Computers and Operations Research 84 (2017) 62–72

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

An open source Spreadsheet Solver for Vehicle Routing Problems

Güneş Erdoğan

School of Management, University of Bath, Bath BA2 7AY, UK

A R T I C L E I N F O

Article history:
Received 17 November 2016
Revised 16 February 2017
Accepted 18 February 2017
Available online 16 March 2017

Keywords:
Vehicle Routing Problem
Metaheuristics
Spreadsheets
Open source software

A B S T R A C T

The Vehicle Routing Problem (VRP) is one of the most frequently encountered optimization problems in logistics, which aims to minimize the cost of transportation operations by a fleet of vehicles operating out of a base. This paper introduces VRP Spreadsheet Solver, an open source Excel based tool for solving many variants of the Vehicle Routing Problem (VRP). Case studies of two real-world applications of the solver from the healthcare and tourism sectors that demonstrate its use are presented. The solution algorithm for the solver, and computational results on benchmark instances from the literature are provided. The solver is found to be capable of solving Capacitated VRP and Distance-Constrained VRP instances with up to 200 customers within 1 h of CPU time.

1. Introduction

The Vehicle Routing Problem (VRP) is one of the most frequently encountered optimization problems in logistics, which aims to minimize the cost of transportation operations by a fleet of vehicles operating out of a base called *depot*. It arises in many industries and contexts at tactical and operational levels. The VRP has been introduced more than 50 years ago by Dantzig and Ramser (1959), and many variants of the VRP that incorporate additional features such as time windows (intervals in which the customers may be visited) and fleet composition (Laporte, 2009) have been studied. Despite its operational nature, the VRP is considered to be in the academic domain of Operations Research rather than Operations Management. This is due to the inherent difficulty of solving a VRP, not only due to the complexity of the associated solution algorithms but also practical considerations regarding the implementation of the solution.

From a practitioner's perspective, there exist a number of barriers to develop an in-house VRP solver. Developing a solution algorithm for VRP is a daunting task, and even if an open-source academic code is to be used as the solution algorithm, most academics develop algorithms in C++ and the resulting codes are not designed for the faint-hearted. The travel distance and duration data have to be repeatedly retrieved from a GIS, due to their dynamic nature, which introduces either a recurring cost of acquisition or the requirement for in-house specialist knowledge. It is not straightforward to manually compute the existing cost of the vehicle routes, much less so to visualize and compare the existing and optimized solutions, which is important to demonstrate the benefit of an optimization tool.

Although there exist many commercial software packages to solve VRPs, any package must be integrated with the existing software infrastructure of the company, and needs to be learned by the planning managers. Most commercial VRP software packages have a black-box component, the algorithm determining the vehicle routes, since the developers will want to protect their intellectual property. Finally, every real-world application of the VRP has specific needs to which the software should be custom-tailored, which requires a constant link with the company that developed it. In case the company ceases to exist, the software faces the risk to become obsolete in a few years.

In this study, we introduce VRP Spreadsheet Solver that overcomes the problems stated above through the familiarity of its interface, ease of use, flexibility, and accessibility. Microsoft Excel is arguably the standard software for small to medium scale quantitative analysis for businesses, and is being used in almost every corner of the world, in both academia and industry alike (Hesse and Scerno, 2009). Many software packages have built-in functionality to exchange information with Excel, which eases the integration of the solver. The code for the solver, developed using Visual Basic for Applications (VBA), is open-source and can be understood and modified by medium-level programmers. VRP Spreadsheet Solver has built-in functions to query a GIS web service, from which the distances, driving times, and maps can be retrieved. The solver is available for download on an academic website at no cost (Erdoğan, 2013), and has been downloaded over 2000 times.

E-mail address: G.Erdogan@bath.ac.uk

http://dx.doi.org/10.1016/j.cor.2017.02.022

VRP Spreadsheet Solver has been used in practice by multiple organizations in diverse sectors and countries. The organizations that provided feedback include two US companies in the oil industry, an Argentinian company in the agriculture industry, a Finnish company in the tourism sector, and two chains of chilled food delivery in Taiwan and Turkey, all of which report significant savings. We believe that VRP Spreadsheet Solver has the potential to be used throughout the world and achieve savings for many Small and Medium-sized Enterprises (SMEs), and consequently reduce CO_2 emissions. Furthermore, new VRP variants have emerged through our interaction with the users that are relevant to other sectors, and they are our contribution to the literature.

The rest of this paper is organized as follows. In Section 2, we provide a brief list of applications of the VRP. In Section 3, we present two case studies of the application of VRP Spreadsheet Solver, and a resulting new VRP variant. We provide a brief overview of how to use VRP Spreadsheet Solver in Section 4. In Section 5, we present a unified formulation that encompasses all variants of the VRP that VRP Spreadsheet Solver can handle, a metaheuristic optimization algorithm that VRP Spreadsheet Solver utilizes, and the results of VRP Spreadsheet Solver on a number of benchmark instances. Finally, in Section 6, we give our concluding remarks.

2. Applications of the VRP

With the progress of computer hardware and optimization software, better algorithms and implementations for the VRP emerge every year. The computational reach of exact algorithms for the VRP is limited to 200 customers for the most studied basic variants of the VRP, e.g. the column-and-cut generation algorithm of Baldacci et al. (2011), and decreases significantly for more realistic variants that include features such as a heterogeneous fleet or distance constraints. On the other hand, state-of-the-art metaheuristic algorithms e.g. Adaptive Large Neighborhood Search (Pisinger and Ropke, 2007), Iterated Local Search (Subramanian et al., 2010), and Unified Hybrid Genetic Search (Vidal et al., 2014) can handle much larger instances and detailed operational constraints but cannot offer a mathematical guarantee of performance.

As stated in the introduction, many commercial and free solvers exist for the VRP. A recent survey (Partyka and Hall, 2014), based on the answers to a questionnaire by 15 software vendors, have provided a number of characteristics of available VRP software packages. More recent surveys (Bräysy and Hasle, 2014; Wang et al., 2015) list a number of commercial and free VRP software packages, the latter including VRP Spreadsheet Solver, and provide features required of VRP software packages. We refer the interested reader to the comprehensive book of Toth and Vigo (2014) for critical reviews of many variants of the VRP and the associated solution algorithms.

In the rest of this section, we provide a brief list of applications of the VRP. The list is by no means complete, but is provided to give an impression of the generality of VRP and the diversity of the industries and contexts it arises in. The most straightforward applications of the VRP are found in the logistics sector. Companies in the small package shipping industry, for example, aim to minimize the routing cost while keeping the routes of the drivers as consistent as possible (Groër et al., 2009). A decision support tool utilized by Toyota for selecting third party logistics service providers based on optimized vehicle routes is presented in Schittekat and Sörensen (2009). The problem of a large Benelux logistics service provider that aims to minimize the total transportation cost in a multi-depot system, which consists of the speed-related, distance-related, and vehicle-related costs of transportation is analyzed in Demir et al. (2014).

Some examples of the VRPs arising in urban transportation are the efficient routing of the school buses (Bektaş and Elmastaş, 2007), and the design of tourist tours for visiting multiple points of interest in a city (Gavalas et al., 2014). The joint problem of bin allocation and vehicle routing to optimize solid waste collection is studied in Hemmelmayr et al. (2014). There is also a growing body of literature on optimizing the rebalancing operations for shared bicycle systems that aim to minimize the total cost and maximize the user satisfaction (Forma et al., 2015).

The importance of the use of VRP models in humanitarian logistics have been underlined in the survey by Van Wassenhove (2006). Optimizing post-disaster relief operations by minimizing estimated total travel time of vehicles is studied in Özdamar and Demir (2012). The problem of finding the optimal routes for teams that survey a disaster area to assess damage and relief needs is analyzed in Huang et al. (2013). Planning of fuel distribution operations in the case of a domestic disaster is studied in Nerg and Stuckenschneider (2014), where the authors use VRP Spreadsheet Solver to optimize the routes.

With the increasing emphasis on climate change and environmental concerns, a venue of research within the VRP has emerged in the past decade, named as Green Vehicle Routing Problems (G-VRP). The objective function of these problems focus on minimizing CO_2 emissions, noise pollution, and accidents, as stated in the recent survey on G-VRP (Eglese and Bektaş, 2014). The problem of minimizing the risk of running out of fuel, which is prominent in the route planning for Alternative Fuel Vehicles, is studied by Erdoğan and Miller-Hooks (2012). The problem of determining the optimal size and mix of a vehicle fleet of electric vehicles is analyzed by Hiermann et al. (2016).

Applications of the VRP are not limited to companies with a sole focus on logistics. Indeed, variants of VRP may arise in any context where a pickup or delivery service is performed. Specifically, examples from the healthcare sector include routing of nurses for home health care (Mankowska et al., 2014) and the transportation of blood donations to storage centers (Şahinyazan et al., 2015; Yi, 2003), and the delivery services for biological samples collected from patients to testing laboratories (Andrade-Pineda et al., 2013).

3. Case studies

VRP Spreadsheet Solver can solve more than 64 variants of the VRP, based on features related to selective visits to customers, simultaneous pickups and deliveries, time windows, fleet composition, distance constraint, and the final destination of the vehicles. Some of these variants are relevant in practice but have not been formally studied. VRP Spreadsheet Solver can hence provide a starting point and a benchmark result for future studies on such problems. In the rest of the section we go over two case studies in which VRP Spreadsheet Solver was used, and a new VRP variant that we introduce as a result.

3.1. Healthcare sector

A non-profit organization based in Istanbul, Turkey provides a set of healthcare services at home, including visits by medical doctors and nurses providing physiotherapy and psychotherapy, as well as logistic services such as patient transport, domestic cleaning, and personal hygiene. The services are performed at no cost and are provided for the poor, elderly, and disabled citizens. The number of registered patients is over 3000, and an average of 1000 patients in separate locations are served each day. The organization operates out of three bases and owns a fleet of 90 vehicles, each of which can by driven by the health specialists and contains a small medical inventory.

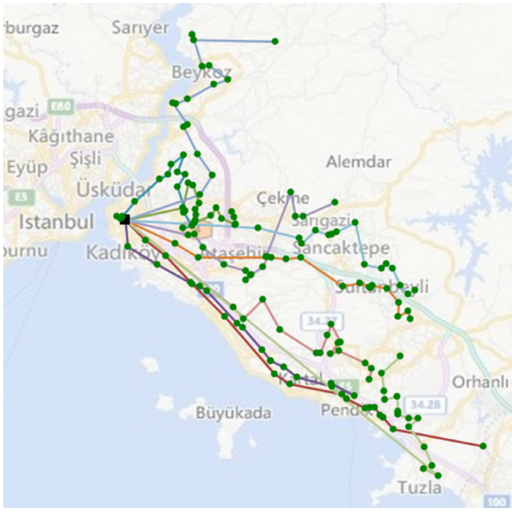

Fig. 1. Visualization of the result for the case study in the healthcare sector.

The logistics planning manager of the organization intended to use the nurse service as a test case for VRP Spreadsheet Solver and apply it to the planning of other services if the results were successful. This service operates out of one of the bases on the Anatolian side of the city with 20 vehicles and serves 150 patients per day on the average. The visits are planned in advance with no time window specified, and all patients scheduled for a visit on a given day must be visited. The service time per patient, although there are slight variations, was assumed to be constant. The vehicles are refueled during the night and their range is enough to cover a day trip, so the distance limit is not a binding constraint. Even though the vehicles are from different makers and models, they are considered identical in terms of operational parameters. Each nurse has a driving time limit of 8 h as per the general regulations about driving, and a working time limit of 9 h including the lunch break.

Capacity of the vehicles does not seem to be a binding constraint, since there is no patient transport service involved and the medical supplies in a vehicle can last a day. However, concerns of equal work allocation to the nurses were brought up that were not addressed by the working time limit. Given the size of the city and heavy traffic, a solution may contain routes with long driving times and a small number of patient visits, and other routes with short driving times and a significantly larger number of visits. To prohibit unequal work distribution in terms of patients visited, each vehicle was assigned a capacity that is slightly larger than the ratio of the number of patients to be served to the number of vehicles, and each patient location was assigned a demand value of 1.

Initial runs with this setting seemed to satisfy the workload equity concerns. However, problems with the solution were realized upon a detailed analysis of the results. The driving durations retrieved from the GIS web service are based on using the main arteries of the transportation infrastructure (i.e. highways), and although the speed of transportation on the arteries are acceptable, getting into and out of the arteries is time consuming. Hence, a solution that visits multiple districts of the city accumulates a higher driving time than the GIS driving durations provide. To overcome this problem, the users at the organization modified the driving durations by adding a penalty term to the durations between locations in separate districts to prohibit the routes from changing districts multiple times. After a few experiments with this penalty parameter, the users found a setting for which the resulting routes were observed to be more realistic. The visualization of the resulting solution for a day with 150 patient locations is provided in Fig. 1. The return arcs to the depot are omitted for a clearer visualization.

The organization refrained from providing figures regarding cost savings, but the feedback indicated that the planning process was more transparent to their staff and the use of the tool increased their awareness of the finer details of the transportation component of their operation. This case study serves to show that the decision makers often have managerial concerns that are not directly addressed in the routing literature, the GIS data may not give an accurate reflection of the reality, and parameters of VRP Spreadsheet Solver can be used to generate solutions that can address managerial concerns as well as shortcomings of the available data.

3.2. Tourism sector

A tourism company based in Finland offers numerous types of travel packages, the main one being a ferry trip between the cities of Helsinki and Tallinn (Estonia). The customers use the ferry for a day trip or an overnight visit to Tallinn, or to other cities in Estonia / Baltic countries for 1–7 days. These routes are operated from 4 to 7 days during the week, depending on the season. The company aims to maximize profitability by planning the travel packages based on demand forecasts and vehicle capacities. A significant portion of their cost is due to the bus service they outsource from subcontractors to pick up customers from their houses on the morning of their trip, and return them back to their houses at the end of their trip.

The buses subcontracted by the company are based in 7 depots, one of them in Helsinki. The buses are of different models and make, and can have different carrying capacities. The subcontractors charge a fixed price per day of use, so the problem becomes minimizing the number of buses needed, a problem of packing as well as routing. The customers need to be at the ferry terminal in Helsinki 15 min before the ferry departs, and some depots are located quite far away from Helsinki, which pushes the driving time and working time limits to be binding constraints. There are no time windows for the customers, and all locations containing customers must be serviced. The buses based in Helsinki perform closed tours, in which they pick up customers and return to the ferry port. On the other hand, the buses based at the other depots perform "open" tours, i.e. finish their trip in a location that is not their depot. These buses eventually go back to their respective depots using the exact same route traversed in the reverse direction. A sample solution is depicted in Fig. 2, where locations that do not contain any customers are not visited.

The problem at hand is then an instance of the Close–Open Mixed Vehicle Routing Problem (COMVRP) introduced in Liu and Jiang (2012). We agree with the authors of this paper that the COMVRP has applications in many sectors in which the transportation service is subcontracted to companies based at multiple cities. An extension of the COMVRP is the *Close-Open Mixed Team Orienteering Problem*, in which the customers are serviced selectively based on the capacity and the distance limit of the vehicles. A formulation that is capable of solving both these new variants as well as others is presented in Section 5.

4. How to use VRP Spreadsheet Solver

In this section, we will briefly describe the structure of the worksheets and the menu of VRP Spreadsheet Solver. We will be focusing on its usability rather than the technical details, for which the interested user can refer to the user's manual.

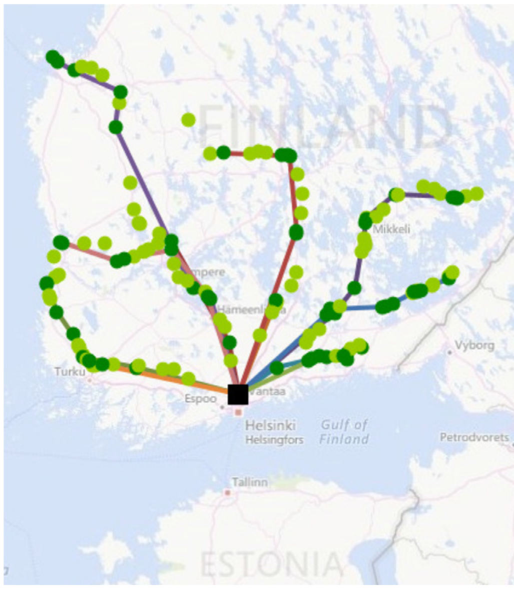

Fig. 2. Visualization of the result for the case study in the tourism sector.

4.1. Structure of the spreadsheets

VRP Spreadsheet Solver keeps the data about the elements of a VRP in separate worksheets, and adopts an incremental flow of information. Initially, the workbook only contains the worksheet named VRP Solver Console. The remaining worksheets, 1.Locations, 2.Distances, 3.Vehicles, 4.Solution, and 5.Visualization, are generated in the sequence denoted by their indices. Fig. 3 depicts the information flow between the spreadsheets, where the arrows signify the dependence of a worksheet on another worksheet.

To guide the user about which cells of a spreadsheet to work on, we have adopted the following color scheme. The cells with a black background are set by the worksheets and should not be modified. The cells with a green background are parameters or decisions to be set by the user. The cells with a yellow background are to be computed by the worksheets, but they can be edited by the user for what-if analysis. The cells with an orange background signal a warning, e.g. a vehicle arriving before the beginning of the time window of a customer. The cells with a red background signal an error, e.g. a vehicle violating the capacity constraint. The worksheets described below utilize this color scheme.

4.1.1. VRP Solver Console

This worksheet stores and provides information to the rest of worksheets. It contains various parameters regarding the size of the instance being solved and its characteristics including the number of depots and customers, number of vehicle types, the width of time windows. In addition, the user can set the options about GIS data retrieval, and the time that the user allows the solver to work on the problem. A screenshot of the worksheet is presented in Fig. 4.

4.1.2. 1.Locations

The details about the locations including their names, addresses, coordinates, time windows, and pickup and delivery service requirements are kept in this worksheet. The coordinates can be input manually, or copied and pasted from an external source, or populated using the GIS web service based on the addresses input by the user. It is good practice to provide a postcode with every address, since vague addresses may correspond to unreachable points e.g. the address of a park being resolved to be in the middle of a lake. It is possible to prohibit the vehicles from visiting certain customers using the options in this worksheet, for quick what-if analysis without data modification. Fig. 5 displays a screenshot of the worksheet.

4.1.3. 2.Distances

This worksheet contains the distances and travel durations between every two points that are specified in the 1.Locations worksheet. As of the time of this writing, using the GIS web service to populate the distances and driving distances takes about 5 min for 50 locations and 45 min for 150 locations. The number of locations for which the distance matrix can be computed is limited by the GIS web service and the type of access the user has to it. VRP Spreadsheet Solver provides an estimate the time requirement for this step by simply multiplying the number of entries in the distance matrix by a factor of 0.1 s.

The parameter about the type of route (shortest or fastest) is crucial. Choosing the shortest route usually finds routes that go through city centers, which are subject to strict speed limits and heavy traffic. Hence, using the fastest route is usually a better option for large distance delivery operations. On the other hand, fastest routes may end up using peripheral highways of the city too frequently, and consequently the shortest paths may be better suited to companies performing intra-city delivery operations. It is also possible to retrieve real-time driving durations based on the traffic, which is computed and provided by the GIS web service. The user can prohibit the vehicles from traveling between two given locations by manually setting the relevant distance to a high value. A screenshot of the 2.Distances worksheet is depicted in Fig. 6.

4.1.4. 3.Vehicles

The data about the vehicle types are kept in this worksheet. The user can set the number of vehicles of each type that are kept at each depot. The data includes cost parameters such as the cost per unit distance and the cost per trip, as well as operational parameters e.g. the depot, capacity, driving time limit, and the distance limit of the vehicle. Only one capacity parameter exists, which may correspond to the weight capacity of trucks in the case of an excavation operation, the volume capacity of tanker trucks in the case of oil transport, or the maximum number of passengers in the case of school bus routing. Fig. 7 displays a screenshot of the worksheet.

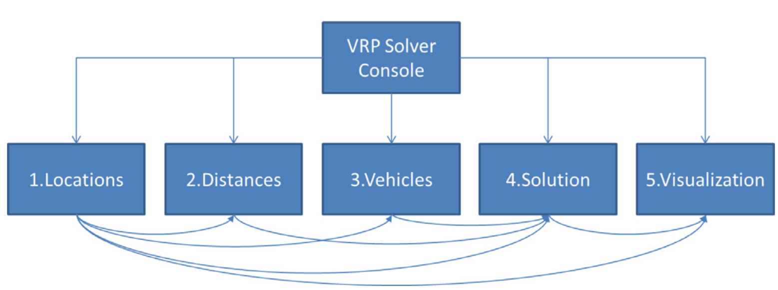

Fig. 3. Spreadsheet structure of VRP Spreadsheet Solver.

Sequence	Parameter	Value	Remarks
0.Optional - GIS License	Bing Maps Key		You can get a free key at https://www.bingmapsportal.com/
1.Locations	Number of depots	1	[1,20]
	Number of customers	10	[5,200]
2.Distances	Distance / duration computation	Bing Maps driving distances (km)	
	Bing Maps route type	Fastest	Recommendation: use Fastest
	Average vehicle speed	70	Not used for the 'Bing Maps driving distances' options
3.Vehicles	Number of vehicle types	1	Heterogeneous VRP if greater than 1
4.Solution	Vehicles must return to the depot?	Yes	Open VRP if no return
	Time window type	Hard	
	Backhauls?	No	If activated, delivery locations must be visited before pickup locations
5.Optional - Visualization	Visualization background	Bing Maps	
	Location labels	Blank	
6.Solver	Warm start?	Yes	
	Show progress on the status bar?	No	May slow down the optimization algorithm
	CPU time limit (seconds)	60	Recommendation: At least 60 seconds

Fig. 4. VRP Solver Console spreadsheet.

Location ID	Name	Address	Latitude (y)	Longitude (x)	Time window start	Time window end	Must be visited?	Service time	Pickup amount	Delivery amount	Profit
0	Depot	London, UK	51.5064201	-0.1272100	00:00	23:59	Starting location	0:00	0	0	0
1	Customer 1	Leicester, UK	52.6346016	-1.1283000	00:00	23:59	Must be visited	0:00	0	0	0
2	Customer 2	Nottingham, UK	52.9452019	-1.1407501	00:00	23:59	Must be visited	0:00	0	0	0
3	Customer 3	Bristol, UK	51.4537888	-2.5916800	00:00	23:59	Must be visited	0:00	0	0	0
4	Customer 4	Southampton, UK	50.9146004	-1.3452899	00:00	23:59	Must be visited	0:00	0	0	0
5	Customer 5	Portsmouth, UK	50.8170013	-1.0736500	00:00	23:59	Must be visited	0:00	0	0	0
6	Customer 6	Colchester, UK	51.8832016	0.9101500	00:00	23:59	Must be visited	0:00	0	0	0
7	Customer 7	Reading, UK	51.4535217	-0.9630100	00:00	23:59	Must be visited	0:00	0	0	0
8	Customer 8	Coventry, UK	52.4370003	-1.5264100	00:00	23:59	Must be visited	0:00	0	0	0
9	Customer 9	Cambridge, UK	52.2284012	0.1118500	00:00	23:59	Must be visited	0:00	0	0	0
10	Customer 10	Oxford, UK	51.7471008	-1.2568700	00:00	23:59	Must be visited	0:00	0	0	0

Fig. 5. 1.Locations spreadsheet.

From	To	Distance	Duration	Method:	Bing Maps driving distances (km)
Depot	Depot	0.00	0:00		
Depot	Customer 1	165.96	2:03		
Depot	Customer 2	204.20	2:26		
Depot	Customer 3	190.83	2:05		
Depot	Customer 4	133.86	1:32		
Depot	Customer 5	118.63	1:34		
Depot	Customer 6	108.03	1:25		
Depot	Customer 7	65.75	0:55		
Depot	Customer 8	162.24	1:56		
Depot	Customer 9	107.05	1:18		
Depot	Customer 10	95.77	1:17		
Customer 1	Depot	166.45	2:02		
Customer 1	Customer 1	0.00	0:00		
Customer 1	Customer 2	43.33	0:44		
Customer 1	Customer 3	190.82	2:15		
Customer 1	Customer 4	234.88	2:32		
Customer 1	Customer 5	257.21	2:46		
Customer 1	Customer 6	203.74	2:28		
Customer 1	Customer 7	168.02	2:08		

Fig. 6. 2.Distances spreadsheet.

Starting depot	Vehicle type	Capacity	Fixed cost per trip	Cost per unit distance	Distance limit	Work start time	Driving time limit	Working time limit	Return depot	Number of vehicles
Depot	Minibus	15	0.00	1.00	450.00	08:00	9:00	10:00	Depot	2
	Midibus	30	0.00	1.25	500.00	08:00	9:00	10:00	Depot	2
	Bus	42	0.00	1.75	560.00	08:00	9:00	10:00	Depot	1

Fig. 7. 3.Vehicles spreadsheet.

Total net profit:	-1569.90										
Vehicle:	V1 (Minibus)	Stops:	4	Net profit:	-404.69					Vehicle:	V2 (Minibus)
Stop count	Location name	Distance travelled	Driving time	Arrival time	Departure time	Working time	Profit collected	Load		Stop count	Location name
0	Depot	0.00	0:00		08:00	0:00	0	0		0	Depot
1	Customer 10	95.77	1:17	09:17	09:17	1:17	0	0		1	Customer 5
2	Customer 3	213.47	2:40	10:40	10:40	2:40	0	0		2	Customer 4
3	Customer 7	339.35	4:04	12:04	12:04	4:04	0	0		3	Depot
4	Depot	404.69	4:59	12:59		4:59	0	0		4	
5										5	
6										6	
7										7	
8										8	
9										9	
10										10	
11										11	
Detected reasons of infeasibility											

Fig. 8. 4.Solution spreadsheet.

4.1.5. 4.Solution

This worksheet is generated to contain the list of stops for each vehicle specified in 3.Vehicles, and it uses the information in 1.Locations to regarding service times and pickup / delivery amounts, as well as the distance and duration in 2.Distances to compute the departure / arrival times the cost of traveling between customers. The worksheet computes the net profit rather than cost, to accommodate variants of the VRP that accumulate profits when customers are selectively visited. This worksheet contains a number of conditional formatting features that are designed to visually identify infeasible solutions and facilitate manual solution construction. For example, a vehicle exceeding its capacity or distance limit, or a customer being visited out of its time window are highlighted in red. It is also possible to copy and paste lists of customers between vehicle routes for the purpose of manual modification of the routes. A screenshot of the 4.Solution worksheet is provided in Fig. 8.

4.1.6. 5.Visualization

The locations and the routes of the vehicles can be visually inspected by generating this optional worksheet. Options in the VRP Solver Console may be set to display various details about the locations including their pickup / delivery amounts or service times. This worksheet simply contains a scatter graph with the map of the region retrieved from the GIS web service. It can be formatted, e.g. made smaller or larger, or display axes, for the needs of the user as any other graph object of Excel. Fig. 9 displays a screenshot of the worksheet.

4.2. Structure of the menu of VRP Spreadsheet Solver

The menu is located in the “Add-ins” tab of the ribbon, and consists of 5 core and 3 optional commands to set up the worksheets. It also includes the command to engage the solver, as well the optional calls to a feasibility checker for manually modified solutions and an external solver that advanced users may develop and compile into a Dynamically Linked Library (DLL) file. The numerical indices of the commands match the numerical indices of the worksheets for ease of use (Fig. 10).

5. A unified formulation for the VRP and the solution algorithm

The field of VRP research is mature and many solution algorithms have been developed. The best known heuristic algorithm is arguably the savings algorithm (Clarke and Wright, 1964). Many metaheuristic algorithms have been proposed in the last decade, the most successful being the Adaptive Large Neighborhood Search (Pisinger and Ropke, 2007), Iterated Local Search (Subramanian et al., 2010), and Genetic Algorithms (Vidal et al., 2014). In the rest of this section we provide a unified formulation for the VRP that encompasses all variants of the VRP that VRP Spreadsheet Solver can handle, the pseudocode of the metaheuristic solution algorithm implemented with VRP Spreadsheet Solver, the details of how the infeasible solutions are handled, and the computational results of our algorithm on benchmark instances.

5.1. A unified formulation for the VRP

We first provide the notation that we will use to state the formulation. Let us define the vertex set V_D to contain the depot(s), V_C to contain the customers, and $V = V_D \cup V_C$. Furthermore, we define $V_M \subseteq V_C$ as the set of customers that must be visited. Let $G = (V, A)$ be the complete directed network on which we will solve the VRP. We define the profit of servicing a customer $i \in V_C$ as p_i, the pickup service amount for the customer as q_i, the delivery service amount as $\hat{q}_i$, and the service time required by the customer as s_i. Furthermore, we define the time interval for the customer as $[a_i, b_i]$. Note that there is also a time interval for each depot vertex.

Let us denote the set of vehicles as K, and define for each vehicle $k \in K$ the origin depot of the vehicle as $o^k \in V_D$, the work start time of the vehicle as τ^k, the fixed cost of using the vehicle as f^k the capacity of the vehicle as Q^k, the distance limit as D^k, the driving time limit as $\hat{D}^k$, the working time limit as W^k, and the return depot of the vehicle as r^k. Associated with each arc $(i, j) \in A$, there is a distance d_{ij} and driving duration $\hat{d}_{ij}$. In addition, for each vehicle $k \in K$, there is a travel cost c^k_{ij} on arc (i, j).

Next, we present the parameters related to the operational constraints. Let us define Ω to be equal to 1 if the vehicles have to return to their specified return depots and 0 otherwise. Similarly, let us define β to be 1 if there is a backhaul constraint, and 0 otherwise. In addition, we define Θ to be equal to 1 if the time windows can be violated at the cost of a penalty Π per unit time, and 0 otherwise.

We are now ready to define the decision variables. Let x^k_{ij} be equal to 1 if vehicle k traverses arc (i, j) and 0 otherwise. Furthermore, let y^k_i be equal to 1 if vehicle k visits and serves vertex i, and 0 otherwise. The amount of the pickup commodity and the delivery commodity carried by vehicle k on arc (i, j) is defined as w^k_{ij} and z^k_{ij}, respectively. We also define t^k_i as the time at which vehicle k arrives at vertex i, and v_i as the amount of violation of the time

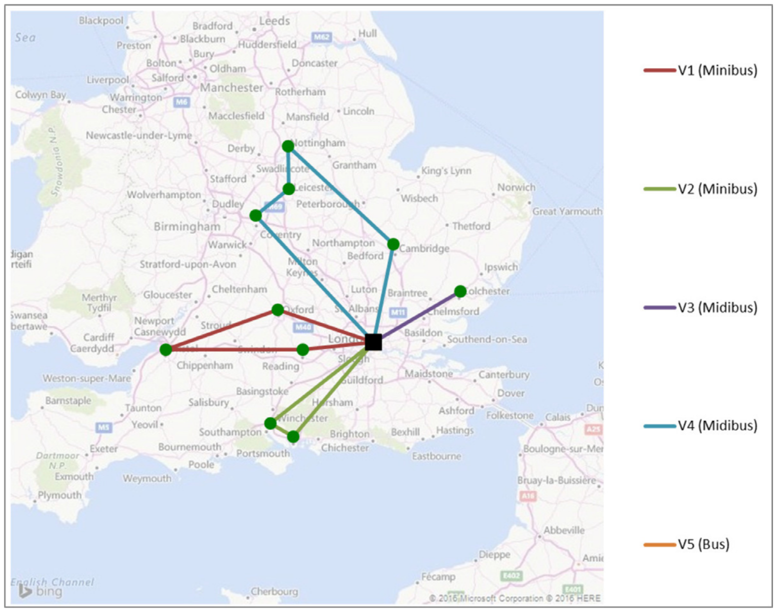

Fig. 9. 5.Visualization spreadsheet.

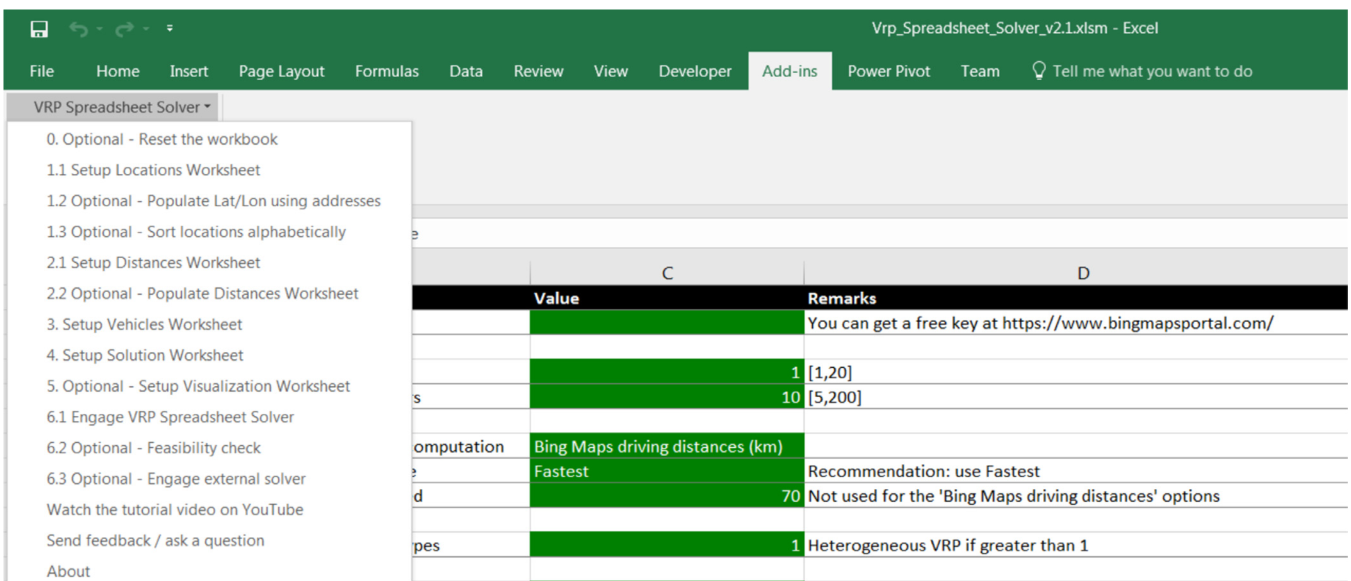

Fig. 10. VRP Spreadsheet Solver menu.

window of vertex i. The formulation for the unified VRP is then:

$$\text{Maximize} \sum_{i\in V_C}\sum_{k\in K} p_i y_i^k - \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}^k x_{ij}^k - \sum_{j\in V_C}\sum_{k\in K} f^k x_{o^k,j}^k - \Pi \sum_{i\in V} v_i \tag{1}$$

$$\text{subject to} \sum_{k\in K} y_i^k = 1 \quad \forall i \in V_M, \tag{2}$$

$$\sum_{k\in K} y_i^k \le 1 \quad \forall i \in V_C \setminus V_M, \tag{3}$$

$$\sum_{j\in V\setminus\{i\}} x_{ij}^k \le \sum_{j\in V\setminus\{i\}} x_{ji}^k \quad \forall j \in V_C, k \in K, \tag{4}$$

$$\sum_{p\in S, q\in V\setminus S} x_{pq}^k \ge y_i^k \quad \forall i \in V_C,\ k\in K,\ S\subset V : o^k \in S, i \in V\setminus S, \tag{5}$$

$$\sum_{p\in S, q\in V\setminus S} x_{pq}^k \ge \Omega y_i^k \quad \forall i \in V_C,\ k\in K,\ S\subset V : i \in S, r^k \in V\setminus S, \tag{6}$$

$$\sum_{j\in V_C} x_{o^k,j}^k \le 1 \quad \forall k \in K, \tag{7}$$

$$\sum_{k\in K} x_{ij}^k \le 1-\beta \quad \forall (i,j)\in A : q_i > 0 \text{ and } \hat{q}_j > 0 \tag{8}$$

$$\sum_{j\in V\setminus\{i\}} w_{ij}^k - \sum_{j\in V\setminus\{i\}} w_{ji}^k = q_i y_i^k \quad \forall i \in V_C, k \in K, \tag{9}$$

$$\sum_{i\in V_C} w_{i,r^k}^k = \sum_{j\in V_C} q_j y_j^k \quad \forall k \in K, \tag{10}$$

$$\sum_{j\in V\setminus\{i\}} z^k_{ji} - \sum_{j\in V\setminus\{i\}} z^k_{ij} = \hat{q}_i y^k_i \quad \forall i\in V_C, k\in K, \tag{11}$$

$$\sum_{i\in V_C} z^k_{o^k,j} = \sum_{i\in V_C} \hat{q}_i y^k_i \quad \forall k\in K, \tag{12}$$

$$t^k_i + (\hat{d}_{ij} + s_i)x^k_{ij} - W^k(1 - x^k_{ij}) \leq t^k_j \quad \forall (i,j)\in A : j\in V_C, k\in K, \tag{13}$$

$$a_i \leq t^k_i \leq b_i - s_i + v_i \quad \forall i\in V_C, k\in K, \tag{14}$$

$$v_i \leq M.\Theta \quad \forall i\in V_C, \tag{15}$$

$$t^k_{o^k} = \tau^k \quad \forall k\in K, \tag{16}$$

$$t^k_i + (s_i + \hat{d}_{ij})x^k_{i,r^k} \leq b_{r^k} + v_{r^k} + M(1-\Omega) \quad \forall (i,j)\in A : i\in V_C, k\in K, \tag{17}$$

$$w^k_{ij} + z^k_{ij} \leq Q^k x^k_{ij} \quad \forall (i,j)\in A, k\in K, \tag{18}$$

$$\sum_{(i,j)\in A} d_{ij}x^k_{ij} \leq D^k \quad \forall (i,j)\in A, k\in K, \tag{19}$$

$$\sum_{(i,j)\in A} \hat{d}_{ij}x^k_{ij} \leq \hat{D}^k \quad \forall (i,j)\in A, k\in K, \tag{20}$$

$$\sum_{i\in V_C} s_i y^k_i + \sum_{(i,j)\in A} \hat{d}_{ij}x^k_{ij} \leq W^k \quad \forall (i,j)\in A, k\in K, \tag{21}$$

$$x^k_{ij} \in \{0,1\} \quad \forall (i,j)\in A, k\in K, \tag{22}$$

$$y^k_i \in \{0,1\} \quad \forall i\in V_C, k\in K, \tag{23}$$

$$v_i \geq 0 \quad \forall i\in V_C, \tag{24}$$

$$w^k_{ij} \geq 0 \quad \forall (i,j)\in A, k\in K, \tag{25}$$

$$z^k_{ij} \geq 0 \quad \forall (i,j)\in A, k\in K. \tag{26}$$

The objective function (1) maximizes the total profit collected minus the travel cost of vehicles, fixed cost of using vehicles, and the penalty for violating time windows. We first state the constraints set the visit rules for the customers by the vehicles. Constraint (3) enforces a visit to the customers that must be visited, and constraint (2) ensures that every customer is visited at most once. Constraint set (4) is a weak form of the well-known flow conservation constraints, which require an inflow if there is an outflow, and accommodates the VRP variants in which the vehicle does not have to return to its depot. Constraints (5) provide the connectivity between the origin depot of vehicle k and the customers visited by this vehicle, and constraints (6) dictate the vehicle to return to its depot if it is required to. Constraints (7) state that each vehicle can be used at most once, whereas the backhaul constraint is enforced by constraint (8).

Next, we present the constraints that set the customer requirements. The flow conservation for the pickup commodity is provided by constraints (9) and (10). Similarly, the flow conservation for the delivery commodity is provided by constraints (11) and (12). Constraints (13) are formulated based on the Miller–Tucker–Zemlin subtour elimination constraints (Miller et al., 1960) and provide the framework for the time windows. The lower and upper limits of the time window for each customer, and the variable to account for violation are stated in constraints (14) and (15).

The final set of constraints state the restrictions related to vehicles. Constraints (16) and (17) set the start of the working time for vehicle k, and ensures that the vehicle returns to its depot on time if it is required to. Constraint (18) prohibit the violation of the vehicle capacities. Constraints (19)–(21) state the distance, driving time, and working time limits for each vehicle, respectively. Finally, constraints (22) – (26) are integrality and nonnegativity constraints.

To the best of our knowledge, there has not been any attempts to formulate a VRP with all the constraints stated above. Although the formulation can be solved to optimality only for small instances, it defines the problem precisely, demonstrates its complexity, and will serve as a reference formulation for the future studies on the VRP. Next, we provide our algorithm to solve this formulation.

5.2. Pseudocode of the solution algorithm

We have opted to implement a variant of the Adaptive Large Neighborhood Search of Pisinger and Ropke (2007) within VRP Spreadsheet Solver, due to its flexibility to accommodate many variants of the VRP. The algorithm diversifies the search by randomly removing customers from the solution at hand, and intensifies through re-insertion of the customers and local search. A high level pseudocode is provided below, named as Algorithm 1.

Algorithm 1 LNS algorithm implemented within VRP Spreadsheet Solver.

```
procedure LNS(depots, customers, distances, durations, vehicles)
Construct an incumbent solution by adding customers to the routes, choosing the customer that results in the maximal profit increase (equivalently, minimal cost increase) at every step
Improve the incumbent solution using local search with the EXCHANGE, 1-OPT, 2-OPT, and VEHICLE-EXCHANGE operators
Record the incumbent solution as the best known solution
repeat
    Destroy the incumbent solution by randomly removing vertices
    Repair the incumbent solution heuristically by adding vertices
    Improve the incumbent solution using local search with the EXCHANGE, 1-OPT, 2-OPT, and VEHICLE-EXCHANGE operators
    if the incumbent solution is better than the best known solution then
        Record the incumbent solution as the best known solution
    else
        Replace the incumbent solution by the best known solution with probability p
until time elapsed is larger than the CPU time allowed
return best known solution
end LNS
```

Four local search operators have been utilized in Algorithm 1, namely EXCHANGE, 1-OPT, 2-OPT, and VEHICLE-EXCHANGE. The EXCHANGE operator searches all possible pairs of customers in a given solution and checks if exchanging them would result in a better objective function value. The operator 1-OPT examines the possibility of removing every customer within a given solution and re-inserting it to a different position within the routes to improve the objective value. The 2-OPT operator attempts to remove two arcs from the solution at a time, e.g. the arc from customer a to customer b and the arc from customer c to customer d. To retain feasibility, it then adds the arc from customer a to customer d and the arc from customer c to customer b, and checks if the resulting solution has a better objective value. All three operators described so far have a neighborhood size of $O(|V|^2)$, and we refer the interested reader to the review by Groër et al. (2010) for their details. The operator VEHICLE-EXCHANGE attempts to exchange all the customers in the routes of two vehicles with different types, has a neighborhood size of $O(|K|^2)$, and is particularly useful for the case of heterogeneous fleets.

Two constructive heuristics are employed in step 7 are *greedy insertion* and *max regret*. The latter heuristic is based on selecting the customer, for which the difference between the cost of the cheapest insertion and the second cheapest insertion decisions is the largest. Both heuristics are chosen with equal probability at each iteration. Each heuristic finds a number of best candidates (a parameter set by the algorithm) and chooses randomly among them at each step. The probability p of rejecting an incumbent solution is set at 10% in the beginning and decreases linearly with time to reach 0% at the end of the CPU time allowance.

5.3. Handling infeasibility

Infeasibility of a solution refers to the case when one or more constraints are violated by a solution, whereas infeasibility of an instance refers to the case when it is not possible to find any feasible solution for the instance. Infeasibility is a common occurrence in the field of algorithm development. However, a software package that returns an infeasibility message with no suggestions for remedial action is of little use to a practitioner, and the computer (or real) time spent waiting for the result is perceived as wasted time.

Mathematically, a solution is either feasible or infeasible, and there is no comparison between two infeasible solutions. However, many solutions that would be declared infeasible by a computer could be useful in practice. For example, an infeasible solution with a single route exceeding the capacity of the vehicle by a fraction of the vehicle capacity can be converted into a feasible one by renegotiating the delivery amount with a customer on the route. Similarly, a vehicle exceeding the working time limit can be made feasible by paying the driver for the extra time. As a consequence, all solutions may be infeasible but in practice, some are less infeasible. This gives rise to the need of a method of penalizing infeasibility based on its severity.

An intuitive way of penalizing infeasibility is to include it into the objective function with a penalty coefficient. Let us denote the capacity of a vehicle in a homogeneous fleet by Q, the capacity required for route to be Q', and the penalty coefficient to be a large constant M. We then need to include a penalty term of $\max\{Q' - Q, 0\} \times M$ to the objective function to ensure that violating the capacity constraint will be penalized. However, minimizing this term does not necessarily make the resulting solution useful. Consider the case with k vehicles of capacity Q, and a number of customers with a total demand of $(k+1) \times Q$. In this case, the penalty value of a solution with the first vehicle containing $2Q$ units and the rest of the vehicles containing Q units is equal to the penalty value of a second solution with all vehicles carrying $Q+1$ units. However, many practitioners would find the latter solution to be more useful, since there is a smaller degree of modification required on each vehicle route.

The only hard constraint within the solution algorithm of VRP Spreadsheet Solver is to visit customers that must be visited, and this constraint is enforced on all solutions throughout the algorithm. The rest of the constraints are all treated as soft constraints, and their violations are penalized. To prioritize infeasible solutions with less severe violations, the solution algorithm uses a quadratic scaling method for the penalty. Following the example in the previous paragraph, the penalty term for a vehicle would be $(\max\{Q' - Q, 0\}/Q)^2 \times M$. As a result, a violation of the capacity constraint by 5% of the capacity would be penalized by $0.0025 \times M$ whereas a violation of 10% would be penalized by $0.01 \times M$. Similar formulas apply for the violation of the time windows, distance limit, driving time limit, and working time limit.

The solver component of VRP Spreadsheet Solver first performs a feasibility check of the data and searches for possible reasons of infeasibility. The search entails customers that must be visited but cannot be reached or serviced by any vehicle within the given time limit, as well as pickup / delivery amounts that cannot fit in any of the vehicle types. It also compares the overall carrying capacity of the fleet to the total pickup / delivery requirement of the customers. If any of these issues are found, the user is alerted with a message, and given a choice to stop or proceed. If the user decides to proceed, the resulting solution will certainly be infeasible but may still be useful.

5.4. Computational results on benchmark instances

Before presenting our computational results on benchmark instances, we would like to make a brief comparison of the computational tests carried out in the routing literature and the use of VRP optimization software in practice. The academic studies are usually run on a state-of-the art computer dedicated to the task, with the algorithms coded in C++. Additionally, the academic studies limit the CPU time by restricting the algorithm a number of iterations, which may result in different CPU times on different computers. On the other hand, a practitioner is more likely to run the VRP optimization algorithm on an ordinary laptop computer, possibly with other programs running in the background. Furthermore, a practitioner will need a solution in a given amount of time, since the output of the software will be an input of the decision process rather than its result.

We also would like to emphasize that the speed of VBA is orders of magnitude lower than that of C++. To the best of our knowledge, there is no academic source that provides a speed comparison, so we have performed the following small experiment. We have created an array of integers with 10,000 elements, and we have filled this array with $\lfloor U[0, 1]^*1000 \rfloor$, and repeated this process 10,000 times. On the average of 10 runs of this small program, C++ implementation of this small program has taken 1.56 s, whereas the VBA implementation has required 4.20 s, approximately 2.7 times that of C++. In terms of memory management and pointers, C++ is known to be very efficient, the importance of which is realized for problems with large memory requirements and results in larger performance deviations.

Based on the reasons stated above, we have decided to test the solution algorithm using a laptop computer with an Intel i7 CPU running at 2.5 GHz with 8 GB of RAM, a configuration that would reflect the computers used in practice. We have set the CPU time limit for the solution algorithm VRP Spreadsheet Solver to 15 min for instances with 50 customers, and increased it linearly with the number of customers for larger instances. We do not claim that a single algorithm can solve all the variants of the VRP to near-optimality, and we believe that a computational

Table 1
Computational results on benchmark instances.

Instance name	Number of customers	Fleet size	Vehicle capacity	Distance limit	Best known solution value	VRP Spreadsheet Solver			
						Average	Average gap	Best	Best gap
vrpnc1	50	5	160	N/A	524.61	**524.61**	**0.00%**	**524.61**	**0.00%**
vrpnc2	75	10	140	N/A	835.26	840.67	0.65%	**835.26**	**0.00%**
vrpnc3	100	8	200	N/A	826.14	841.05	1.80%	831.28	0.62%
vrpnc4	150	12	200	N/A	1028.42	1052.22	2.31%	1040.81	1.20%
vrpnc5	199	17	200	N/A	1291.29	1341.19	3.86%	1323.08	2.46%
vrpnc6	50	6	160	200	555.43	556.77	0.24%	**555.43**	**0.00%**
vrpnc7	75	11	140	160	909.68	913.13	0.38%	**909.68**	**0.00%**
vrpnc8	100	9	200	230	865.94	876.40	1.21%	**865.94**	**0.00%**
vrpnc9	150	14	200	200	1162.55	1181.77	1.65%	1170.81	0.71%
vrpnc10	199	18	200	200	1395.85	1435.27	2.82%	1415.02	1.37%
vrpnc11	120	7	200	N/A	1042.11	1047.82	0.55%	1047.61	0.53%
vrpnc12	100	10	200	N/A	819.56	821.29	0.21%	821.29	0.21%
vrpnc13	120	11	200	720	1541.14	1565.01	1.55%	1554.51	0.87%
vrpnc14	100	11	200	1040	866.37	886.41	2.31%	869.96	0.41%

experiment to solve all existing variants is beyond the scope of this paper. Hence, we have opted to use the well-known and widely used benchmark data set by Christofides et al. (1981) that contains two of the main variants of the VRP, the Capacitated VRP and the Distance Constrained VRP.

The computational results are provided in Table 1 and show that the algorithm performs very well for up to 100 customers, and returns acceptable results for larger instances. The performance of the algorithm is better for instances with a distance constraint, due to the reduced search space. The performance is slightly degraded for the last four instances, which consist of artificially constructed clusters of customers. We recommend a larger CPU time allowance for such instances.

A copy of VRP Spreadsheet Solver containing a real-world Pickup-and-Delivery VRP instance with 27 customer locations has been made available for researchers to verify the solution algorithm and use the instance as a benchmark for future studies (Erdoğan, 2017). As a final note, we would like to state that we welcome contributions from the researchers developing algorithms for the VRP, in the form of DLL files containing their solution algorithms. Contributed implementations will be hosted online jointly with VRP Spreadsheet Solver, with the full credit of each DLL file being attributed to the researchers who contributed it.

6. Concluding remarks

In this paper, we have introduced VRP Spreadsheet Solver, an open source spreadsheet solver for the VRP and its many variants. We have demonstrated its realized and potential benefits on two case studies, one from the healthcare sector and the other one from the tourism sector. We have shown the performance of the solution algorithm on a set of benchmark instances from the literature. We believe that VRP Spreadsheet Solver has the potential to be used throughout the world due to its ease of use, flexibility, and accessibility, and achieve savings for many SMEs as well as CO_2 emissions.

Any decision support tool should be able to generate alternative solutions for the decision maker. VRP Spreadsheet Solver currently returns and displays a single solution, for the sake of simplicity. As future work, we plan to add a parameter for the number of alternative solutions required by the user, record the corresponding number of best solutions encountered during the solver run, and return the results.

Acknowledgment

We thank Maria Battarra for her constructive comments on a draft of this paper. Thanks are also due to the two anonymous reviewers, whose comments have helped to improve the paper. Finally, this study has been partially supported by the Impact Development Grant of School of Management, University of Bath. This support is gratefully acknowledged.

References

Andrade-Pineda, J.L., Gonzalez-R, P.L., Framinan, J.M., 2013. A decision-making tool for a regional network of clinical laboratories. Interfaces 43 (4), 360–372.
Baldacci, R., Mingozzi, A., Roberti, R., 2011. New route relaxation and pricing strategies for the vehicle routing problem. Oper. Res. 59 (5), 1269–1283.
Bektaş, T., Elmastaş, S., 2007. Solving school bus routing problems through integer programming. J. Oper. Res. Soc. 58 (12), 1599–1604.
Bräysy, O., Hasle, G., 2014. Software tools and emerging technologies for vehicle routing and intermodal transportation. In: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications, 18. SIAM, pp. 351–380.
Christofides, N., Mingozzi, A., Toth, P., 1981. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. Math. Program. 20 (1), 255–282.
Clarke, G., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Oper. Res. 12 (4), 568–581.
Dantzig, G.B., Ramser, J.H., 1959. The truck dispatching problem. Manage. Sci. 6 (1), 80–91.
Demir, E., Woensel, T.V., de Kok, T., 2014. Multidepot distribution planning at logistics service provider nabuurs b.v. Interfaces 44 (6), 591–604.
Eglese, R., Bektaş, T., 2014. Green vehicle routing. In: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications, 18. SIAM, pp. 437–458.
Erdoğan, G., 2017. Glasgow Bike Share – Rebalancing Problem. Data in Brief Submitted.
Erdoğan, G., 2013. VRP Spreadsheet Solver. Available for download at http://verolog.deis.unibo.it/vrp-spreadsheet-solver.
Erdoğan, S., Miller-Hooks, E., 2012. A green vehicle routing problem. Transp. Res. Part E 48 (1), 100–114.
Forma, I.A., Raviv, T., Tzur, M., 2015. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. Transp. Res. Part B 71, 230–247.
Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G., 2014. A survey on algorithmic approaches for solving tourist trip design problems. J. Heuristics 20 (3), 291–328.
Groër, C., Golden, B., Wasil, E., 2009. The consistent vehicle routing problem. Manuf. Serv. Oper. Manage. 11 (4), 630–643.
Groër, C., Golden, B., Wasil, E., 2010. A library of local search heuristics for the vehicle routing problem. Math. Program. Comput. 2 (2), 79–101.
Hemmelmayr, V.C., Doerner, K.F., Hartl, R.F., Vigo, D., 2014. Models and algorithms for the integrated planning of bin allocation and vehicle routing in solid waste management. Transp. Sci. 48 (1), 103–120.
Hesse, R., Scerno, D.H., 2009. How electronic spreadsheets changed the world. Interfaces 39 (2), 159–167.
Hiermann, G., Puchinger, J., Ropke, S., Hartl, R.F., 2016. The electric fleet size and mix vehicle routing problem with time windows and recharging stations. Eur. J. Oper. Res. 252 (3), 995–1018.
Huang, M., Smilowitz, K.R., k, B.B., 2013. A continuous approximation approach for assessment routing in disaster relief. Transp. Res. Part B 50, 20–41.
Laporte, G., 2009. Fifty years of vehicle routing. Transp. Sci. 43 (4), 408–416.
Liu, R., Jiang, Z., 2012. The close-open mixed vehicle routing problem. Eur. J. Oper. Res. 220 (2), 349–360.
Mankowska, D.S., Meisel, F., Bierwirth, C., 2014. The home health care routing and scheduling problem with interdependent services. Health Care Manage. Sci. 17 (1), 15–30.
Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems. J. ACM 7 (4), 326–329.

Nerg, A., Stuckenschneider, K., 2014. Domestic Disasters and Geospatial Technology for the Defense Logistics Agency. Naval Postgraduate School, Monterey, California MBA thesis.

Özdamar, L., Demir, O., 2012. A hierarchical clustering and routing procedure for large scale disaster relief logistics planning. Transp. Res. Part E 48 (3), 591–602.

Partyka, J., Hall, R., 2014. Vehicle routing software survey: VR delivers the goods. OR/MS Today 41, 40–46.

Pisinger, D., Ropke, S., 2007. A general heuristic for vehicle routing problems. Comput. Oper. Res. 34 (8), 2403–2435.

Şahinyazan, F.G., Kara, B.Y., Taner, M.R., 2015. Selective vehicle routing for a mobile blood donation system. Eur. J. Oper. Res. 245 (1), 22–34.

Schittekat, P., Sörensen, K., 2009. OR Practice–supporting 3PL decisions in the automotive industry by generating diverse solutions to a large-scale location-routing problem. Oper. Res. 57 (5), 1058–1067.

Subramanian, A., Drummond, L.M.A., Bentes, C., Ochi, L.S., Farias, R., 2010. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. Comput. Oper. Res. 37 (11), 1899–1911.

Toth, P., Vigo, D., 2014. Vehicle routing: problems, methods, and applications. SIAM, 18.

Van Wassenhove, N.L., 2006. Humanitarian aid logistics: supply chain management in high gear. J. Oper. Res. Soc. 57 (5), 475–489.

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2014. A unified solution framework for multi-attribute vehicle routing problems. Eur. J. Oper. Res. 234 (3), 658–673.

Wang, X., Battarra, M., Golden, B., Wasil, E., 2015. Vehicle routing and scheduling. In: Teodorovic, D. (Ed.), The Routledge Handbook of Transportation. Routledge, London, U. K., pp. 238–256.

Yi, J., 2003. Vehicle routing with time windows and time-dependent rewards: a problem from the american red cross. Manuf. Serv. Oper. Manage. 5 (1), 74–77.