A Heuristic Algorithm for Finding Cost-Effective Solutions to Real-World School Bus Routing Problems

R. Lewis[1] and K. Smith-Miles[2]

[1]School of Mathematics, Cardiff University, Cardiff, CF24 4AG, Wales.
[2]School of Mathematics and Statistics, University of Melbourne, VIC 3010, Australia.,
lewisR9@cf.ac.uk, smith-miles@unimelb.edu.au

October 2, 2018

Abstract

This paper proposes a heuristic algorithm for designing real-world school transport schedules. It extends previously considered problem models by considering some important but hitherto overlooked features including the splitting and merging of routes, gauging vehicle dwell times, the selection of stopping points, and the minimisation of walking distances. We show that this formulation contains a number of interacting combinatorial subproblems including the time-constrained vehicle routing problem, set covering, and bin packing. As a result, a number of new and necessary algorithmic operators are proposed for this problem which are then used alongside other recognised heuristics. Primarily, the aim of this algorithm is to minimise the number of vehicles used by each school, though secondary issues concerning journey lengths and walking distances are also considered through the employment of suitable multiobjective techniques.

Keywords: School Bus Routing; Vehicle Routing; Set Covering; Bin Packing.

1 Introduction

This paper considers the problem of arranging school bus transport using a generic model that considers both the costs incurred by the provider and the quality of service experienced by users. The problem was originally provided to us by decision makers in local government in Wales [21], although very similar problems are also faced in other countries such as England and Australia [8, 32, 29].

In this problem school transport is organised by local government. In Wales, for instance, county councils have a legal duty to provide free school transport to any child who attends their nearest suitable school but who also lives more than 4.8 km (3 miles) away, measured according to the shortest safe walking route. For children under eleven this distance is reduced to 3.2 km (2 miles). In England and Australia similar rules apply, though these thresholds can sometimes change slightly for low-income families, people living in remote areas, and for those with mobility problems.

A few months before the start of the school year, administrators within local government compile a list of addresses of all children deemed eligible for school transport. Schools are then considered individually, and a list of potential pick-up points for the relevant addresses are identified. Pick-up points may be municipal bus stops or other areas deemed appropriate such as a lay-by, a car park, or an actual home address. However, students should not be expected to walk too far to these pick-up points—in Wales and Victoria, for example, these should be within 1.6 km (1 mile) of a student's home.

Once all addresses and potential pick-up points have been chosen, local government administrators then draw up a set of bus routes for each school. This set of routes can use a subset of pick-up points, but must serve all qualifying students. Some bus routes may also overlap by visiting the same pick-up points, although mixed loads—that is, students from different schools travelling on the same bus—are not permitted.

Finally, the constructed bus routes are put out for public tender and local bus companies are invited to bid for the contracts. In the UK, a yearly contract for a 70-seat bus typically ranges from GBP£25,000 to £35,000, though these costs can increase further for longer journeys and for routes requiring a chaperone (i.e., routes with young children). It is therefore critical for local government to try to limit the number of buses used by each school. Note, however, that guidelines also specify that ride times should not be too long—in Wales, for example, this is stated to be less than 45 minutes for under elevens and less than one hour for older students, though exceptions can sometimes be made for schools with very large catchment areas.

2 Literature Review

School bus routing problems belong to a wider family of vehicle routing problems (VRPs), which involve identifying routes for a fleet of vehicles that are to serve a set of customers. VRPs can be expressed using an edge-weighted directed graph $G = (V, E)$, where the vertex set $V = \{v_0, v_1, \ldots, v_n\}$ represents a single depot and n customers (v_0 and $v_1, \ldots, v_n$ respectively), and the weighting function $d(u, v)$ gives the travel distance (or travel time) between each pair of vertices $u, v \in V$.

Since the work of Datzig and Ramser in the late 1950s [7], a multitude of VRP formulations have been considered in the literature. These include using time windows for visiting certain customers, placing limitations on the lengths of individual routes, the partitioning of customers into pick-up and delivery locations, and the dynamic recalibration of routes subject to the arrival of new customer requests during the transportation period [17, 25]. Solutions to most VRP problems can be expressed by a set of routes $\mathcal{R} = \{R_1, \ldots, R_k\}$ using one vehicle per route. In the *classical* VRP, each route should be a simple cycle in G such that:

$$R_i \cap R_j = \{v_0\} \quad \forall R_i, R_j \in \mathcal{R} \tag{1}$$

$$\bigcup_{i=1}^{k} R_i = V. \tag{2}$$

These constraints specify that each customer should be assigned to exactly one route, and that all routes should start and end at the depot v_0. A variation on this is the *open* VRP in which, instead of using cycles, all routes must be simple *paths* containing v_0 as one terminal vertex, meaning that routes either start or end at the depot, but not both [19]. In the *time constrained* VRP, extra realism is added by specifying that the total weight of edges in each route should be less than a given maximum—e.g., to ensure that driving time regulations are obeyed. In the *capacitated* VRP, meanwhile, maximum capacities are specified for each vehicle, and weights are also added to the vertices $v_1, \ldots, v_n$ in G. These vertex weights represent the size of the items being delivered to each customer, and the total size of items delivered by each vehicle should not exceed its maximum capacity.

Objective functions for the VRP can depend on many factors. Most commonly we seek to minimise the number of vehicles used, the total length of the routes, or some combination of the two. In other cases we might also be concerned with the waiting times of customers, the obeying of time windows, avoiding traffic jams, or meeting individual drivers' needs. A useful survey presenting a taxonomy of the various types of VRP is due to Eksioglu et al. [11].

The problem of arranging school transport has often been cited as a type of VRP applicable in the real-world, though historically it has been less studied than other variants. One reason for this is that school transport solutions often involve visiting only a subset of the available pick-up points (bus stops); hence the issue of choosing *which* bus stops to visit adds an extra layer of complexity to the problem. Indeed, in their survey paper, Park and Kim note that bus stop selection is often omitted in the VRP literature altogether [23]. Of those papers that have considered this issue, Park and Kim make note of two general decomposition schemes. The first involves first choosing a subset of stops, assigning students to these stops, and then routing the buses via these stops. An obvious disadvantage of this approach is that, if the wrong choice of bus stops is made, the resultant solution may be far from optimal. The second scheme involves first allocating students to clusters such that the number of students per cluster does not exceed the available vehicle capacities. One bus route for each cluster is then formed using a subset of stops from that cluster, and finally students are assigned to bus stops within their cluster.

In the past decade or so the interdependence of bus stop selection and routing has led to the proposal of algorithms that consider these issues simultaneously. Riera-Ledesma and Salazar-González [27] consider these issues in a number of integer programming (IP) formulations that attempt to minimise a solution's cost, calculated as the sum of all route lengths and all walk lengths. Their experiments generally focus on artificial problem instances where the combined number of bus stops and addresses equals one hundred, although their algorithm is shown to be able to achieve optimal solutions for slightly larger problems in a small number of cases. A similar formulation is also considered by Schittekat et al. [28] who use a local search-based method to approximately solve artificially generated problems of up to eighty potential bus stops and eight hundred students. In their case, the algorithm seeks to minimise the total distance travelled by all vehicles. In most cases students are assigned to the route that passes closest to their address, but this can be overridden on occasion to facilitate more efficient in-vehicle seating allocations. This problem formulation was also used by Kinable et al. [16] who developed an approach based on column generation. This was shown to be capable of solving problems of up to forty bus stops and eight hundred students, though the authors also note that problems with larger numbers of bus stops present much more difficulty to the algorithm.

A notable feature of the three papers mentioned in the previous paragraph is that all routes in a solution are required to be disjoint—that is, bus stops can only be visited once in a solution. In practice, this requirement can be somewhat restrictive. In some cases, the number of students assigned to a bus stop might exceed the available bus capacity; in others, an early and late bus might be offered to students. Often it may be sufficient to schedule two or more buses to follow the same bus route, one after the other; however, extra flexibility is also available if routes are permitted to split and merge as required. In our experience, this is often the approach that local government administrators follow when

Term	Description
m_{w}	Maximum distance a student is required to walk from their home address to a bus stop.
m_{t}	Maximum length (in time) of a school bus route. Typically 45 minutes or one hour.
m_{e}	Minimum walking distance between a school and student address for the student to be deemed eligible for school transport.
Q	Bus seating capacity.
V_1	Set of vertices comprising one school, v_0, together with n bus stops $v_1, \ldots, v_n$ that can potentially be used in a solution.
$V_1 - \{v_0\}$	Set of n bus stops that can potentially be used in a solution.
V_2	Set of addresses of students eligible for school transport.
$t(u, v)$	If $u, v \in V_1$, denotes the driving time between u and v; if $u \in V_2$ and $v \in V_1$, denotes the walking time between u and v.
$d(u, v)$	If $u, v \in V_1$, denotes the driving distance between u and v; if $u \in V_2$ and $v \in V_1$, denotes the walking distance between u and v.
E_1	Edge set containing directed edges between each $u, v \in V_1$ in each direction.
E_2	Edge set containing all walking routes between addresses and bus stops that are less than m_{w} distance units in length.
$\mathcal{R}$	A set of *routes* $\mathcal{R} = \{R_1, \ldots, R_k\}$ in which each route $R \in \mathcal{R}$ is a simple path among bus stopsserved by a single bus.
V_1'	$V_1' \subseteq (V_1 - \{v_0\})$: the subset of bus stops used by the solution $\mathcal{R}$.
$V_2(v)$	The subset of addresses assigned to bus stop $v \in V_1'$.
$s(v)$	If $v \in V_1'$, denotes the number of students assigned to bus stop v; if $v \in V_2$, denotes the number of students at address v.
$s(R)$	Total number of students assigned to the bus serving route R.
$s(v, R)$	Number of students required to board the bus serving route R at bus stop $v \in V_1'$.
$t(R)$	Total journey time of route R including all dwell times (see Definition 2).
$\mathcal{A}$	Archive set of solutions used with the multiobjective optimisation algorithm (Section 5).
δ	Discretisation parameter used with the multiobjective optimisation algorithm.
$\lfloor \cdot \rfloor_\delta$	Denotes the value between the braces rounded down to the nearest multiple of δ.
f_1, f_2	Cost functions used with the multiobjective optimisation algorithm.

Table 1: Summary of the notation used for the SBRP considered in this paper.

planning school transport [21].

In the VRP literature, problems that allow routes to split and merge are known as split delivery VRPs. This variant extends the capacitated VRP by relaxing Constraint (1) to simply: $v_0 \in R, \forall R \in \mathcal{R}$, allowing more than one vehicle to visit a customer and therefore permitting a delivery to be made in many parts. In contrast to the capacitated VRP, this relaxation allows the minimum number of routes/vehicles in a solution to meet the lower bound of $\lceil \left(\sum_{i=1}^{n} w(v_i)\right) / Q \rceil$, where $w(v)$ gives the weight of a vertex and Q is the maximum capacity of the vehicles. Dror and Trudeau [9], who originally formulated the problem note that significant savings can be made by allowing split deliveries; indeed, in the best case they can result in solutions using just half the number of routes compared to the capacitated VRP [2]. Dror and Trudeau [10] have also shown that for any problem whose distances satisfy the triangle inequality, there exists an optimal solution where no pair of routes has more than one customer in common. A helpful review of this problem is due to Archetti et al. [3].

In the next section we specify the particular school bus routing problem faced by local governments in the UK and Australia. This formulation extends previous work in this problem area by considering a number of additional real-world features including the splitting and merging of routes, measuring dwell times, the selection of stopping points, and the minimisation of walking distances. We also analyse this problem by noting relationships with other difficult combinatorial problems such as bin packing, set covering, and the VRP itself. In Section 4 we then draw on these relationships to give a specialised heuristic algorithm that attempts to find feasible solutions for this problem using the minimum number of routes. This algorithm, which involves various new search operators, is designed particularly to cope with practical-sized problems, which can often contain many hundreds of bus stops and students. Section 5 then shows how multiobjective techniques can be used for making further adjustments to solutions in order to minimise their financial costs and maximise convenience to passengers. This latter analysis focusses on real-world problem instances created for this research, whose interactive solutions can be viewed online [1]. Finally Section 6 concludes the paper and conducts further discussions. A summary of the notation used in this paper is given in Table 1.

3 Problem Definition and Analysis

The school bus routing problem (SBRP) considered here can be stated using two sets of vertices. The first vertex set V_1 contains one school, v_0, together with n bus stops $v_1, \ldots, v_n$ that can potentially be used in a solution. The edge set E_1 then contains directed edges between each $u, v \in V_1$ in each direction, making the graph (V_1, E_1) a complete digraph. For each edge $(u, v) \in E_1$, nonnegative weighting functions $t(u, v)$ and $d(u, v)$ are used to define the shortest driving time and corresponding driving distance (respectively) from u to v.

The second vertex set V_2 defines the set of student addresses, with the weight $s(v) \in \mathbb{Z}^+$ of each vertex $v \in V_2$ giving the number of students at this address requiring school transport. A parameter m_{w} is also defined, stating the maximum distance that students are expected to walk from their home address to a bus stop. A second set of edges is then used to signify bus stops within walking distance of each address: $E_2 = \{\{u, v\} : u \in V_2 \ \wedge \ v \in (V_1 - \{v_0\}) \ \wedge \ d(u, v) \leq m_{\mathrm{w}}\}$. Here, $d(u, v)$ gives the shortest walking distance between each $u \in V_2$ and $v \in (V_1 - \{v_0\})$. These

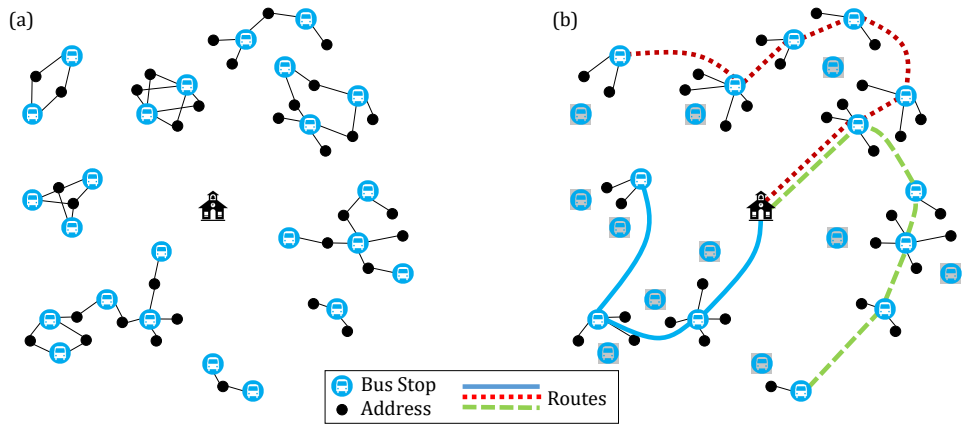

Figure 1: (a) An example problem instance; (b) an example solution using $k = 3$ routes. Grey bus stops in (b) are not used (i.e., are not members of V_1').

edges also have a corresponding walking time $t(u, v)$. In some cases, further edges can also be removed from E_2 if they signify a walk that is deemed unsafe by local administrators. In addition, students living within m_e distance units of their school are not considered eligible for school transport; consequently, $d(u, v_0) \geq m_e \; \forall u \in V_2$.

The graph $(V_1 - \{v_0\}, V_2, E_2)$ therefore constitutes an undirected bipartite graph with potentially many components, as illustrated in Figure 1(a). Note that if there exists an address $u \in V_2$ with just one incident edge $\{u, v\} \in E_2$, then the bus stop $v \in (V_1 - \{v_0\})$ is *compulsory*, since it must be included in a solution in order to satisfy the needs of address u. We can also assume that $(V_1 - \{v_0\}, V_2, E_2)$ contains no isolated vertices: such vertices in $(V_1 - \{v_0\})$ would give a bus stop with no address within walking distance and can therefore be removed from the problem; isolated vertices in V_2 define an address with no suitable bus stop, making the problem unsolvable (in practice, an additional bus stop would need to be added to serve such an address).

Definition 1. *A* feasible *solution to the SBRP is a set of* routes $\mathcal{R} = \{R_1, \ldots, R_k\}$ *in which each route* $R \in \mathcal{R}$ *is a simple path served by a single bus of capacity* Q*, and where each bus travels to the school* v_0 *after visiting the terminal vertex on its path. The following constraints need to be satisfied:*

$$\bigcup_{i=1}^{k} R_i = V_1' \tag{3}$$

$$\forall u \in V_2 \quad \exists v \in V_1' : \{u, v\} \in E_2 \tag{4}$$

$$s(R) \leq Q \quad \forall R \in \mathcal{R} \tag{5}$$

$$t(R) \leq m_t \quad \forall R \in \mathcal{R} \tag{6}$$

Here $V_1' \subseteq (V_1 - \{v_0\})$ *is the subset of bus stops used by the solution.* V_1' *should satisfy Constraint (4) in that, for each address* $u \in V_2$*,* V_1' *should contain at least one bus stop within walking distance. Constraint (5) then specifies that the total number of students boarding the bus on a route* R*, denoted by* $s(R)$*, does not exceed the maximum bus capacity* Q*. Similarly, Constraint (6) states that the total journey time* $t(R)$ *of each route should not exceed the stated time limit* m_t*.*

Due to the high costs of yearly bus contracts noted in Section 1, the overriding aim of the SBRP is to produce a feasible solution that minimises the number of routes k. Once this has been achieved, other secondary objectives such as minimising journey times and walking distances can then also be considered (see Section 5). An example feasible solution for this problem is shown in Figure 1(b), where we see that each address is adjacent to a used bus stop as required. Note that the constraints in Definition 1 also allow bus stops to be included in more than one route, as is the case with one bus stop in the diagram. We call these bus stops *multistops*.

3.1 Additional Problem Features

In defining this SBRP, four additional real-world features also need to be considered.

- First, in contrast to the model used in [28], students are always assigned to the bus stop in V_1' closest to their home. This encourages equity of service, and avoids situations where a student might be forced to walk to a more distant bus stop (perhaps past classmates who are waiting at a closer bus stop), in order to access school transport.

- Second, all students are given a bus pass that only allows them to travel on one particular route. This avoids situations where too many students might board a bus at a multistop, making it too full to serve another bus stop later in its route.
- Third, solutions for this problem only concern buses travelling *to* school. After-school routes are assumed to follow the same paths in reverse, with any discrepancies in travel time due to one-way streets, etc. not being considered. This means that students on a route who have the longest journey times in the morning also have the longest journey times in the evening.

Finally, *dwell times* of buses also need to be accounted for within a route. These measure the time spent servicing each bus stop, including decelerating, opening doors, loading passengers, and rejoining the traffic stream. Dwell times are influenced by many factors including the number of boarding passengers, the size and position of the doors, the age of the passengers, the type of bus stop, and the density of traffic. Commonly, simple linear models $y = a + bx$ are used to estimate a dwell time y, where x gives the number of boarding passengers, b gives the boarding time per passenger, and a captures all remaining delays. We follow this approach here, giving the following:

Definition 2. *Let $s(u_i, R)$ denote the number of students boarding the bus on route R at bus stop u_i. The journey time $t(R)$ of route $R = (u_1, u_2, \ldots, u_l) \in \mathcal{R}$ is calculated as:*

$$t(R) = \left(\sum_{i=1}^{l-1} t(u_i, u_{i+1}) \right) + t(u_l, v_0) + \left(\sum_{i=1}^{l} a + b \cdot s(u_i, R) \right). \tag{7}$$

In our case we use the values $a = 15$ and $b = 5$ (seconds), which are consistent with those recommended in [5, 31, 33].

3.2 Problem Generation

For this research we consider both artificially generated and real-world problem instances. A set of artificial instances was previously made available in [28], though we cannot use this here for a number of reasons: (a) the largest instances in this set are too small for our purposes, containing just 80 bus stops; (b) travel times between locations are not specified; (c) values for m_{e} are not considered; and (d) in many cases, allowable walking distances from homes to bus stops are far larger than the driving distances from bus stops to the school, which is not a realistic feature of the SBRP.

In our case, artificial problem instances were generated by placing a school at the centre of a circle with radius $r > m_{\mathrm{e}}$ km. Bus stops were then randomly placed within this circle, followed by a set of addresses, ensuring that each address was at least m_{e} km from the school, but within m_{w} km of at least one bus stop. In all cases, we assume that distances are Euclidean, vehicles travel at a constant speed of 50 km/h, and students walk at 5 km/h. The number of students at each address was generated randomly according to the following distribution: 1 (45%), 2 (40%), 3 (14%), and 4 (1%), which approximates the relevant statistics in the UK and Australia.

We also used our own software to generate a number of real-world problem instances. Privacy requirements prevent us from being able to use real students' addresses; instead, each instance was constructed using the following process. First, the location of a school and its catchment area was identified using public records. Random residential addresses were then generated within the local area, but at least m_{e} km from the school, and a number of students were assigned to these addresses according the distribution above. Next, all bus stops in the local vicinity were added while ensuring that (a) each address had at least one stop within walking distance, and (b) each stop had at least one address within walking distance. The locations of these bus stops were also determined using public records. Finally, the shortest driving times and distances between each pair of bus stops, and shortest walking times and distances between all bus stops and addresses were determined using web mapping services (either Google Maps or Bing Maps). Where possible, the direction of travel from each bus stop was also taken into account to ensure that buses are not required to make illegal driving manoeuvres, such as U-turns on inappropriate roads.

A summary of these ten problem instances is given in Table 2. These locations were chosen to reflect a wide range of problem features including problem size (in terms of school enrolment figures and the number of bus stops in their catchment areas), urban or rural settings, coastal and inland, and organic or grid city layouts. Note in particular that central urban environments feature large numbers of bus stops per address and addresses per stop, reflecting higher population and bus stop densities. Visualisations of two instances are given in Figure 2. All instances can be viewed and downloaded at [1].

3.3 Problem Analysis

In this section we now make some observations about the complexity of the SBRP and its underlying subproblems, which will help to inform our algorithm design in later sections.

Theorem 1. *The task of finding a feasible solution with a minimum number of routes is NP-hard.*

Location	Country/State	$\lvert V_1\rvert - 1$	$\lvert V_2\rvert$	Students	m_e	m_w	Stops per address	Addresses per stop
Brisbane	Queensland	1817	438	757	3.2	1.6	79.2	19.1
Adelaide	South Australia	1118	342	565	1.6	1.6	100.0	28.8
Edinburgh-1	Scotland	959	409	680	1.6	1.6	84.6	36.1
Edinburgh-2	Scotland	917	190	320	1.6	1.6	256.1	53.1
Bridgend	Wales	633	221	381	4.8	1.6	45.3	15.9
Milton Keynes	England	579	149	274	4.8	1.6	64.5	16.6
Cardiff	Wales	552	90	156	4.8	1.6	63.9	10.4
Canberra	ACT	331	296	499	4.8	1.0	13.3	11.9
Suffolk	England	174	123	209	4.8	1.6	10.4	7.4
Porthcawl	Wales	153	42	66	3.2	1.6	53.5	14.8

Table 2: Summary of the ten real-world problem instances, listed in order of the number of available bus stops ($|V_1|-1$). Recall that V_2 is the set of student addresses, m_e is the minimum distance from the school for which students are deemed eligible for school transport, and m_w is the maximum walking distance to a bus stop. All distances are given in km.

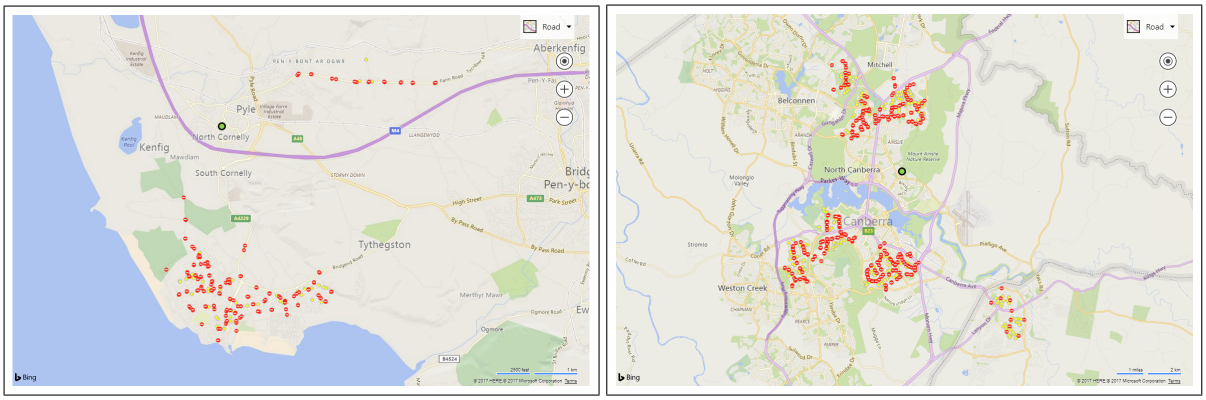

Figure 2: Problem instances for Porthcawl, Wales (left) and Canberra, Australia (right). Red icons show bus stops, yellow icons show addresses, and the school is shown in green.

Proof. Let (V_1, V_2, E_2) be a graph such that $\deg(u) = 1\ \forall u \in V_2$. This means that, for all bus stops $v \in (V_1 - \{v_0\})$, (a) v is compulsory and must appear in at least one route, and (b) the number of boarding students is fixed at $\sum_{\forall u \in \Gamma(v)} s(u)$. This also implies the dwell times at each bus stop are fixed. This special case is equivalent to the NP-hard time-constrained capacitated split-delivery VRP, itself a generalisation of the NP-hard time-constrained VRP [7]. □

A similar proof of NP-hardness considers a generalisation of the above in which each component in $(V_1 - \{v_0\}, V_2, E_2)$ is a complete bipartite graph. In this case, all students can be assigned to a bus by including in V_1' exactly one bus stop from each component, making the problem a multi-vehicle version of the NP-hard generalised travelling salesman problem [18].

As stated, the primary aim in the SBRP is to minimise the number of routes (buses) being used in a solution. It is therefore desirable to efficiently fill buses where possible, bringing parallels with the NP-hard bin packing problem [12]. Indeed, if multistops were not permitted in a solution, then the identification of a solution using k routes while obeying Constraints (3)–(5) would result in a one-dimensional bin packing problem with bin capacity Q and item sizes equal to the number of students boarding at each bus stop. As noted, multistops *are* permitted in this SBRP meaning that students boarding at a particular bus stop can be assigned to different routes if needed (or, equivalently, items in the corresponding bin packing problem can be split across different bins). This allows us to always produce a solution $\mathcal{R} = \{R_1, \ldots, R_k\}$ satisfying Constraints (3)–(5) and meeting the lower bound of $k = \lceil (\sum_{i=1}^{n} s(v_i))\,/Q \rceil$, though of course these routes could be rather long, perhaps causing Constraint (6) to be violated.

A second bin packing problem also arises once a final feasible solution $\mathcal{R}$ has been produced. Let $V_2(v) \subseteq V_2$ be the set of addresses assigned to a particular bus stop $v \in V_1'$. If v appears in just one route $R \in \mathcal{R}$, then all students from addresses in $V_2(v)$ will be assigned to R; hence, $s(v, R) = \sum_{u \in V_2(v)} s(u)$. On the other hand, if v is a multistop serving routes $R_1, \ldots, R_l$, then a decision also needs to be made about which addresses in $V_2(v)$ to assign to which routes. In general it is desirable to assign students from the same address to the same route (e.g., to keep siblings together); however, ensuring this happens is equivalent to solving an instance of the NP-complete variable sized bin packing problem [13] using items sizes $\{s(u) : u \in V_2(v)\}$ and bin sizes $s(v, R_1), \ldots, s(v, R_l)$. In practice it might therefore be necessary to sometimes assign students from the same address to different routes, although this should generally be avoided if possible.

From a different perspective, the issue of choosing which subset V_1' of bus stops to include in a solution is closely related to the set covering problem. Set covering involves taking a set of integers $U = \{1, 2, ..., n\}$ known as the "universe" and a set S whose elements are subsets of the universe. We then wish to identify the smallest subset $S' \subseteq S$ whose union equals the universe—i.e., S' needs to "cover" U. For example, given $U = \{1, 2, 3, 4\}$ and $S = \{\{1\}, \{1, 2\}, \{1, 3\}, \{3, 4\}, \{4\}\}$ the optimal covering is $S' = \{\{1, 2\}, \{3, 4\}\}$, which contains just two elements. The following definitions are now helpful.

Definition 3. *Given U and S, $S' \subseteq S$ is a* complete *covering if and only if $\bigcup_{s \in S'} = U$. A* minimal *covering is a complete covering in which the removal of any element in S' results in an incomplete covering. An* optimal covering *is a minimum cardinality solution among all minimal coverings.*

In general, identifying an optimal covering in the set covering problem is NP-hard [14]; however, approximate solutions can be found using the greedy algorithm of Chvatal [6], which operates by selecting the element of S containing the largest number of uncovered elements, adding this to S', and repeating until a complete covering is achieved. The task of constructing minimal coverings is also easily carried out in polynomial time. For example, starting with an arbitrary complete covering S', at each step we might simply remove any element $s \in S'$ for which $(S' - \{s\})$ is still a complete covering, repeating until S' is minimal.

With regards to the SBRP, using the bipartite graph $(V_1 - \{v_0\}, V_2, E_2)$, let S be the set whose elements correspond to the addresses within walking distance of each bus stop, $S = \{\Gamma(v) : v \in (V_1 - \{v_0\})\}$, where $\Gamma(v)$ is the set of vertices adjacent to v. According to Constraint (4), all addresses in a feasible solution must be served by a bus stop; hence, the task of identifying a subset $V_1' \subseteq (V_1 - \{v_0\})$ meeting this criterion is equivalent to the problem of finding a complete covering of the universe V_2 using the set S.

We are now able to make the following statement concerning minimal coverings.

Theorem 2. *Consider the SBRP in which multistops are not permitted (i.e., $R_i \cap R_j = \emptyset \ \forall R_i, R_j \in \mathcal{R}$), and let (V_1, E_1) be a graph whose pairwise distances satisfy the triangle inequality. Now let $\mathcal{R} = \{R_1, \ldots, R_k\}$ be a solution satisfying Constraints (3)–(5) that has the minimum total journey time $\sum_{i=1}^{k} t(R_i)$. Then the subset of bus stops V_1' used in $\mathcal{R}$ corresponds to a minimal covering S'.*

Proof. The removal of any bus stop $v \in V_1'$ corresponds to the removal of the element $\Gamma(v)$ in S' which, by definition, results in an incomplete covering and violation of Constraint (4). Conversely, the addition of an extra element $\Gamma(v)$ to S' will result in a complete but non-minimal covering; however, this corresponds to the addition of an extra bus stop v in at least one route in $\mathcal{R}$ which, due to the triangle inequality, will maintain or increase the total journey time of $\mathcal{R}$. □

Note that this theorem does not hold when multistops are permitted in a solution. This is because the addition of an extra bus stop may allow a route to be shortened by redirecting it from a multistop and through this new stop. The triangle inequality is also necessary for this theorem, though this requirement is acceptable here because it is satisfied by both real-world road maps (in which minimum distances/times between each pair of locations are used) and Euclidean graphs. Indeed, because of the extra delays incurred by dwell times in this SBRP, this inequality can actually be strengthened to $\forall u_1, u_2, u_3 \in V_1, t(u_1, u_2) + t(u_2, u_3) > t(u_1, u_3)$.

4 Finding a Feasible Solution

In this section we describe a heuristic algorithm that attempts to find a feasible solution for the SBRP using a minimal number of routes (Sections 4.1 to 4.3). In Section 4.4 we then test this algorithm on a large number of artificially generated instances. Our strategy here is to use a fixed number of routes k and allow the violation of Constraint (6) while ensuring that the remaining constraints, (3) to (5), are always satisfied. Specialised operators are then used to try to shorten the routes in a solution such that Constraint (6) also becomes satisfied, giving a feasible solution. If this cannot be achieved at a certain computation limit, k is increased by one, and the algorithm is repeated. Initially, k is set to the lower bound $\lceil (\sum_{i=1}^{n} s(v_i)) / Q \rceil$. An alternative approach would be to allow the search operators themselves to alter k and then use its value as part of the cost function. However, evidence from the literature for similar partition-based problems suggests the former to usually be a more suitable approach [20, 26].

Our method for this task is similar to iterated local search in that it combines local search (Section 4.2) with a more disruptive "kick" operator (4.3). Our local search routine operates with a fixed subset of bus stops $V_1' \subseteq (V_1 - \{v_0\})$ using neighbourhood operators that make small changes to the solution that can be evaluated quickly. In contrast, changes to V_1' tend to be much more disruptive, with potentially many passengers having to be reassigned to different bus stops and multiple routes having to be repaired. As a result, these changes are best carried out by an operator that is periodically applied between applications of local search.

In our algorithm two options are also available for defining the solution space of bus stop subsets. The first option is to allow V_1' to be any subset of $(V_1 - \{v_0\})$ that satisfies Constraints (3) and (4) (that is, any subset of stops

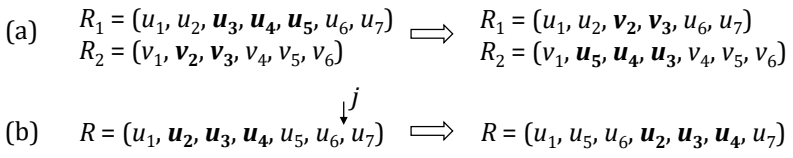

Figure 3: Example application of (a) the Section Swap operator (here, the section u_3, u_4, u_5 has been inverted before insertion into R_2); and (b) the Extended Or-opt operator.

corresponding to a complete covering of V_2). The second, more restricted, option is to only allow V_1' to correspond to minimal coverings in an attempt to eliminate solutions with overly long routes from being considered. Both of these options are explored in more detail below.

4.1 Initial Solution and Cost Function

An initial solution $\mathcal{R} = \{R_1, \ldots, R_k\}$ is constructed by first generating a subset of bus stops V_1' corresponding to a complete covering. In our case, this is achieved using Chvatal's greedy heuristic (Section 3.3). If the more restricted solution space is being used, appropriate bus stops can then also be randomly selected and removed from V_1' until the covering is seen to be minimal.

Having generated V_1', each student is assigned to their closest bus stop in V_1', determining $s(v)$ $\forall v \in V_1'$. We then need to assign each of these bus stops to an appropriate route (vehicle) such that no route is overfilled: that is, we have a bin packing problem with a set of "item weights" $\{s(v) : v \in V_1'\}$ and k "bins" of capacity Q. To do this, we use a variant of the first-fit descending heuristic for bin packing. Specifically, at each step, the bus stop v with the largest number of boarding students is chosen. A suitable route R is then chosen, and v is added to the end of the route. A route is selected for v according to two criteria: (a) R already contains v, and (b) R has adequate spare capacity for v (because $Q - s(R) \geq s(v)$). If more than one route satisfies both criteria, ties are broken by selecting the one containing the fewest students; similarly, if no routes are seen to satisfy (a), then ties between those satisfying (b) are broken in the same way.[1] If criterion (b) is not satisfied, then it is evident that no single route has sufficient capacity to serve all students boarding at stop v. In this case the route R with the largest spare capacity $x = Q - s(R)$ is chosen and v is assigned to this route along with x of its students. This has the effect of creating a multistop, since a copy of bus stop v, along with its remaining students will also need to be assigned to a different route in a subsequent iteration.

On completion, this bin packing-based heuristic will have produced a solution $\mathcal{R}$ obeying Constraints (3)–(5). It is then evaluated according to the cost function $f(\mathcal{R}) = \sum_{i=1}^{k} t'(R)$, where

$$t'(R) = \begin{cases} t(R) & \text{if } t(R) \leq m_{\mathrm{t}} \\ m_{\mathrm{t}} + W(1 + t(R) - m_{\mathrm{t}}) & \text{otherwise.} \end{cases} \tag{8}$$

Here, W introduces a penalty cost for routes whose journey times exceed m_{t}. In our case we set W to m_{t} and include the addition of one in the formula to ensure that a route with $t(R) > m_{\mathrm{t}}$ is always penalised more heavily than two routes with individual journey times of less than m_{t}.

4.2 Local Search

As mentioned, our local search routine makes a series of changes to a solution to reduce its cost while ensuring that Constraints (3)–(5) are never violated. As with many other VRP variants, our local search routine uses a combination of both inter- and intra-route neighbourhood operators. The following inter-route operators act on two routes $R_1, R_2 \in \mathcal{R}$. Without loss of generality, assume that $R_1 = (u_1, u_2, \ldots, u_{l_1})$ and $R_2 = (v_1, v_2, \ldots, v_{l_2})$.

Section Swap: Take two vertices in each route, u_{i_1}, u_{i_2} $(1 \leq i_1 \leq i_2 \leq l_1)$ and v_{j_1}, v_{j_2} $(1 \leq j_1 \leq j_2 \leq l_2)$, and use these to define the sections $u_{i_1}, \ldots, u_{i_2}$ and $v_{j_1}, \ldots, v_{j_2}$ within R_1 and R_2 respectively. Now swap the two sections, inverting either if this leads to a superior cost. (The inversion of a section involves simply reversing its ordering—see Figure 3(a) for an example.)

Section Insert: Take a section in R_1, defined by u_{i_1} and u_{i_2} as above, together with an insertion point j $(1 \leq j \leq l_2 + 1)$ in R_2. Now remove the section from R_1 and insert it before vertex v_j in R_2, inverting the section if this leads to a better cost. If $j = l_2 + 1$, add the section to the end of R_2.

Create Multistop: Let $u \in R_1$ for which $s(u, R_1) \geq 2$, and let $s(R_2) < Q$. If $u \notin R_2$, add a copy of u to R_2. Now, transfer some of the students that board bus stop u in R_1 to the occurrence of u in R_2.

[1] Note that when building an initial solution, criterion (a) will never be met; however, we include it in our description here because it is also used when rebuilding a solution from a partial solution during the kick operator, described in Section 4.3.

Note that the first two inter-route operators can result in too many students being assigned to R_1 or R_2, leading to a violation of Constraint (5). In our case, such moves are disallowed. Since multistops are permitted, it is also possible that applications of these operators will result in a route that contains a vertex $v \in V_1'$ more than once. Since each route must be a simple path, these repetitions need to be deleted. Assuming without loss of generality that a new route is to be produced by inserting the section $u_{i_1}, \ldots, u_{i_2}$ (possibly inverted) into a route $R = (v_1, v_2, \ldots, v_{l_2})$, and that some vertex in the section is already present in R, this is done by simply removing the relevant vertex from the section and reassigning its students to the other occurrence of the vertex in R.

Under the triangle inequality and using our method of calculating dwell times, the addition of an extra stop due to the Create Multistop operator means that R_2 will always be lengthened more than R_1 is shortened. However, due to the weighting coefficient used in Equation (8), it will still reduce a solution's cost if $t(R_1) > m_{\mathrm{t}}$ and $t(R_2) < m_{\mathrm{t}}$ before its application, but both $t(R_1) \leq m_{\mathrm{t}}$ and $t(R_2) \leq m_{\mathrm{t}}$ after its application. In our case, if u is not already present in R_2, then it is inserted into the route at the position that causes the smallest increase in $t(R_2)$. If $t(R_1) \leq m_{\mathrm{t}}$, then just one passenger is transferred to R_2—the intention being to lengthen R_2 by as small an amount as possible. Otherwise $t(R_1) > m_{\mathrm{t}}$, in which case we shorten R_1 as much as possible by transferring $\min(s(u, R_1) - 1, Q - s(R_2))$ students to R_2. This ensures that bus stop u has at least one student boarding in both R_1 and R_2, but that neither route is overfilled.

Our three intra-route neighbourhood operators act on a single route $R = (u_1, u_2, \ldots, u_l) \in \mathcal{R}$. Note that their application does not affect the satisfaction of Constraints (3)–(5), nor do they introduce duplicate vertices into a route.

Swap: Take two vertices u_{i_1}, u_{i_2} $(1 \leq i_1 \leq i_2 \leq l)$ in R and swap their positions.
2-opt: Take two vertices u_{i_1}, u_{i_2} $(1 \leq i_1 \leq i_2 \leq l)$ and invert the section $u_{i_1}, \ldots, u_{i_2}$ within R.
Extended Or-opt: Take a section defined by u_{i_1} and u_{i_2} as above, together with an insertion point j outside of this section (i.e., $1 \leq j < i_1$ or $i_2 + 1 < j \leq l + 1$). Now remove the section and insert it before vertex u_j. If $j = l + 1$, then add the section to the end of the route. Also, invert the section if this leads to a better cost (see Figure 3(b)).

Together, our six operators generalise a number of neighbourhood operators commonly featured in the literature. For example, our first two inter-route operators include and extend the six outlined for the capacitated VRP in [30]. They also extend the three neighbourhood operators used for school bus routing in [28]. Our Extended Or-opt operator also generalises the more basic Or-opt, which only involves sections of up to three vertices [4].

Here, our local search procedure follows the steepest descent methodology: in each cycle all moves in all neighbourhoods are evaluated, and the move offering the largest reduction in cost is performed, breaking ties randomly. The procedure terminates when no improving moves are identified. Note that the number of moves considered in each cycle is $O(m^4)$, where $m = \sum_{i=1}^{k} |R_i|$ is the size of the incumbent solution. Of course, it would also possible to use more sophisticated search methodologies here, such as simulated annealing or tabu search. However, the intention here is to quickly identify a local optimum, which is then passed on to the other complementary parts of the overall algorithm.

4.3 Generating New Coverings via a Kick Operator

While our local search routine is able to alter and improve the cost of a solution, note that it does not alter the subset V_1' of bus stops being used. One way of doing this, as suggested in [28], is to either swap a bus stop in a route with a currently unused bus stop, or simply remove a bus stop from a route altogether. However, besides not allowing the number of bus stops in V_1' to increase, this is unsuitable for this problem because it fails to ensure the satisfaction of Constraint (4).

In order to make use of alternative bus stops not currently in V_1', we introduce a *kick operator*. Given a feasible solution $\mathcal{R}$ using a subset of bus stops V_1', this works by forming a new subset of bus stops $V_1'' \neq V_1'$ that also satisfies Constraint (4). Repairs are then made to $\mathcal{R}$ so that it only uses the bus stops in V_1''. To form V_1'', a randomly chosen non-compulsory bus stop $v \in V_1'$ is first removed from V_1', followed by x additional non-compulsory bus stops.[2] If the result is a partial covering, then some new bus stops need to be added. This is done by selecting bus stops from the set $(V_1 - \{v_0, v\})$ using a randomised version of Chvatal's heuristic that, at each stage, arbitrarily selects and adds any bus stop that will serve some currently unserved students, and repeats until all students are served. This provides a new complete covering V_1'', as required. If the option of only allowing minimal coverings has been chosen, appropriate randomly selected bus stops can then also be removed until the covering is seen to be minimal.

Having produced a new subset of bus stops $V_1'' \neq V_1'$, each address in V_2 is then reassigned to its closest bus stop in V_1''. Bus stops with no addresses assigned to them are then removed from $\mathcal{R}$ and V_1''. Next, the UPDATE-SOLUTION procedure in Figure 4 is invoked. As shown, this removes any bus stops from $\mathcal{R}$ that are not included in V_1'' (Step (10)). If fewer students are assigned to the bus stop $v \in V_1''$ than in V_1', changes to $\mathcal{R}$ are also made in Step (8). Primarily though, UPDATE-SOLUTION is used for constructing a list of bus stops and associated students that need to be added

[2] In our case a value for x is selected randomly according to a binomial distribution $X \sim B(|V_1'|, 3/|V_1'|)$.

	UPDATE-SOLUTION $(\mathcal{R}, V_1', V_1'')$
(1)	**for each** bus stop $v \in (V_1 - \{v_0\})$
(2)	**if** $v \in V_1''$ **then**
(3)	Let $s(v)'$ be the number of students assigned to v in V_1'
(4)	Let $s(v)''$ be the number of students assigned to v in V_1''
(5)	**if** $s(v)' < s(v)''$ **then**
(6)	Add v, together with $s(v)'' - s(v)'$ students, to the list of stops that need to be added to $\mathcal{R}$
(7)	**else if** $s(v)' > s(v)''$ **then**
(8)	Remove $s(v)' - s(v)''$ students from occurrences of v in $\mathcal{R}$. If this results in an occurrence of v with no assigned students, then remove it from $\mathcal{R}$
(9)	**else if** $v \in V_1'$ **then**
(10)	Remove all occurrences of v in $\mathcal{R}$

Figure 4: Procedure for updating a solution $\mathcal{R}$ according to an old and new subset of bus stops, V_1' and V_1''.

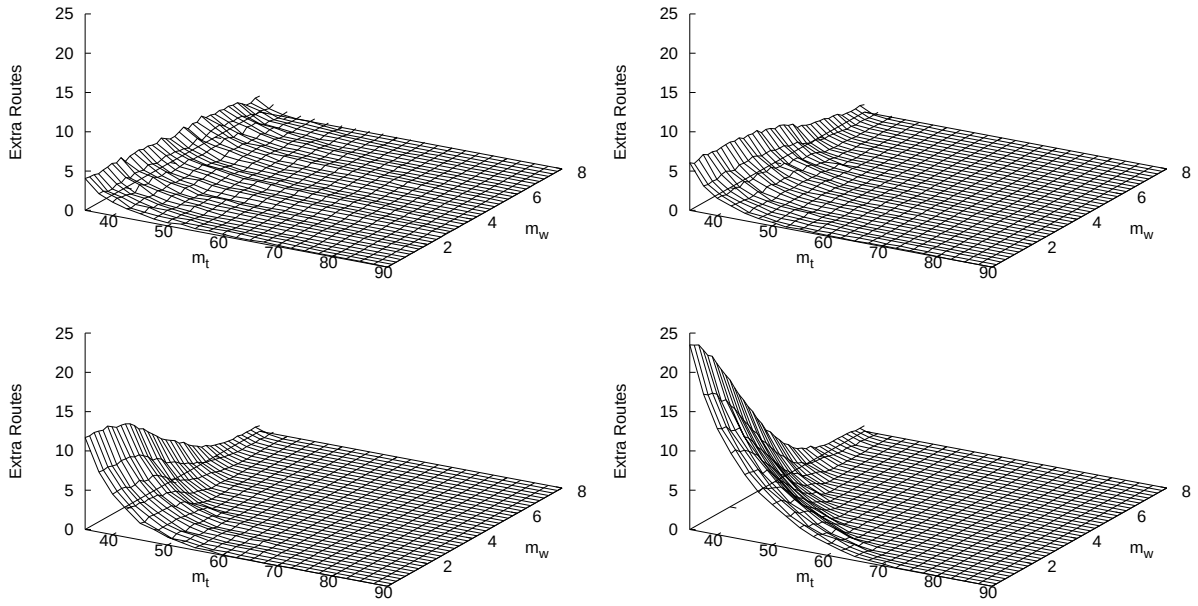

Figure 5: Number of extra routes (buses) required by solutions for various journey time limits m_{t} (in minutes) and walking distance limits m_{w} (in km), for problem instances involving $|V_1 - \{v_0\}| = 25$, 50, 100 and 250 bus stops respectively. Each point is averaged across twenty randomly generated problem instances.

to the solution, as shown in Step (6). On termination of this procedure, the bus stops in this list are added to $\mathcal{R}$ using the same bin packing heuristic as used in Section 4.1.

4.4 Experimentation

In this section we use our algorithm to examine the issues that affect the number of extra routes (buses) required in a solution compared to the lower bound of $\lceil (\sum_{i=1}^{n} s(v_i)) / Q \rceil$. To do this, artificial problem instances were generated with 25, 50, 100 and 250 bus stops using maximum walking distances m_{w} ranging from of 0.2 to 8.0 km, incrementing in steps of 0.2 km. Twenty instances were produced in each case, giving 3,200 in total. In all cases we used a radius r of 25 km and set the number of addresses at 585, giving approximately 1,000 students per instance. Finally, the maximum bus capacity Q and minimum eligibility distance m_{e} were set to 70 and 5 km respectively, with thirty seconds of execution time permitted for each value of k.[3]

Figure 5 shows the effect of altering the maximum journey time m_{t} with these different problem instances. We see that an increased number of routes are needed when both the maximum journey times m_{t} and the maximum walking distances m_{w} are low, and that these increases are more drastic for instances with many bus stops. For low values for m_{t} this is quite natural: stricter journey limits imply the need for additional routes in feasible solutions. For low values of m_{w} on the other hand, our method of instance generation means that addresses will be clustered tightly around bus stops; consequently, nearly all bus stops will be compulsory, making the problem very similar to

[3] All algorithms were written in C++ and executed on 3.2 GHtz Windows 7 machines with 8 GB RAM [1].

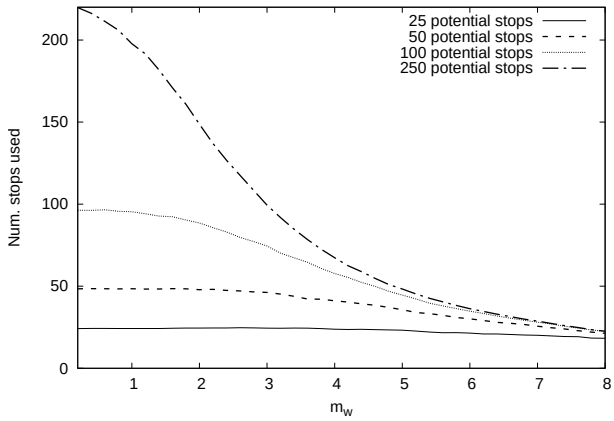

Figure 6: Number of bus stops used in solutions for instances containing $|V_1 - \{v_0\}| = 25$, 50, 100 and 250 potential bus stops, for various values of m_{w}. Each point is averaged across twenty problem instances.

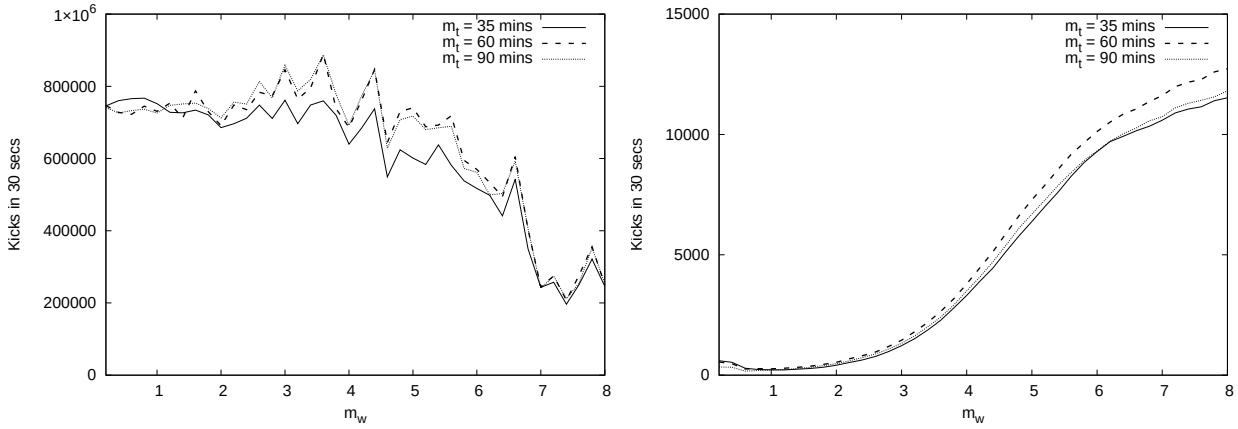

Figure 7: Effects that the maximum walking distance m_{w} has on the number of kicks performed by our algorithm within a thirty second time limit for instances with $|V_1 - \{v_0\}| = 25$ and 250 potential bus stops (respectively), for three values of maximum journey time m_{t}. All points are averaged across twenty randomly generated problem instances. Note also that the second graph's vertical axis has a much smaller range.

that described in Theorem 1. This means that any savings that can be achieved by only using a subset of the bus stops are not available, creating a need for additional routes in a solution. These effects also increase for larger numbers of bus stops because, for low values of m_{w} in particular, more bus stops need to be visited, resulting in a need for more routes. The relationship between m_{w} and the number of bus stops used in the final solutions is shown in Figure 6.

Considering multistops, we find that these occur more frequently with instances where it is advantageous or necessary to assign larger numbers of students to individual bus stops. This occurs for high values of m_{w}, where students are able to walk larger distances to bus stops (implying that fewer bus stops are needed in V_1'), or when the number of bus stops in the problem instance is small. From a bin packing perspective, more students per stop equates to larger items to pack into the bins, meaning that more of these items will need to be "split", resulting in a multistop.

Note that the results in Figure 5 were generated using the restricted solution space in which only bus stop subsets V_1' corresponding to minimal coverings are permitted. We also carried out this same set of experiments using the more expansive solution space in which V_1' can be any complete covering. In most cases, we saw no difference between the two approaches in terms of the average number of extra routes needed. However, solutions with marginally fewer routes were sometimes returned using the latter space for instances with 25 or 50 bus stops, large values of m_{w}, and short journey times. As explained above, solutions to these instances tend to feature larger numbers of multistops, meaning that Theorem 2 does not apply; hence, the additional coverings contained in this space allow the algorithm to identify better subsets of bus stops and therefore achieve better solutions. On the other hand, superior solutions were consistently achieved using the more restricted space when tackling instances with 100 or 250 bus stops, where multistops occur less frequently.

Moving on, Figure 7 demonstrates the number of kicks—and therefore applications of local search—performed by our algorithm within the thirty second time limit used for each value of k. For instances with 25 potential bus stops, this number is highest for instances where addresses are tightly clustered around the bus stops (low values for m_{w}). In

these cases most, if not all bus stops are compulsory and so the kick operator can only make very small changes to a solution, if any. As a result, kicked solutions will already be close to a local optimum meaning that the subsequent local search routine will complete very quickly. On the other hand, for instances with 250 bus stops the sizes of solutions $\sum_{i=1}^{k} |R_i|$ are much larger, meaning that applications of local search will be more expensive, allowing fewer kicks within the time limit. This is particularly the case for clustered instances, where a high proportion of the stops are used in solutions: indeed, in the most extreme case, only 170 kicks were performed within the 30 second time limit, suggesting that longer run times may be beneficial in some cases.

On the whole, the results of this section demonstrate that, at least for these artificial instances, our algorithm is often able to find solutions using the lower bound of $\lceil (\sum_{i=1}^{n} s(v_i)) / Q \rceil$ routes. This is particularly so for instances where only a small proportion of bus stops need to be used, such as when the maximum walking distance of students is set quite high. Note, however, that in cases where all bus stops need to be used, our proposed kick operator has no effect, and it would probably be better to focus on extending the local search operator by, for example, including a tabu element.

One notable feature of this approach is that, in attempting to find the solutions with short routes, the set of students is often assigned to a relatively small number of bus stops, rather than using more convenient bus stops that are closer to student homes. In the next section, we will investigate how we might improve these solutions so their effects on users are also taken into account.

5 Improving Financial Costs and Quality of Service

Once a feasible solution using as few routes as possible has been achieved, it is in our interest to make further adjustments to allow the service to be both cost efficient for the local government and convenient for the users. From the financial point of view, administrators are interested in keeping the lengths of each route as short as possible because this will attract lower quotes from the bidding bus companies. On the other hand, users want to be assigned to bus stops close to their homes, keeping walks short and discouraging parents from driving to the bus stop. Clearly these objectives are in conflict, because a bus route that visits many stops (allowing shorter walks) will also tend to be longer and therefore more expensive to run. Decision makers are therefore interested in looking at a set of non-dominating solutions that, for a fixed number of vehicles k, shows how these objectives influence one another, allowing them to choose a solution seen to be an appropriate compromise of the two. This naturally points to the suitability of multiobjective-based optimisation methods.

The two cost functions that we seek to optimise at this stage are the average time spent walking by each student:

$$f_1(\mathcal{R}) = \frac{\sum_{u \in V_2} s(u) \cdot t(u, v^*)}{\sum_{u \in V_2} s(u)} \tag{9}$$

(where $v^* \in V_1'$ is the closest used bus stop within walking distance of address u), and the average driving time of the routes:

$$f_2(\mathcal{R}) = \frac{\sum_{i=1}^{k} t(R)}{k}. \tag{10}$$

Note, therefore, that f_2 does not measure the travel times of individual students as such. Instead, it acts as a proxy for the eventual financial cost of the routes, which will be unknown at this point.

5.1 A Multiobjective Local Search Approach

Our strategy here is to make incremental adjustments to the subset of used bus stops V_1', thereby changing the assignments of students to bus stops and causing an alteration to f_1. Solutions are then repaired and optimised using our local search procedure from Section 4.2, hopefully reducing f_2. A suitable framework for this is the Pareto local search algorithm of Paquete et al. [22]. This generalises a single-objective local search algorithm in that it returns an archive solution set that approximates the actual Pareto set of a problem instance. It is also quite intuitive in that it uses no parameters and requires no aggregation of the different cost functions.

A pseudocode description of this approach is given in Figure 8. The algorithm starts with an archive set $\mathcal{A}$ of solutions, each that are marked as "unvisited". In each iteration an unvisited solution $\mathcal{R}$ is then selected from the archive set. New solutions $\mathcal{R}'$ are then generated from $\mathcal{R}$ by adding or deleting bus stops, repairing, and then employing local search. These solutions are added to $\mathcal{A}$ if they are not dominated by a solution already in the archive. In addition, if a new solution is seen to dominate any solutions currently in the archive, then these are deleted from $\mathcal{A}$. This repeats until all solutions in $\mathcal{A}$ have been visited, resulting in an archive of mutually nondominating solutions.

In Line (6) of this procedure a bus stop v is added to the solution $\mathcal{R}'$ in the following way. First, all students for whom v is now their closest bus stop are assigned to v.[4] Bus stops that, as a result, have no assigned students are then

[4]We assume here that v has at least one student for whom this would now be their closest bus stop. If this is not the case, then the resultant solution need not be considered further, as v will not be used by any passengers.

```
MULTIOBJECTIVE-LOCAL-SEARCH (A)
(1) while ∃R ∈ A that has not yet been visited
(2)     Mark R as having been visited
(3)     for each v ∈ (V_1 − {v_0})
(4)         R' ← R
(5)         if v ∉ V_1'
(6)             Insert v into R' and run local search
(7)         else if v ∈ V_1' and v is not compulsory
(8)             Remove v from R' and run local search
(9)         UPDATE-ARCHIVE(A, R')
```

```
UPDATE-ARCHIVE (A, R')
(1) for each R ∈ A
(2)     if R' dominates R
(3)         Remove R from A
(4)     else if ⌊f_1(R')⌋_δ ≥ ⌊f_1(R)⌋_δ and ⌊f_2(R')⌋_δ ≥ ⌊f_2(R)⌋_δ
(5)         exit
(6) Mark R as not visited, and add it to A
```

Figure 8: Our multiobjective procedure. Here, V_1' corresponds to the set of bus stops being used in solution $\mathcal{R}$. The notation $\lfloor \cdot \rfloor_\delta$ denotes the value within the braces rounded down to the nearest multiple of δ.

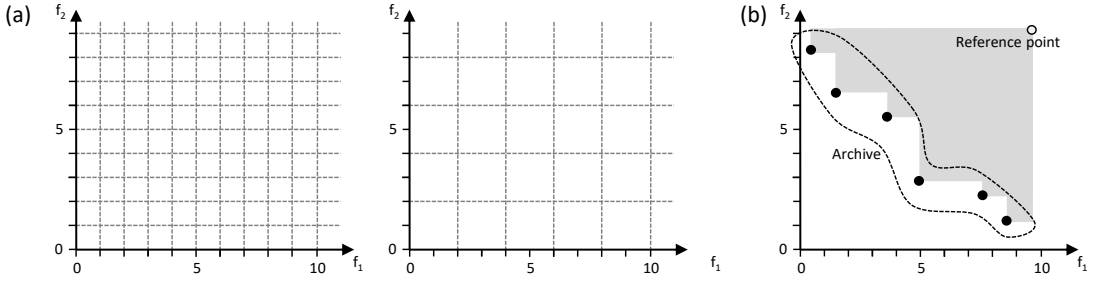

Figure 9: (a) Illustrating the discretization of the cost space for settings of $\delta = 1$ (left) and $\delta = 2$ (right). Archives feature a maximum of one solution per square. If $\delta = 0$ then archive sizes are not limited. (b) Demonstration of the S-metric using an archive set of mutually nondominating solutions and an appropriate reference point. The value of the S-metric corresponds to the hypervolume indicated by the shaded part.

removed from $\mathcal{R}'$. Finally, v is inserted into one or more route using the bin packing procedure from Section 4.1, and local search is used to improve the solution.

In a similar fashion, in Line (8) a non-compulsory bus stop v is removed from $\mathcal{R}'$. Each address $u \in V_2$ is then considered in turn and reassigned to the closest bus stop within walking distance that is still being used in $\mathcal{R}'$. If no such bus stop exists for u, then the bus stop $v' \neq v$ nearest to their house is added to the end of a randomly selected route in $\mathcal{R}'$ and all relevant passengers are reassigned to u. Once all addresses have been considered, this gives a solution obeying Constraints (3) and (4), but perhaps not Constraint (5). In this case, randomly chosen bus stops are removed from any overfilled routes until Constraint (5) has been satisfied. The solution is then repaired using our bin packing heuristic before executing local search.

In their original paper, Paquete et al. [22] note that run times of this multiobjective framework can be very high for problems featuring large archive sets. This is because each solution in the archive needs to be visited, which may then result in further unvisited solutions being added to the archive, and so on. This was also seen to be the case with our algorithm, with larger problem instances such as Brisbane and Edinburgh-1 sometimes taking days to complete. To alleviate this, our UPDATE-ARCHIVE procedure of Figure 8 contains an additional mechanism for limiting the archive size. This appears in Lines (4) and (5), where the costs of the solutions $\mathcal{R}'$ and $\mathcal{R}$ are rounded down to the nearest multiple of δ before being compared (where $\delta \geq 0$ is a parameter specified by the user). This means that when $\mathcal{R}'$ is mutually nondominating to all solutions in $\mathcal{A}$, it is still only added to $\mathcal{A}$ if it is seen to have costs sufficiently different to solutions already in $\mathcal{A}$. From a different perspective, we can consider this mechanism as a way of discretizing the cost space into a grid of squares whose boundaries correspond to multiples of δ, as demonstrated in Figure 9(a). Since only one solution per square is permitted, larger values for δ result in smaller archive sets. This brings shorter run times but also, by reducing the number of solutions considered during a run, potentially less accurate approximations of the Pareto set. These issues are considered in the next subsection.

		$\delta = 1$			$\delta = 10$			$\delta = 30$		
Location	k	Time (s)	$\lvert\mathcal{A}\rvert$	$S \pm$ CV	Time (s)	$\lvert\mathcal{A}\rvert$	$S \pm$ CV	Time (s)	$\lvert\mathcal{A}\rvert$	$S \pm$ CV
Brisbane	11	31444.0	399.3	$256.2 \pm 1.0\%$	2329.1	20.2	$217.4 \pm 3.4\%$	539.3	2.9	$150.6 \pm 7.3\%$
Adelaide	9	9326.8	350.5	$341.3 \pm 0.5\%$	1191.5	22.5	$315.8 \pm 1.3\%$	125.1	3.5	$250.7 \pm 4.2\%$
Edinburgh-1	10	16155.9	314.9	$337.5 \pm 0.5\%$	906.5	19.9	$308.8 \pm 1.4\%$	159.1	3.2	$257.0 \pm 3.2\%$
Edinburgh-2	5	582.5	313.0	$431.6 \pm 0.4\%$	135.7	38.1	$421.3 \pm 0.3\%$	17.7	11.1	$394.5 \pm 1.7\%$
Bridgend	6	2743.7	296.2	$140.5 \pm 2.5\%$	518.3	24.1	$128.2 \pm 3.2\%$	332.9	4.8	$103.0 \pm 7.3\%$
Milton Keynes	5	491.6	318.9	$303.9 \pm 0.6\%$	108.8	34.9	$293.0 \pm 1.0\%$	36.2	10.2	$257.0 \pm 2.3\%$
Cardiff	3	7.1	172.4	$308.1 \pm 0.4\%$	2.4	38.1	$305.8 \pm 0.6\%$	1.2	13.4	$299.8 \pm 0.9\%$
Canberra	8	356.0	150.8	$233.4 \pm 0.6\%$	32.8	8.8	$224.3 \pm 0.9\%$	11.6	3.3	$215.5 \pm 1.2\%$
Suffolk	4	28.6	127.5	$185.4 \pm 2.7\%$	8.8	17.6	$180.0 \pm 2.7\%$	5.9	5.6	$171.4 \pm 3.0\%$
Porthcawl	1	4.4	127.5	$152.6 \pm 1.7\%$	0.9	43.4	$149.1 \pm 2.9\%$	0.5	18.1	$141.7 \pm 3.3\%$

Table 3: Results produced for the ten real-world problem instances using our multiobjective approach for three values of δ. S-values were calculated using the reference point (15.0, 45.0) in all cases. All figures in the table are averages taken from twenty runs per instance. CV is the coefficient of variation.

5.2 Experiments

In this section we investigate the quality of solutions produced by our multiobjective algorithm using the ten real-world problem instances listed in Table 2, bus capacities $Q = 70$, and maximum journey times $m_{\text{t}} = 45$. For nine of these instances, all versions of our algorithm from Section 4 were able to achieve feasible solutions using the lower bound of $k = \lceil (\sum_{i=1}^{n} s(v_i)) / Q \rceil$ routes in less than three seconds. The one exception, Suffolk, which represents a wide sparsely populated rural area, seems to require one additional vehicle. (Although, interestingly, if we use $m_{\text{t}} = 50$ with this instance, this extra bus is not needed.)

For our trials, the archive sets $\mathcal{A}$ were seeded with two initial solutions. The first was the feasible solution produced by the algorithm from Section 4. The second was produced by identifying the closest bus stop to each address and then executing the steps described in Sections 4.1 and 4.2. These two solutions can be considered as representing the extremes of our Pareto front approximation—the first comprises short routes, few bus stops, and potentially long walks; the second, features minimal-length walks but potentially many bus stops and long routes. Note also that the second solution obeys Constraints (3) to (5), but not necessarily Constraint (6). Such solutions are therefore permitted in the archive set during a run, though these are deleted from $\mathcal{A}$ on completion and are not considered in the statistics presented below.

The results of our trials are summarised in Table 3. Three values of δ are used here, measured in seconds (e.g., a value of $\delta = 1$ means that f_1 and f_2 are rounded down to the nearest second during execution). Three features of the algorithm's output are considered: execution time, the number of solutions in the final archive set $|\mathcal{A}|$, and the archive's S-metric. The S-metric gives the hypervolume between the archive set and a single reference point [34]. An illustration is provided in Figure 9(b). To be valid, the reference point must be dominated by all solutions in the archive. Larger hypervolumes indicate an archive that better approximates the (optimal) Pareto set.

Table 3 clearly shows that run times fall as δ is increased, but that this also brings smaller, less accurate Pareto front approximations.[5] It also shows the large range of run times required across the different problem instances. Using $\delta = 1$, for example, runs with the Porthcawl instance were seen to complete in less than 5 seconds, while runs with the Brisbane instance took nearly nine hours. Unfortunately, these run times seem difficult to predict, with contributing factors including the number of bus stops and addresses, the sizes of the various solutions considered, the number of routes, the number of iterations required in each application of local search, and the number of solutions entering the archive during the entire run.

Figure 10 compares the archive sets returned from runs on a large and small problem instance using the three different δ-values. An interesting feature in the large Adelaide instance is the gap occurring near the centre of the $\delta = 10$ archive. These gaps often occur when we have a bus stop that is relatively far from other bus stops but close to a large number of addresses. Adding or removing this bus stop then causes average walking distances and average route lengths to change quite considerably in a solution. Note also that the Porthcawl instance only uses one route, allowing solutions with an average route length of close to 45 minutes to be achieved. Archives for all ten instances, together with interactive visualisations of the actual solutions produced by this algorithm can be found online at [1].

6 Conclusions and Discussion

In this paper we analysed a real-world school bus routing problem that builds on previous models proposed in the literature by including bus stop selection, multistops, and dwell times. In doing so, relationships have been drawn with

[5] For all ten instances, the S-metrics for $\delta = 1$ were seen to be significantly different to the other values of δ at the 0.001 level (according to a Wilcoxon signed rank test).

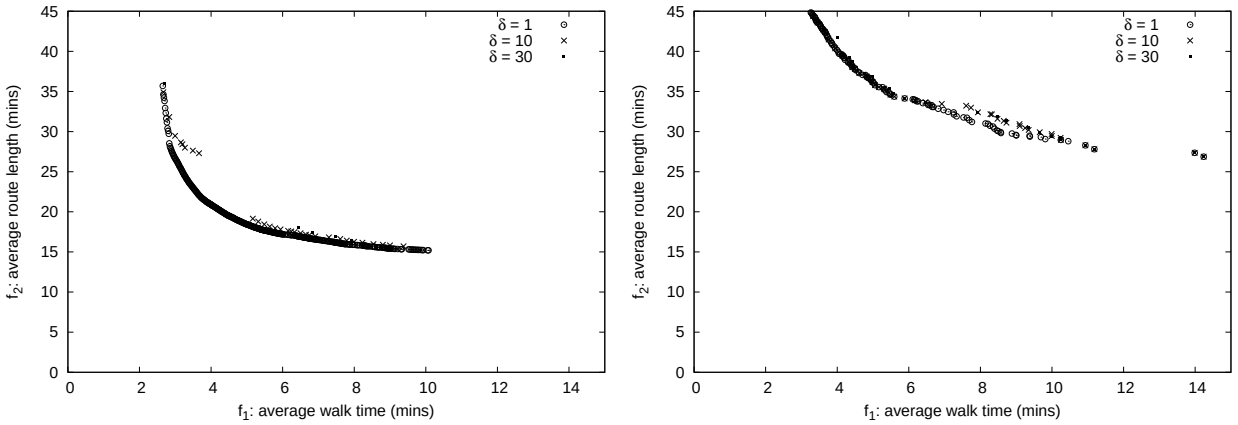

Figure 10: Archive sets returned for the Adelaide (left) and Porthcawl (right) problem instances, using $\delta = 1$, 10 and 30 seconds.

three well known combinatorial optimisation problems, which have helped inform the design of a heuristic method that can cope with large realistic-sized problem instances. The multiobjective nature of this problem has also been noted, and an appropriate algorithm has been proposed that is able to supply the user with a range of non-dominating solutions. As this is a new problem, we are unable to provide comparisons to other methods as this stage. We have, however, made our problem instances and algorithm source code publicly available to allow others to follow on from this research in the future [1].

One of the advantages of an automated bus routing system is that it can be used to explore various what-if scenarios, helping to make informed policy decisions. For example, what are the financial consequences of allowing longer maximum journey times? Are more vehicles needed if students are only expected to walk 1 km to a bus stop? How much money will be saved if eligibility distances are increased?

One issue that has not been considered in this paper is "multi-tripping". This is when a single vehicle performs multiple journeys, perhaps for different schools, one after the other. In general, we found that this issue is not considered by local government administrators. Instead, once the proposed routes have been put out for public tender, it is the bus companies' responsibility to spot opportunities for multi-trips, allowing them to give lower quotations. A simple way of encouraging multi-tripping is to design routes that start near another school. For example, if School A is known to start at 8am and School B at 8:45am, we might want to a encourage a route for School B to start near to School A, increasing the chances that both can be served by the same vehicle. Relevant work in this area can be found in [15, 24], where methods are proposed that can take solutions to multiple schools and then merge them so that buses are able to serve multiple routes.

Another real-world issue is that people often become accustomed to commuting to their bus stop at the same time every day. This is also important for parents, who may use their children's schedules to arrange an appropriate starting time at work. From their perspective, it is therefore important that solutions do not change drastically year on year. A simple way to ensure this is to use the previous year's solutions as the starting solution to the current year, which must then be repaired to cope with any changes. It would also be possible to use the difference between the old and new solution as some sort of cost function that needs to be minimised.

In real-world problems, we also need to consider the fact that some schools serve very wide geographical areas and might, therefore, only feature solutions with journey lengths that exceed the stated journey time limit m_{t}. In our problem model, this feature can be formalised using the concept of *outlier* bus stops.

Definition 4. *Let $v \in (V_1 - \{v_0\})$ be a compulsory bus stop, $V_2(v) \subseteq V_2$ be the subset of addresses for which v is the closest bus stop, and $x = \sum_{u \in V_2(v)} s(u)$. Then bus stop v is an* outlier *if $t(v, v_0) + a + bx > m_t$ (where a and b are defined in Definition 2).*

In other words, a bus stop is an outlier if it must be included in a solution, but its minimum dwell time plus its journey time to the school exceeds the stated journey time limit. One way to cope with outlier bus stops would be to simply increase m_t; another would involve considering any route containing an outlier as feasible, regardless of its actual length. Other methods may also be forthcoming, however. This, as with the other issues mentioned above, are each worthy of further research.

Acknowledgements

This research was partially supported by the Cardiff University International Collaboration Seedcorn Fund 2017.

References

[1] Source code and example solutions. `http://www.rhydlewis.eu/sbrp/`. Accessed 2018-8-30.

[2] C. Archetti, M. Savelsbergh, and M. Speranza. Worst case analysis for split delivery vehicle routing problems. *Transportation Science*, 40:226–234, 2006.

[3] C. Archetti and M. Speranza. The split delivery vehicle routing problem: A survey. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*, pages 103–122. Springer US, 2008.

[4] G. Babin, S. Deneault, and G. Laporte. Improvements to the Or-opt heuristic for the symmetric travelling salesman problem. *Journal of the Operational Research Society*, 58(3):402–407, 2007.

[5] R. Bertini and A. El-Geneidy. Modeling transit trip time using archived bus dispatch system data. *Journal of Transportation Engineering*, 130(1), 2004.

[6] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[7] G. Dantzig and J. Ramser. The truck dispatching problem. *Management Science*, 60(1):80–91, 1959.

[8] Department for Education. Home to school travel and transport guidance: Statutory guidance for local authorities. Technical report, Department for Education (England), July 2014.

[9] M. Dror and P. Trudeau. Savings by split delivery routing. *Transportation Science*, 23:141–145, 1989.

[10] M. Dror and P. Trudeau. Split delivery routing. *Naval Research Logistics*, 37:383–402, 1990.

[11] B. Eksioglu, V. Volkan, and A. Reisman. The vehicle routing problem: A taxonomic review. *Computers and Industrial Engineering*, 57(4):1472–1483, 2009.

[12] M. Garey and D. Johnson. *Computers and Intractability - A guide to NP-completeness*. W. H. Freeman and Company, San Francisco, first edition, 1979.

[13] A. Kang and S. Park. Algorithms for the variable sized bin packing problem. *European Journal of Operational Research*, 147(2):365–372, 2003.

[14] M. Karp. *Complexity of Computer Computations*, chapter Reducibility Among Combinatorial Problems, pages 85–103. Plenum, New York, 1972.

[15] B. Kim, S. Kim, and J. Park. A school bus scheduling problem. *European Journal of Operational Research*, 218:577–585, 2012.

[16] J. Kinable, F. Spieksma, and G. Vanden Berghe. School bus routing—a column generation approach. *International Transactions in Operational Research*, 21:453–478, 2014.

[17] G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43:408–416, 2009.

[18] G. Laporte, H. Mercure, and Y. Nobert. Generalized travelling salesman problem through n sets of nodes: the asymmetrical case. *Discrete Applied Mathematics*, 18(2):185–197, 1987.

[19] A. Letchford, J. Lysgaard, and R. Eglese. A branch-and-cut algorithm for the capacitated open vehicle routing problem. *Journal of the Operational Research Society*, 58:1642–1651, 2007.

[20] R. Lewis. *A Guide to Graph Colouring: Algorithms and Applications*. Springer, 2015.

[21] R. Lewis, K. Smith-Miles, and K. Phillips. The school bus routing problem: An analysis and algorithm. In *Proceedings of IWOCA 2017, the 28th International Workshop on Combinatorial Algorithms, Newcastle, Australia*, 2017.

[22] l. Paquete, M. Chiarandini, and T. Stützle. Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study. In X. Gandibleux, M. Sevaux, K. Sörensen, and V. Tkind, editors, *Metaheuristics for Multiobjective Optimisation*, volume 535 of *Lecture Notes in Economics and Mathematical Systems*, page 177200. Springer, Berlin, Germany, 2004.

[23] J. Park and B. Kim. The school bus routing problem: A review. *European Journal of Operational Research*, 202:311–319, 2010.

[24] J. Park, H. Tae, and B. Kim. A post-improvement procedure for the mixed load school bus routing problem. *European Journal of Operational Research*, 217:204–213, 2012.

[25] V. Pillac, M. Gendreau, C. Guéret, and A. Medagila. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225:1–11, 2013.

[26] H. Qin, W. Ming, Z. Zhang, Y. Xie, and A. Lim. A tabu search algorithm for the multi-period inspector scheduling problem. *Computers and Operations Research*, 59:78–93, 2015.

[27] J. Riera-Ledesma and J. Salazar-González. A column generation approach for a school bus routing problem with resource constraints. *Computers and Operations Research*, 40:566–583, 2013.

[28] P. Schittekat, J. Kinable, K. Sörensen, M. Sevaux, F. Spieksma, and J. Springael. A metaheuristic for the school bus routing problem with bus stop selection. *European Journal of Operational Research*, 229:518–528, 2013.

[29] School Bus Program, Victoria. School bus program: Policy and procedures. Technical report, Public Transport Victoria, 2016.

[30] M. Silva, A. Subramanian, and L. Satoru Ochi. An interated local search heuristic for the split delivery vehicle routing problem. *Computers and Operations Research*, 53:234–239, 2015.

[31] Transit Cooperative Research Program. Transit capacity and quality of service manual. Technical report, Transit Research Board (US), 2017. isbn: 978-0-309-28344-1.

[32] Transport for NSW. The school student transport scheme. Technical report, Transport for New South Wales, December 2016.

[33] C. Wang, Y. Zhirui, W. Yuan, X. Yueru, and W. Wei. Modeling bus dwell time and time lost serving stop in China. *Journal of Public Transportation*, 19(3):55–77, 2016.

[34] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach,. *IEEE Transactions on Evolutionary Computation*, 3(4):257271, 1999.