38$ | $18{,}381 \pm 202$ | $15{,}584 \pm 797$ | $16{,}624 \pm 970$ |
| Valletta | 6 | $5519 \pm 40$ | $19{,}850 \pm 72$ | $19{,}012 \pm 356$ | $19{,}419 \pm 473$ |
| Birkirkara | 6 | $4436 \pm 21$ | $18{,}185 \pm 107$ | $16{,}826 \pm 474$ | $17{,}014 \pm 463$ |
| Ħamrun | 7 | $2975 \pm 10$ | $10{,}419 \pm 78$ | $10{,}343 \pm 290$ | $10{,}731 \pm 260$ |
| Cardiff | 2 | $16{,}809 \pm 55$ | $26{,}543 \pm 178$ | $26{,}503 \pm 104$ | $26{,}503 \pm 100$ |
| Milton Keynes | 4 | $8919 \pm 11$ | $20{,}670 \pm 59$ | $19{,}722 \pm 469$ | $20{,}097 \pm 382$ |
| Bridgend | 6 | $4186 \pm 18$ | $12{,}163 \pm 51$ | $10{,}622 \pm 110$ | $11{,}053 \pm 109$ |
| Edinburgh-2 | 4 | $5813 \pm 20$ | $5846 \pm 20$ | $5848 \pm 25$ | $5851 \pm 22$ |
| Edinburgh-1 | 9 | $1848 \pm 2$ | $6637 \pm 27$ | $6384 \pm 146$ | $6463 \pm 95$ |
| Adelaide | 8 | $1825 \pm 8$ | $6127 \pm 41$ | $5754 \pm 75$ | $5866 \pm 87$ |
| Brisbane | 10 | $764 \pm 3$ | $3379 \pm 36$ | $3264 \pm 35$ | $3316 \pm 42$ |

All figures are averaged across the 25 runs, rounded to the nearest integer, plus/minus the standard deviation

routes. Another observation is that Variant IV performs a higher average number of iterations than Variant III for all instances except Cardiff. The reason for this is that 50% of the time, Variant IV uses the subset of bus stops from the previous iteration. In this case, local search takes a shorter time than when the subset corresponding to the best feasible solution is used, because fewer changes are made.

Boxplots displaying the 25 total journey times (in min) reached by each algorithm variant for each instance are displayed in Fig. 2. In each boxplot, the horizontal axis represents the total journey time, the endpoints of the whiskers represent the minimum and maximum of all journey times, the red box displays the variability between the 25 and 50th percentiles, and the grey box displays the variability between the 50 and 75th percentiles. Figure 2 indicates that the performance varies significantly across variants for all instances except Mġarr. For the latter instance, all variants provided the same total journey time in all runs. Variants III and IV are seen to perform best for the majority of the instances.

To demonstrate that the results of the four algorithm variants are statistically significantly different, Kruskal–Wallis and post-hoc Bonferroni-adjusted pairwise comparison tests were performed. As expected, significant differences between the variants (all $p$ values $< 0.001$) were revealed for all instances except Mġarr. The pairwise comparison tests indicated that, for 16 instances, the total journey times of Variants I and II are significantly different at the 0.05 level

of significance than those of Variants III and IV. Ten of these instances also saw a significant difference between Variants I and II. Two instances (Porthcawl and Edinburgh-2) saw significant differences between Variant III and all other variants, whereas the Cardiff instance saw significant differences between Variant II and all other variants.

Moving on to Table 3, Columns 5 to 8 display each instance's best total journey time for Variants I to IV, respectively. Each best total journey time corresponds to the endpoint of the left whisker in the boxplot (refer to Fig. 2). Each instance's best reported result across all variants is displayed in bold and the number of runs giving that result is shown in brackets. According to the table, Variants I to IV produce the best reported results in 3, 4, 15, and 12 instances, respectively. Variants I and II seem to be most appropriate for small-sized instances, Variant IV for small-to-moderately-sized instances, while Variant III is more effective for large instances. Furthermore, Variants I to IV produced best total journey times that are at most 13.93%, 10.54%, 4.28%, and 1.31%, respectively, worse than the best reported results. Our best reported result for eleven instances was achieved in only one run, whereas multiple runs reached the best reported result for the other nine instances. Some or all multiple runs for all the latter instances except Porthcawl and Edinburgh-2 also have different corresponding subsets of bus stops. The total number of alternative subsets of bus stops is given in Column 4.
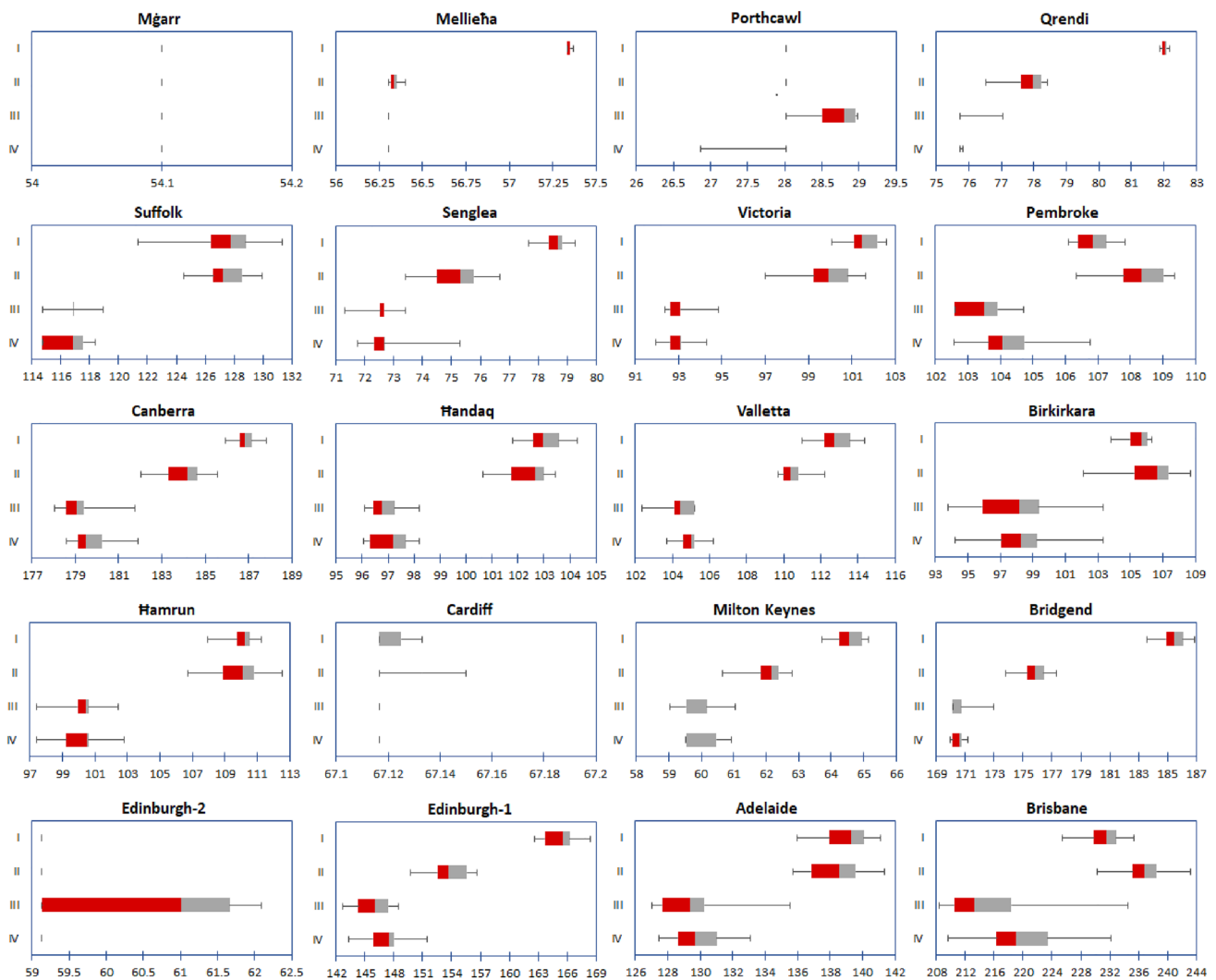
**Fig. 2** Boxplots displaying the performance of the variants on all 20 instances. The horizontal axis in each boxplot represents the total journey time (in minutes)

Column 2 of Table 3 gives the bus capacities assigned to the routes of the best reported result for each instance. The superscript displayed after each capacity indicates the required number of buses of that capacity. It must be pointed out that the capacity assignment of different solutions yielding the best reported result is not necessarily the same. For example, the two solutions for the Senglea instance have different assignments; one uses one 14-passenger bus and five 53-passenger buses, while the other uses one 18-passenger bus, one 44-passenger bus, and four 53-passenger buses. In such a case, the assignment which provides the lowest number of empty seats is presented. This number is displayed in Column 3 of Table 3.

Recall that our algorithm follows the steepest descent (SD) direction, meaning that the best improving move from all neighbourhoods is executed in each iteration. To evaluate whether a change in this strategy would significantly affect the quality of the solutions obtained, the algorithm was altered, such that it follows a variable neighbourhood descent (VND) direction. Specifically, in each iteration, the first improving move from a randomly selected neighbourhood out of the six considered is executed. The process continues until all neighbourhoods do not contain improving moves. As expected, this alteration led to an increase in the average number of iterations performed by each algorithm variant for the majority of the instances. In fact, 18 of the 20 instances saw a percentage increase ranging between 0.44 and 124.88% for all four variants, implying that VND iterations take less time on average. The Porthcawl and Edinburgh-2 instances again stand out from the rest, with the former having percentage increases in the range [− 0.30%, 0.04%] and the latter in the range [− 1.45%, − 1.24%]. When the best total route journey times of the SD and VND strategies were compared, it was observed that these were the

**Table 3** Best total route journey times (in min) achieved by our heuristic algorithm across 25 runs

| Location | Capacities | Empty | Subsets | Heuristic Algorithm | | | | MIP (Gurobi) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Variant I | Variant II | Variant III | Variant IV | Inc. | Improv. | MIP Gap |
| Mġarr | $44^2, 53^2$ | 4 | 8 | **54.1** (25) | **54.1** (25) | **54.1** (25) | **54.1** (25) | 53.17 | 1.73% | 20.91% |
| Mellieħa | $36^1, 44^1, 53^2$ | 15 | 2 | 57.33 | **56.3** (1) | **56.3** (25) | **56.3** (25) | 55.17 | 2.01% | 27.22% |
| Porthcawl | $70^1$ | 4 | 1 | 28.02 | 28.02 | 28.02 | **26.87** (2) | 26.87 | – | 42.49% |
| Qrendi | $53^5$ | 10 | 4 | 81.87 | 76.52 | **75.73** (24) | **75.73** (24) | 74.70 | 1.36% | 46.72% |
| Suffolk | $65^2, 80^1$ | 1 | 3 | 121.33 | 124.52 | **114.73** (1) | **114.73** (6) | 114.73 | – | 50.03% |
| Senglea | $18^1, 44^1, 53^4$ | 8 | 2 | 77.67 | 73.4 | **71.32** (2) | 71.77 | 68.37 | 4.14% | 57.12% |
| Victoria | $36^1, 44^1, 53^2$ | 15 | 1 | 100.08 | 97.02 | 92.38 | **91.95** (1) | 91.95 | – | 64.76% |
| Pembroke | $44^2, 53^5$ | 18 | 1 | 106.08 | 106.33 | 102.6 | **102.58** (1) | 100.97 | 1.58% | 56.36% |
| Canberra | $59^1, 61^1, 78^3, 80^2$ | 15 | 1 | 185.95 | 182.03 | **178.05** (1) | 178.6 | 178.05 | – | 55.05% |
| Ħandaq | $44^1, 53^5$ | 24 | 1 | 101.78 | 100.67 | 96.12 | **96.05** (1) | 96.03 | 0.02% | 61.07% |
| Valletta | $14^1, 53^5$ | 11 | 1 | 110.98 | 109.7 | **102.38** (1) | 103.72 | 102.38 | – | 67.04% |
| Birkirkara | $53^6$ | 12 | 1 | 103.83 | 102.12 | **93.77** (1) | 94.23 | 93.77 | – | 61.66% |
| Ħamrun | $44^2, 53^5$ | 32 | 5 | 107.95 | 106.7 | **97.4** (3) | **97.4** (3) | 97.4 | – | 57.96% |
| Cardiff | $78^1, 80^1$ | 2 | 6 | **67.12** (19) | **67.12** (5) | **67.12** (25) | **67.12** (25) | 67.12 | – | 66.08% |
| Milton Keynes | $53^1, 70^1, 78^2$ | 5 | 1 | 63.72 | 60.65 | **59.03** (1) | 59.53 | 59.03 | – | 52.99% |
| Bridgend | $23^1, 49^1, 78^1, 80^3$ | 9 | 1 | 183.57 | 173.83 | 170.13 | **169.95** (1) | – | – | – |
| Edinburgh-2 | $80^4$ | 0 | 1 | **59.13** (25) | **59.13** (25) | **59.13** (6) | **59.13** (25) | 55.85 | 5.55% | 50.04% |
| Edinburgh-1 | $65^1, 74^3, 78^1, 80^4$ | 5 | 1 | 162.63 | 149.77 | **142.75** (1) | 143.37 | – | – | – |
| Adelaide | $39^1, 65^1, 78^2, 80^4$ | 15 | 1 | 135.98 | 135.72 | **127.02** (1) | 127.47 | – | – | – |
| Brisbane | $65^1, 70^1, 78^4, 80^4$ | 10 | 1 | 225.38 | 230.32 | **208.35** (1) | 209.63 | – | – | – |

Column 9 (Inc.) gives the total route journey time of the best incumbent solution achieved by Gurobi. Column 10 displays the percentage improvement between the best reported result from our heuristic algorithm and the result in Column 9. The last column gives the relative gap between the result in Column 9 and the best-known lower bound on the optimal total route journey time

same for eleven instances. For five instances (Pembroke, Canberra, Valletta, Birkirkara, and Brisbane), the VND's best solution was worse by 0.02–1.86%. On the contrary, the Victoria, Ħandaq, Edinburgh-1, and Adelaide instances saw superior best solutions from the VND strategy by 0.04–1.21%. In addition, a sign test was performed for each instance to determine whether the results of the two strategies are statistically significantly different at the 0.05 level. Significant differences were found for seven instances, with Mellieħa registering better results from VND, and Qrendi, Pembroke, Canberra, Ħandaq, Birkirkara, and Brisbane registering better results from SD. This suggests that, in our case, SD overall performs better than VND.

## Mixed Integer Programming Results

An attempt was also made to improve each instance's best reported result from our heuristic algorithm. For this purpose, a mixed integer programming (MIP) model was formulated for our SBRP, as shown in the appendix. This model was executed using Gurobi 9.1 with a run time limit of 2 h per instance.

For each instance, the best reported solution from the heuristic algorithm was provided as an initial solution. For instances with multiple best solutions, one solution was chosen randomly. Warm-starting the MIP in this way helps to speed up the convergence of branch-and-cut, which is the method used by Gurobi.

The MIP results are presented in Columns 9 to 11 of Table 3. Column 9 gives the total route journey time of the best incumbent solution found. Column 10 displays the percentage improvement between the best reported result from our heuristic algorithm and the best incumbent result. Finally, Column 11 gives the relative MIP optimality gap between the best incumbent result and the best-known lower bound on the optimal total route journey time.

We see that Gurobi was able to make improvements for seven instances and the percentage improvements of these range between 0.02 and 5.55%. On the other hand, Gurobi was not able to improve the heuristic algorithm solution of nine instances within the time limit. For the remaining four instances, the solver could not provide any results before the time limit was reached.

## Comparison to Existing Benchmarks

Further experimentation on the performance of our heuristic algorithm was carried out by applying it to a set of 100 instances generated by Sales et al. [29]. In these instances, the sizes of $V_1 \setminus \{v_0\}$ and $V_2$ range from 25 to 250 bus stops and from 500 to 5250 addresses (or students, as each address is assumed to correspond to one student). Six

assumptions in [29] that differ from the ones we employ in this paper are the following: (i) routes are defined as cycles that start and end at the school, (ii) the heterogeneous fleet is limited in size, (iii) fixed costs are associated with bus usage which vary according to the bus capacities, (iv) driving distance is taken into account instead of driving time (hence, no bus dwell time is incorporated), (v) no maximum limit on the routes' length is considered, and (vi) students are assigned to the stop, within maximum walking distance, that is closest to the school. Moreover, the objective in [29] is to minimize the sum of the total distance travelled by all buses and the fixed costs associated with the buses used.

In [29], the instances were tackled via a memetic algorithm that we describe briefly. Initially, each student is assigned to a stop and the stops with at least one student are identified. An initial population of 100 solutions is constructed by randomly selecting the initial stop for each solution and then, at each step, appending a random stop from the three closest stops to the previously visited stop. Buses are then selected for each solution using the following strategy. If there are $k$ buses with capacities $C_1, C_2, \ldots, C_k$ and fixed costs $F_1, F_2, \ldots, F_k$, then the bus $i \in \{1, 2, \ldots, k\}$ which minimizes $(C_i - D_i)F_i$ is selected, where $D_i$ is the highest accumulated number of students (following the order of the stops in the solution) not exceeding $C_i$. Subsequently, two parent solutions are selected through the binary tournament method, and the edge recombination crossover operator is applied to these solutions to create an offspring. Up to 50 attempts are made to improve the offspring via the lambda interchange local search procedure with $\lambda = 2$. If the offspring is better than the worse parent, then it replaces the latter in the population; if not, the offspring still has a 2% chance of replacing the worse parent. This process is repeated for 5000 iterations.

The following are the changes that were made to our heuristic algorithm to test it on the above-mentioned instances:

- The cost function (6) was changed as in [29]. In particular, we incorporated fixed bus costs and considered driving distances instead of driving times. The bus dwell times and maximum riding times were removed, and routes were also defined as cycles starting and ending at the school, rather than paths starting at a bus stop and ending at the school.
- Students were assigned to the stop, in a selected subset of bus stops, that is closest to the school, rather than to the one that is closest to their residence.
- Given that the fleet size is limited, to verify that an inter-route operator move is feasible, the capacities of the buses performing the two routes as well as the capacities of the unused buses were checked to see

**Table 4** Average total cost, rounded to the nearest integer, achieved by our heuristic algorithm (HA) across 12 runs (3 runs per variant) versus average total cost achieved by Sales et al. [29]'s memetic algorithm (MA) across 10 runs

| ID | $|V_1|$ | $|V_2|$ | $m_w$ | $MA_{avg}$ | $HA_{avg}$ | Gap (%) | ID | $|V_1|$ | $|V_2|$ | $m_w$ | $MA_{avg}$ | $HA_{avg}$ | Gap (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 26 | 500 | 5 | 4745 | 4475 | − 5.69 | 51 | 151 | 3000 | 5 | 26,191 | 24,612 | − 6.03 |
| 2 | 26 | 500 | 10 | 4442 | 4286 | − 3.50 | 52 | 151 | 3000 | 10 | 25,195 | 24,092 | − 4.38 |
| 3 | 26 | 500 | 15 | 4176 | 4062 | − 2.74 | 53 | 151 | 3000 | 15 | 23,364 | 23,229 | − 0.58 |
| 4 | 26 | 500 | 20 | 4012 | 3848 | − 4.09 | 54 | 151 | 3000 | 20 | 22,056 | 22,135 | 0.36 |
| 5 | 26 | 500 | 25 | 3831 | 3808 | − 0.59 | 55 | 151 | 3000 | 25 | 21,293 | 21,249 | − 0.21 |
| 6 | 26 | 750 | 5 | 6630 | 6474 | − 2.35 | 56 | 151 | 3300 | 5 | 28,763 | 27,113 | − 5.74 |
| 7 | 26 | 750 | 10 | 6392 | 6120 | − 4.26 | 57 | 151 | 3300 | 10 | 27,293 | 26,441 | − 3.12 |
| 8 | 26 | 750 | 15 | 6017 | 5894 | − 2.04 | 58 | 151 | 3300 | 15 | 25,614 | 25,325 | − 1.13 |
| 9 | 26 | 750 | 20 | 5726 | 5615 | − 1.94 | 59 | 151 | 3300 | 20 | 24,407 | 24,666 | 1.06 |
| 10 | 26 | 750 | 25 | 5446 | 5495 | 0.89 | 60 | 151 | 3300 | 25 | 23,244 | 23,554 | 1.33 |
| 11 | 51 | 1000 | 5 | 8806 | 8329 | − 5.42 | 61 | 176 | 3500 | 5 | 30,482 | 28,577 | − 6.25 |
| 12 | 51 | 1000 | 10 | 8788 | 8240 | − 6.24 | 62 | 176 | 3500 | 10 | 28,368 | 27,355 | − 3.57 |
| 13 | 51 | 1000 | 15 | 8127 | 7913 | − 2.63 | 63 | 176 | 3500 | 15 | 26,737 | 26,834 | 0.36 |
| 14 | 51 | 1000 | 20 | 7698 | 7544 | − 2.00 | 64 | 176 | 3500 | 20 | 25,812 | 25,834 | 0.09 |
| 15 | 51 | 1000 | 25 | 7279 | 7436 | 2.16 | 65 | 176 | 3500 | 25 | 24,674 | 25,627 | 3.86 |
| 16 | 51 | 1250 | 5 | 10,823 | 10,229 | − 5.49 | 66 | 176 | 3675 | 5 | 32,212 | 30,307 | − 5.91 |
| 17 | 51 | 1250 | 10 | 10,737 | 10,237 | − 4.66 | 67 | 176 | 3675 | 10 | 30,103 | 29,161 | − 3.13 |
| 18 | 51 | 1250 | 15 | 10,078 | 9698 | − 3.77 | 68 | 176 | 3675 | 15 | 28,326 | 28,033 | − 1.03 |
| 19 | 51 | 1250 | 20 | 9563 | 9331 | − 2.43 | 69 | 176 | 3675 | 20 | 26,892 | 26,956 | 0.24 |
| 20 | 51 | 1250 | 25 | 8925 | 9309 | 4.30 | 70 | 176 | 3675 | 25 | 25,840 | 26,485 | 2.49 |
| 21 | 76 | 1500 | 5 | 13,492 | 12,755 | − 5.46 | 71 | 201 | 4000 | 5 | 35,201 | 33,055 | − 6.10 |
| 22 | 76 | 1500 | 10 | 12,767 | 11,973 | − 6.22 | 72 | 201 | 4000 | 10 | 33,024 | 31,797 | − 3.71 |
| 23 | 76 | 1500 | 15 | 12,066 | 11,795 | − 2.24 | 73 | 201 | 4000 | 15 | 31,072 | 30,745 | − 1.05 |
| 24 | 76 | 1500 | 20 | 11,114 | 11,189 | 0.68 | 74 | 201 | 4000 | 20 | 29,030 | 29,216 | 0.64 |
| 25 | 76 | 1500 | 25 | 10,662 | 10,738 | 0.71 | 75 | 201 | 4000 | 25 | 28,218 | 28,550 | 1.18 |
| 26 | 76 | 1725 | 5 | 15,418 | 14,815 | − 3.91 | 76 | 201 | 4200 | 5 | 36,358 | 34,492 | − 5.13 |
| 27 | 76 | 1725 | 10 | 14,573 | 13,795 | − 5.34 | 77 | 201 | 4200 | 10 | 34,607 | 33,604 | − 2.90 |
| 28 | 76 | 1725 | 15 | 13,545 | 13,504 | − 0.30 | 78 | 201 | 4200 | 15 | 32,220 | 32,245 | 0.08 |
| 29 | 76 | 1725 | 20 | 12,899 | 12,903 | 0.03 | 79 | 201 | 4200 | 20 | 30,831 | 31,130 | 0.97 |
| 30 | 76 | 1725 | 25 | 12,184 | 12,734 | 4.52 | 80 | 201 | 4200 | 25 | 29,538 | 29,582 | 0.15 |
| 31 | 101 | 2000 | 5 | 17,496 | 16,510 | − 5.64 | 81 | 226 | 4500 | 5 | 39,110 | 37,111 | − 5.11 |
| 32 | 101 | 2000 | 10 | 16,960 | 16,103 | − 5.05 | 82 | 226 | 4500 | 10 | 36,542 | 35,503 | − 2.84 |
| 33 | 101 | 2000 | 15 | 15,761 | 15,354 | − 2.58 | 83 | 226 | 4500 | 15 | 34,771 | 34,568 | − 0.58 |
| 34 | 101 | 2000 | 20 | 14,825 | 14,783 | − 0.28 | 84 | 226 | 4500 | 20 | 32,735 | 33,106 | 1.13 |
| 35 | 101 | 2000 | 25 | 14,278 | 14,540 | 1.83 | 85 | 226 | 4500 | 25 | 31,810 | 32,087 | 0.87 |
| 36 | 101 | 2300 | 5 | 20,565 | 19,433 | − 5.50 | 86 | 226 | 4725 | 5 | 41,065 | 38,970 | − 5.10 |
| 37 | 101 | 2300 | 10 | 18,966 | 18,160 | − 4.25 | 87 | 226 | 4725 | 10 | 38,897 | 37,661 | − 3.18 |
| 38 | 101 | 2300 | 15 | 17,932 | 17,557 | − 2.09 | 88 | 226 | 4725 | 15 | 36,466 | 36,420 | − 0.13 |
| 39 | 101 | 2300 | 20 | 17,188 | 17,138 | − 0.29 | 89 | 226 | 4725 | 20 | 34,391 | 34,743 | 1.02 |
| 40 | 101 | 2300 | 25 | 16,559 | 17,336 | 4.69 | 90 | 226 | 4725 | 25 | 33,148 | 32,930 | − 0.66 |
| 41 | 126 | 2500 | 5 | 22,199 | 21,077 | − 5.06 | 91 | 251 | 5000 | 5 | 43,289 | 40,925 | − 5.46 |
| 42 | 126 | 2500 | 10 | 20,502 | 19,706 | − 3.88 | 92 | 251 | 5000 | 10 | 40,833 | 39,522 | − 3.21 |
| 43 | 126 | 2500 | 15 | 19,725 | 19,356 | − 1.87 | 93 | 251 | 5000 | 15 | 37,990 | 38,092 | 0.27 |
| 44 | 126 | 2500 | 20 | 18,326 | 18,459 | 0.72 | 94 | 251 | 5000 | 20 | 36,543 | 37,236 | 1.90 |
| 45 | 126 | 2500 | 25 | 17,699 | 18,106 | 2.30 | 95 | 251 | 5000 | 25 | 34,850 | 35,614 | 2.19 |
| 46 | 126 | 2750 | 5 | 24,498 | 23,177 | − 5.39 | 96 | 251 | 5250 | 5 | 46,289 | 43,854 | − 5.26 |
| 47 | 126 | 2750 | 10 | 22,911 | 21,946 | − 4.21 | 97 | 251 | 5250 | 10 | 42,676 | 41,382 | − 3.03 |
| 48 | 126 | 2750 | 15 | 21,643 | 21,411 | − 1.07 | 98 | 251 | 5250 | 15 | 40,083 | 39,986 | − 0.24 |
| 49 | 126 | 2750 | 20 | 20,544 | 20,540 | − 0.02 | 99 | 251 | 5250 | 20 | 38,037 | 38,594 | 1.47 |
| 50 | 126 | 2750 | 25 | 19,429 | 19,163 | − 1.37 | 100 | 251 | 5250 | 25 | 36,751 | 37,480 | 1.98 |

**Table 4**  (continued)                              Relative gaps between the two results are also presented for each instance

whether there are two sufficient capacities for the new loads;

- The creating multistops operator was considered for any pair of distinct routes $R_1$ and $R_2$, attempting each possible transfer amount $z$ between 1 and the difference between the maximum of the capacity of the bus performing $R_2$ and the highest unused bus capacity, and the load of $R_2$;
- In Variants II to IV, whenever the total number $\alpha$ of non-compulsory stops was positive and at most three, the parameters of the Binomially distributed number of removals were changed to $\alpha$ and $1/\alpha$, and the solution alteration process was skipped whenever $\alpha = 0$.

Table 4 displays the results achieved by our heuristic algorithm as well as those achieved by Sales et al.'s [29] memetic algorithm. Note that the latter was run 10 times for each instance and average results are reported. Our results are averaged across 12 runs, with 3 runs per algorithm variant. The computational time allowed for each run of our algorithm was selected as follows. First, instances for which an average computational time of fewer than 10 s was reported in [29] were run for 10 s. For the remaining instances, the average computational time reported in [29] was rounded up to the nearest 30 s and set as a time limit. This was done to have similar running times for comparison purposes.

We observe in Table 4 that the results obtained by our algorithm are very close to those obtained by the memetic algorithm, and sometimes better. The relative gaps between the results range between −6.25 and 4.69%. We have found better results for 68 of the 100 instances, with the largest improvements being seen for Instances 61, 12, and 22. Our algorithm performs the worst in comparison with the memetic algorithm for Instances 40, 30, and 20. Table 4 shows an overall pattern in the relative gaps achieved for different maximum walking distances $m_w$. One can note that, in most cases, the relative gaps increase with an increase in $m_w$. In fact, our algorithm always performs better for $m_w \in \{5, 10\}$. For $m_w \in \{15, 20, 25\}$, our algorithm performs better for 17, 7, and 4 instances, respectively. This trend is likely to have occurred due to the different order of the bus stop selection and student assignment strategies. While our algorithm first selects bus stops and then assigns students to the stop closest to the school, the memetic algorithm performs the student assignments before the selection of the bus stops. Hence, for large $m_w$, as Sales et al. have observed, the memetic algorithm "can concentrate students at stops closer to the school", thus reducing the total distance travelled by all buses.

## Concluding Remarks and Suggestions

This paper has addressed a real-world SBRP that incorporates heterogeneous fleets and bus stop selection. This problem includes several important features, such as student eligibility, maximum walking distances, maximum riding times, different types of buses, multistops (multiple buses visiting a single bus stop), and bus dwell times. A heuristic algorithm has been developed which encompasses the first three subproblems of the SBRP, as defined in Desrosiers et al. [15].

Our heuristic algorithm has been tested on a variety of real-world problem instances from Malta, the UK, and Australia, with sizes upwards of 1800 potential bus stops and 750 students. For all instances, the algorithm has successfully found high-quality solutions in short amounts of computational time. A merit of our algorithm is that it can sometimes provide multiple subsets of bus stops that yield the same best total journey time. Indeed, here, we have determined alternative subsets for seven instances. From these alternative subsets, the most appropriate one can then be identified based on factors, such as bus depot locations, bus stop accessibility, and average student walking distance.

We have attempted to improve the best reported results of our heuristic algorithm through an MIP. An improvement, ranging between 1 and 197 s, has been found for seven of the 20 best results. Higher improvements could potentially be obtained by allowing a longer time limit for each instance.

We have also applied our heuristic algorithm to 100 instances generated by Sales et al. [29], with sizes up to 250 potential bus stops and 5250 students. We have observed that our algorithm compares well with the memetic algorithm presented in [29] and also provides better results for 68 instances.

In the future, we would like to handle uncertainties in the bus travel times caused by factors such as weather conditions and traffic congestion (e.g., [1, 10, 38]). Due to these uncertainties, solutions may no longer remain the best or even feasible, since, for example, students may arrive late for school. It is of interest to see how the solutions will change when we apply a robust optimization and/or a stochastic programming formulation to our SBRP. Another interesting future development is the incorporation of time windows for arrivals at schools and multi-tripping. In the latter, several routes, possibly pertaining to different schools, are linked, so that buses can perform multiple routes successively.

# Appendix

The following MIP model produces solutions consisting of cycles that start and end at the school. The arc from the school to the first bus stop in each route is then excluded. This is made possible by assuming that the driving time from the school to any stop is zero.

The decision variables of our model are as follows. Binary variable $x_{uvR}$ indicates whether route $R \in \mathcal{R}$ travels from $u$ to $v$, where $u, v \in V_1$ and $u \neq v$. Binary variable $y_{vR}$ indicates whether route $R \in \mathcal{R}$ visits $v \in V_1$. In addition, binary variable $z_{wv}$ indicates whether students in address $w \in V_2$ walk to stop $v \in V_1 \setminus \{v_0\}$. Integer variable $s_{vR} \in \{0, 1, \ldots, C_{\max}\}$ gives the number of students boarding route $R \in \mathcal{R}$ at stop $v \in V_1 \setminus \{v_0\}$. Moreover, integer variable $l_{vR} \in \{0, 1, \ldots, C_{\max}\}$ gives the total load of route $R \in \mathcal{R}$ just after visiting stop $v \in V_1 \setminus \{v_0\}$. Finally, the continuous variable $t_R \in [0, m_t]$ specifies the total journey time of route $R \in \mathcal{R}$. The MIP formulation is as follows:

$$\min \sum_{R \in \mathcal{R}} t_R, \tag{8}$$

$$s.t. \sum_{u \in V_1} x_{uvR} = y_{vR} \forall v \in V_1, R \in \mathcal{R}, \tag{9}$$

$$\sum_{u \in V_1} x_{vuR} = y_{vR} \forall v \in V_1, R \in \mathcal{R}, \tag{10}$$

$$y_{v_0 R} \geq y_{vR} \forall v \in V_1 \setminus \{v_0\}, R \in \mathcal{R}, \tag{11}$$

$$\sum_{\substack{v \in V_1 \setminus \{v_0\} \\ d(w,v) \leq m_w}} z_{wv} = 1 \forall w \in V_2, \tag{12}$$

$$\sum_{R \in \mathcal{R}} y_{vR} \geq z_{wv} \forall v \in V_1 \setminus \{v_0\}, w \in V_2, \tag{13}$$

$$\sum_{w \in V_2} s(w) z_{wv} - \sum_{R \in \mathcal{R}} s_{vR} = 0 \forall v \in V_1 \setminus \{v_0\}, \tag{14}$$

$$y_{vR} \leq s_{vR} \forall v \in V_1 \setminus \{v_0\}, R \in \mathcal{R}, \tag{15}$$

$$C_{\max} y_{vR} \geq s_{vR} \forall v \in V_1 \setminus \{v_0\}, R \in \mathcal{R}, \tag{16}$$

$$l_{uR} + s_{vR} - C_{\max}(1 - x_{uvR}) \leq l_{vR} \forall u, v \in V_1 \mid v \neq v_0, R \in \mathcal{R}, \tag{17}$$

$$l_{uR} + s_{vR} + C_{\max}(1 - x_{uvR}) \geq l_{vR} \forall u, v \in V_1 \mid v \neq v_0, R \in \mathcal{R}, \tag{18}$$

$$\sum_{u,v \in V_1} t(u,v) x_{uvR} + \sum_{v \in V_1 \setminus \{v_0\}} (d_1 y_{vR} + d_2 s_{vR}) = t_R \forall R \in \mathcal{R}. \tag{19}$$

Here, the objective function (8) minimizes the total journey time of all routes. Constraints (9)–(11) relate to stop and school visits. Specifically, (9) and (10) ensure that if route $R \in \mathcal{R}$ visits $v \in V_1$, then route $R$ should enter and leave $v$ exactly once. Next, (11) forces each route $R \in \mathcal{R}$ to visit school $v_0$ whenever it visits at least one stop $v \in V_1 \setminus \{v_0\}$. Constraints (12)–(14) relate to student walks and pickups: (12) ensures that students living in each address $w \in V_2$ walk to exactly one stop within walking distance $m_w$; (13) ensures that no student walks to an unvisited stop; while (14) guarantees that the total number of students boarding at stop $v \in V_1 \setminus \{v_0\}$ is equal to the total number of students walking to that stop. Constraints (15)–(16) relate to student boardings. These force the number of students boarding route $R \in \mathcal{R}$ at stop $v \in V_1 \setminus \{v_0\}$ to be zero if route $R$ does not visit stop $v$. If route $R$ visits stop $v$, then (15) also updates the lower bound on the number of boarding students to one. In addition, Constraints (17)–(18) relate to route loads and also serve as subtour elimination constraints as proposed in [20]. Note that $l_{v_0 R} = 0 \ \forall R \in \mathcal{R}$. These guarantee that no route contains a subtour disconnected from school $v_0$ and that each route load increases in accordance with the number of students boarding the bus on that route. In fact, if route $R \in \mathcal{R}$ goes from $u \in V_1$ to stop $v \in V_1 \setminus \{u, v_0\}$, then the load of route $R$ immediately after visiting stop $v$ is set equal to the load of route $R$ just after visiting $u$ plus the number of students boarding route $R$ at stop $v$. Finally, (19) calculates the total journey time of each route $R \in \mathcal{R}$.

## Declarations

# References

1. Babaei M, Rajabi-Bahaabadi M. School bus routing and scheduling with stochastic time-dependent travel times considering on-time arrival reliability. Comput Ind Eng. 2019;138: 106125.
2. Bektaş T, Elmastaş S. Solving school bus routing problems through integer programming. J Oper Res Soc. 2007;58(12):1599–604.
3. Bertsimas D, Delarue A, Martin S. From school buses to start times: driving policy with optimization. 2019.
4. Bodin LD, Berman L. Routing and scheduling of school buses by computer. Transp Sci. 1979;13(2):113–29.
5. Bögl M, Doerner KF, Parragh SN. The school bus routing and scheduling problem with transfers. Networks. 2015;65(2):180–203.
6. Bowerman R, Hall B, Calamai P. A multi-objective optimization approach to urban school bus routing: formulation and solution method. Transport Res Part A. 1995;29(2):107–23.
7. Braca J, Bramel J, Posner B, Simchi-Levi D. A computerized approach to the New York City School Bus Routing Problem. IIE Trans. 1997;29(8):693–702.
8. Braekers K, Ramaekers K, Nieuwenhuyse IV. The vehicle routing problem: state of the art classification and review. Comput Ind Eng. 2016;99:300–13.
9. busprobs. 2018.
10. Caceres H, Batta R, He Q. School bus routing with stochastic demand and duration constraints. Transp Sci. 2017;51(4):1349–64.
11. Campbell JF, North JW, Ellegood WA. Modeling mixed load school bus routing. Quant Approach Logistic Supply Chain Manage. 2015;3–30.
12. Chen X, Kong Y, Dang L, Hou Y, Ye X. Exact and metaheuristic approaches for a bi-objective school bus scheduling problem. PLoS One. 2015;11(4): e0153614.
13. Corberán A, Fernández E, Laguna M, Martí R. Heuristic solutions to the problem of routing school buses with multiple objectives. J Oper Res Soc. 2002;53(4):427–35.
14. Dantzig GB, Ramser JH. The truck dispatching problem. Manage Sci. 1959;6(1):80–91.
15. Desrosiers J, Ferland JA, Rousseau JM, Lapalme G, Chapleau L. An overview of a school busing system. Sci Manage Transp Syst. 1981;235–43.
16. Dror M, Trudeau P. Savings by split delivery routing. Transp Sci. 1989;23(2):141–5.
17. Dror M, Trudeau P. Split delivery routing. Nav Res Logist. 1990;37(3):383–402.
18. Dulac G, Ferland JA, Forgues PA. School bus routes generator in urban surroundings. Comput Oper Res. 1980;7(3):199–213.
19. Ellegood WA, Solomon S, North J, Campbell JF. School bus routing problem: contemporary trends and research directions. Omega. 2020;95:1–18.
20. Kek AGH, Cheu RL, Meng Q. Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots. Math Comput Model. 2008;47(1–2):140–52.
21. Laporte G, Nobert Y, Taillefer S. Solving a family of multi-depot vehicle routing and location-routing problems. Transp Sci. 1988;22(3):161–72.
22. Lewis R, Smith-Miles K. A heuristic algorithm for finding cost-effective solutions to real-world school bus routing problems. J Discrete Algor. 2018;52–53:2–17.
23. Li LY, Fu Z. The school bus routing problem: a case study. J Oper Res Soc. 2002;53(5):552–8.
24. Lima FM, Pereira DS, Conceição SV, Ramos Nunes NT. A mixed load capacitated rural school bus routing problem with heterogeneous fleet: algorithms for the Brazilian context. Expert Syst Appl. 2016;56:320–34.
25. Newton RM, Thomas WH. Design of school bus routes by computer. Socio Econ Plann Sci.1969; 75–85.
26. Park J, Kim BI. The school bus routing problem: a review. Eur J Oper Res. 2010;202:311–9.
27. Park J, Tae H, Kim B-I. A post-improvement procedure for the mixed load school bus routing problem. Eur J Oper Res. 2012;217(1):204–13.
28. Riera-Ledesma J, Salazar-González J-J. Solving school bus routing using the multiple vehicle traveling purchaser problem: a branch-and-cut approach. Comput Oper Res. 2012;39(2):391–404.
29. Sales LD, Melo CS, Bonates TD, Prata BD. Memetic algorithm for the heterogeneous fleet school bus routing problem. J Urban Plann Dev. 2018;144(2):04018018.
30. Sansone K. No free school transport for 1,000 students in church and independent schools. MaltaToday. 2018. https://www.maltatoday.com.mt/news/national/89608/no_free_school_transport_for_1000_students_in_church_and_independent_schools#.YuO46i0RqCg.
31. Sariklis D, Powell S. A heuristic method for the open vehicle routing problem. J Oper Res Soc. 2000;51(5):564–73.
32. sbrpMaltaHeterogeneousFleets. 2021.
33. Schittekat P, Kinable J, Sörensen K, Sevaux M, Spieksma F, Springael J. A metaheuristic for the school bus routing problem with bus stop selection. Eur J Oper Res. 2013;229(2):518–28.
34. Schittekat P, Sevaux M, Sorensen K. A mathematical formulation for a school bus routing problem. In: 2006 international conference on service systems and service management. 2006. p. 1552–7.
35. Sciortino M, Lewis R, Thompson J. A heuristic algorithm for school bus routing with bus stop selection. Evolutionary Computation in Combinatorial Optimization, Springer Lecture Notes in Computer Science. 2021;12692:202–18.
36. Siqueira VS, Silva EN, Silva RV, Rocha ML. Implementation of the metaheuristic GRASP applied to the school bus routing problem. Int J E-Educ E-Bus E-Manage E-Learn. 2016;6(2):137–45.
37. Spada M, Bierlaire M, Liebling TM. Decision-aiding methodology for the school bus routing and scheduling problem. Transp Sci. 2005;39(4):477–90.
38. Sun S, Duan Z, Xu Q. School bus routing problem in the stochastic and time-dependent transportation network. PLoS One. 2018;13(8): e0202618.