Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots

Alvina G.H. Kek, Ruey Long Cheu, Qiang Meng*

Department of Civil Engineering, National University of Singapore, Singapore 117576, Singapore

Received 2 May 2006; received in revised form 20 January 2007; accepted 7 February 2007

Abstract

This paper proposes two new distance-constrained capacitated vehicle routing problems (DCVRPs) to investigate for the first time, and study potential benefits in flexibly assigning start and end depots. The first problem, DCVRP_Fix is an extension of the traditional symmetric DCVRP, with additional service and travel time constraints, minimization of the number of vehicles and flexible application to both symmetric and asymmetric problems. The second problem, DCVRP_Flex is a relaxation of DCVRP_Fix to enable the flexible assignment of start and end depots. This allows vehicles the freedom to start and end their tour at different depots, while allowing for intermediate visits to any depot (for reloading) during the tour. Network models, integer programming formulations and solution algorithms for both problems are developed and presented in this paper. An analytical comparison of both problems is carried out with Singapore as a case study, considering the impact of depot locations and problem symmetry using four cases. Results show a generation of cost savings up to 49.1% by DCVRP_Flex across all the four cases. A significant portion of this stems from the flexibility to reload at any depot while the rest of it is derived from the flexibility to return to any depot. DCVRP_Flex’s adaptability and superior performance over DCVRP_Fix provides strong motivation for further research on improved exact algorithms and heuristics for this problem.

© 2007 Elsevier Ltd. All rights reserved.

Keywords: VRP; DCVRP; Flexible assignment; Integer programming

1. Introduction

The vehicle routing problem (VRP) is commonly defined as the problem of determining optimal delivery or collection routes from several depots to a set of geographically scattered customers, under a variety of side conditions [1]. Motivated by both its practical relevance and considerable difficulty, this problem has been extensively studied in the last four and a half decades after the seminal paper by Dantzig and Ramser [2]. A collection of noteworthy exact and heuristics methods developed in the last decades for VRP and some of its main variants can be found in Toth and Vigo [3]. The most studied fundamental version of VRP is the capacitated VRP (CVRP), where

* Corresponding author. Tel.: +65 6874 5494; fax: +65 6779 1635.
E-mail address: cvemq@nus.edu.sg (Q. Meng).
capacity restrictions on vehicles are imposed. One variant of the CVRP given particular attention in this paper is the DCVRP, where both vehicle capacity and maximum distance constraints are imposed.

Laporte [4] gave an excellent review of exact and approximate algorithms (i.e. heuristic methods) for the VRP, focusing keenly on CVRP and DCVRP. Extensive surveys conducted on exact methods, such as the branch-and-bound, branch-and-cut and set-covering-based algorithms for these problems found relatively little literature on DCVRP. With the exception of the branch-and-bound algorithms proposed by Laporte et al. [5,6], there has been no new exact algorithm proposed to specifically solve this problem. Laporte and Nobert [7] presented a complete and detailed analysis of the branch-and-bound algorithms proposed up until the late 1980s. It was found that the lower bounds obtained by earlier relaxations for both the symmetric and asymmetric CVRP are generally of poor quality and only allow for the optimal solution of small instances [3]. Several improved bounding techniques which were later proposed considerably increased the size of instances solvable by branch-and-bound. Fischetti et al. [8] developed additive bounding procedures for the asymmetric CVRP, while Fisher [9] and Miller [10] developed bounding procedures based on Lagrangian relaxation for the symmetric CVRP. A bound based on the set partitioning formulation was also developed by Hadjiconstantinou et al. [11]. Classical heuristics and metaheuristics for the VRP with time windows, backhauls and pickup and delivery were also reviewed.

One universal assumption/constraint made in all this previous literature on VRP is that every vehicle belongs to a specific depot and has to start and end its tour at that same depot. At the onset, this appears a practical and necessary measure to avoid major logistical tangles such as non proper zoning of areas, mix-ups in inventory, etc. However, with technological advancements in ICT (Information Communication Technology) and ITS (Intelligent Transportation Systems), the relaxation of this assumption is now a possibility with potential time and cost savings. In light of providing total solutions to fleet management systems in global express and logistics firms, GPS-based vehicle location and radio frequency identification (RFID) are slowly rising in popularity. The inception of this technology heralds the arrival of newer and more efficient operational techniques, one of which is the relaxation of the vehicle assignment constraint mentioned above. This in turn creates opportunities for further optimisation of routing solutions. This paper proposes a new and original idea of relaxing this constraint with potential time and cost savings. With a keen focus on global express and logistics operations (where capacity and range of vehicles are the key defining factors of operational efficiency), the above relaxation is applied on the DCVRP. To the best of our knowledge, this is the first time that such a relaxation is being studied in DCVRP and successful practical application of this in the industry is potentially expected to bring about cost efficiencies in a multitude of areas such as route travel distances, manpower requirements, fleet size, etc. Termed ‘flexible assignment’, this relaxation allows vehicles the freedom to start and end their tour at any depot, at the same time allowing for intermediate visits to any depot during the tour. Two new DCVRP variants, with and without flexible assignment, are proposed and analytically compared to investigate for the first time the potential benefits of this novel relaxation.

To present as accurately as possible the actual benefits of flexible assignment, there arises a need to consider practical issues such as actual travel time, service time, customer demands and number of vehicles utilized. A generalization of these factors as with previous formulations [5–7] may compromise the authenticity of results obtained. Understandably, the inclusion of these additional considerations will serve to further increase the complexity of an already strongly NP-Hard problem. The first variant without flexible assignment, DCVRP_Fix is extended from the symmetric version of the general DCVRP case [6]. It considers additional service and travel time constraints, includes a vehicle utilisation minimisation in its objective function and is applicable to both symmetric and asymmetric problems. The second variant with flexible assignment, DCVRP_Flex is a relaxation of the first variant, DCVRP_Fix to allow for flexible assignment, thus making DCVRP_Fix a special case of DCVRP_Flex. To our knowledge, this is the first time such a relaxation is being explored and successful implementation is potentially expected to bring about cost efficiencies in a multitude of areas such as route travel distances, manpower requirements, fleet size, etc.

In the following sections, the problem definitions of DCVRP_Fix and DCVRP_Flex are first defined. Network models, mixed integer programming formulations and solution algorithms for both variants are then systematically developed and presented. Mixed integer programming formulations enable the derivation of optimal solutions, and when combined with network models, are very useful in the development of efficient heuristics. The study area and simulated data set, presented next are modelled closely after real world conditions in Singapore, considering the impact of depot locations and problem symmetry using four cases. Finally, computational results are tabulated and discussed, touching briefly on the potential benefits, limitations and possible future extensions of flexible assignment in DCVRP.
2. Problem definition

Given several depots and a set of geographically scattered customers, the objective of both DCVRP\_Fix and DCVRP\_Flex is to determine the lowest cost delivery route, under capacity and time constraints, without and with flexible assignment respectively. Cost considerations here include both travel costs as well as vehicle utilisation costs. The problem definitions of the two variants are as described below.

(i) DCVRP\_Fix
   (a) Each customer is visited exactly once by exactly one vehicle.
   (b) Each vehicle route starts and ends at the same depot.
   (c) Each vehicle’s route can only pass through one depot exactly once.
   (d) A non-negative demand is associated with each customer and the sum of demands on any vehicle route may not exceed the vehicle capacity.
   (e) The length of each vehicle route consists of loading time at the depot, inter-customer travel times and service time at each customer. The total length of any vehicle route may not exceed the specified vehicle range.

(ii) DCVRP\_Flex
   These problems are defined by replacing constraints (b)–(d) from (i) with the following:
   (a) Each vehicle route starts and ends at a depot. The start and end depot need not be the same.
   (b) Each vehicle route can pass through any number of depots any number of times as long as range constraints are met.
   (c) Vehicle routes are made up of route segments (the part of the vehicle route between depot visits). A non-negative demand is associated with each customer and the sum of demands on any route segment may not exceed the vehicle capacity.

3. Network representation

Graph transformation has traditionally been a useful technique in solving Travelling Salesman Problem (TSP) extensions [12]. These transformations are such that every feasible solution to the original problem corresponds to one or more Hamiltonian circuits on a subgraph of the new graph. A Hamiltonian circuit is a special path through a graph that visits each vertex exactly once, such that the first vertex of the path corresponds to the last. Any feasible solution on the transformed graph can therefore be interpreted as a solution to the original problem on the original graph. A variation of this technique can similarly be applied to the solution of DCVRP\_Flex.

To begin with, DCVRP\_Fix is first defined on an original graph such that every feasible solution to the problem corresponds to one or more Hamiltonian circuits on a subgraph of this graph. No graph transformation is carried out in this case. Next DCVRP\_Flex makes use of a graph transformation from this original graph and is defined such that every feasible solution to the original problem corresponds to one or more Hamiltonian paths on a subgraph of this new graph. A Hamiltonian path is a path between two vertices of a graph that visits each vertex exactly once. Any feasible solution on this new graph can later be transformed back to the original graph and interpreted as a solution to the original problem.

The following is a list of symbols and their definitions as used in this paper:

- \( A \) A set of arcs in the network, \( G \).
- \( A' \) A set of arcs in the network, \( G' \).
- \( B \) Total number of sets of dummy depot nodes.
- \( C \) Travel costs matrix associated with the set of arcs, \( A \).
- \( C' \) Travel costs matrix associated with the set of arcs, \( A' \).
- \( c_{ij} \) Travel cost from node \( i \) to node \( j \) based on fuel consumption.
- \( D \) Demand vector associated with the set of nodes in \( V \); union of two disjoint sets — \( D_N \) and \( D_H \).
- \( D' \) Demand vector associated with the set of nodes in \( V' \); union of two disjoint sets — \( D_N \) and \( D_H' \).
- \( D_N \) A set of individual demands associated with the set of customer nodes, \( N \).
- \( D_H \) A set of zero values associated with the set of depot nodes, \( H \).
- \( D_H' \) A set of zero values associated with the sets of depot nodes, \( H_0 \) to \( H_B \).
- \( d_i \) Demand at node \( i \).
$G$ A network on which DCVRP_Fix is defined.
$G'$ A network on which DCVRP_Flex is defined.
$H$ A set of depot nodes.
$H_0$ A set of origin depot nodes.
$H_1$ First set of dummy depot nodes.
$H_2$ Second set of dummy depot nodes.
$H_B$ $B$th set of dummy depot nodes.
$K$ A set of available vehicles.
$k'$ Maximum number of vehicles available for use.
$M$ Large positive constant.
$m$ Fixed cost of using one vehicle for one shift (staff cost).
$N$ A set of customer nodes.
$Q$ Vehicle load carrying capacity.
$R$ Vehicle range (length of one shift).
$S$ Service time vector associated with the set of nodes, $V$; union of two disjoint sets — $S_N$ and $S_H$.
$S'$ Service time vector associated with the set of nodes $V'$; union of two disjoint sets — $S_N$ and $S_H$.
$S_N$ A set of individual service times associated with the set of customer nodes, $N$.
$S_H$ A set of constant service times associated with the set of depot nodes, $H$.
$S_H'$ A set of constant service times associated with the sets of depot nodes, $H_0$ to $H_B$.
$s_i$ Service time at node $i$.
$s_{tij}$ Travel time matrix associated with the set arcs, $A$.
$t_{i j}$ Travel time from node $i$ to node $j$.
$V$ A set of nodes in the network $G$; union of two disjoint sets — $N$ and $H$.
$V'$ A set of nodes in the network $G'$; union of $(B + 2)$ pairwise disjoint set of nodes — $N, H_0, H_1, H_2, \ldots H_B$.
$w_{ijk}$ Time variables specifying the time of arrival of vehicle $k$ at node $i$.
$x_{ij}$ Binary flow variables equal to 1 if vehicle $k$ travels from node $i$ to $j$ and 0 otherwise.
$y_k$ Binary flow variables equal to 1 if vehicle $k$ is used and 0 otherwise.
$z_{ik}$ Load variables specifying the total load (including that at node $i$) serviced by vehicle $k$ since its last visit to a depot by the time it reaches customer node $i$.

3.1. Network modelling — DCVRP_Fix

Let’s begin by first defining DCVRP_Fix on a network $G = (V, A, C, T, D, S)$, where $V = \{1, 2, \ldots, i, \ldots\}$ is a set of nodes, $A$ is a set of arcs, i.e. $A \subseteq V \times V$, $C = [c_{ij}]$ and $T = [t_{ij}]$ are matrices associated with $A$ of travel costs and travel time respectively from node $i$ to node $j$ and $D = (d_1, d_2, \ldots, d_i, \ldots)$ and $S = (s_1, s_2, \ldots, s_i, \ldots)$ are vectors associated with $V$ of demands and service times respectively at node $i$. $V$ is a union of two disjoint set of nodes $N$: a set of customer nodes, $H$: a set of depot nodes, namely, $V = N \cup H$ and $N \cap H = \emptyset$. The problem is said to be symmetrical if $c_{ij} = c_{ji}$ and $t_{ij} = t_{ji}$ for all $i, j \in V$ in the matrices $C$ and $T$. Otherwise, it is considered asymmetrical. To prevent redundant travel from depot direct to depot, the value of $c_{ij}$ and $t_{ij}$ associated with arcs between the nodes in set $H$ can be set to infinity. It is observed however that this measure is not necessary as long as matrices $C$ and $T$ satisfy a tighter form of the triangular inequality:

$$c_{ih} + c_{hj} > c_{ij} \quad \text{and} \quad t_{ih} + t_{hj} > t_{ij} \quad \forall h, i, j \in V, h \neq i \neq j. \quad (1)$$

These constraints allow the cost minimization nature of the problem to prevent any such redundancy. $D$ is a union of two disjoint sets $- D_N$: a set of individual demands associated with each customer node in $N$ and $D_H$: a set of zero values associated with the depot nodes in $H$ to denote zero demands at the depots. $S$ is also a union of two disjoint sets $- S_N$: a set of individual service times associated with each customer node in $N$ and $S_H$: a set of constant service time associated with the depot nodes in $H$ to denote loading time at the depots. $K = \{1, \ldots, k'\}$ is a set of vehicles, where $k'$ denotes the maximum number of vehicles available for use. The fixed cost of using a vehicle in one shift is assumed to be a constant value, $m$ and its capacity and range (length of one shift) are the constants, $Q$ and $R$ respectively. All
variables are assumed to be non-negative and calibrated to the same planning horizon as shown in a later section on data set development.

The objective of DCVRP_Fix is thus to determine the lowest cost set of one or more Hamiltonian circuits on subgraphs of this network, such that all customer nodes are visited, under capacity and time constraints, without flexible assignment.

Since it is required that each customer be visited exactly once by exactly one vehicle, an implicit assumption is therefore made that each individual customer’s demand is less than the capacity of the vehicle. Based on this assumption, a trivial upper bound for the maximum number of vehicles, \( k' \) in set \( K \) can be set as equal to the number of customers in set \( N \). It may be noted that setting too high a value of \( k' \) unnecessarily increases the complexity of the problem. However, this effect is mitigated by the node selection strategy to branch on the best bound, in the branch and bound solution algorithm used later.

3.2. Network transformation — DCVRP_Flex

The second variant, DCVRP_Flex is next defined on a transformed graph, \( G' = (V', A', C', T', D', S') \). The new set of nodes \( V' \) is a union of \( (B + 2) \) pairwise disjoint set of nodes — \( N' \): a set of customers nodes; \( H_0 \): a set of depot nodes; \( H_1 \): a set of dummy depot nodes; \( H_2 \): a second set of dummy depot nodes, \ldots, and \( H_B \): a \( B \)th set of dummy depot nodes, where \( B \) denotes the total number of sets of dummy depot nodes and \( (B + 1) \) denotes the maximum number of times vehicle routes can pass through depots. That is, all feasible vehicle routes must start from an original depot node in \( H_0 \) and pass through or end at subsequent depot node(s) in \( H_1 \) to \( H_B \) consecutively. \( A' \) is the new set of arcs between all nodes in \( V' \). \( C' \) and \( T' \) are the new matrices associated with \( A' \), denoting travel costs and travel time respectively. In this model, \( C' \) and \( T' \) must satisfy partially the tight triangular inequality (1), to prevent redundant travel between depots. That is, the tight triangular inequality must be satisfied for all cases except when \( i \) and \( h \), or \( j \) and \( h \) are real or dummy depot nodes representing the same depot. This gives rise to a situation where ‘redundant travel’ may occur within the same depot, between its real and/or dummy nodes. However, this is not a concern since the ‘redundant travel’ is easily eliminated at the solution interpretation stage, where all real and dummy depot nodes of the same depot are merged together. The new vector \( D' \) is a union of two disjoint sets — \( D_N \): a set of individual demands associated with each customer node in \( N \) and \( D_H' \): a set of zero values associated with the nodes in \( H_0 \) to \( H_B \) to denote zero demands at the depots. The new vector \( S' \) is also a union of two disjoint sets — \( S_N \): a set of individual service times associated with each customer node in \( N \) and \( S_H' \): a set of constant service time associated with the nodes in \( H_0 \) to \( H_B \) to denote loading time at the depots. The set \( K \) and variables \( m \), \( Q \) and \( R \) are similarly defined as above. Again, all variables are assumed to be non-negative and calibrated to the same planning horizon. Any feasible solution on \( G' \) can subsequently be transformed back into a solution on \( G \) and interpreted as a solution to the original problem by simply merging all dummy depot nodes with their respective original depot nodes.

The objective of DCVRP_Flex is thus to determine the lowest cost set of one or more Hamiltonian paths on subgraphs of this network, such that all customer nodes are visited, under capacity and time constraints, with flexible assignment.

**Determining the value of** \( B \)

Since all customers need to be serviced, one practical assumption made is the total time (travel time, loading time at depot and service time at customer) taken to service one customer by a vehicle from the nearest depot cannot exceed the vehicle’s range, as expressed in inequality (2). In the event that equality holds at two or more nodes in Eq. (3), the depot, \( i^* \) with a smaller value of \( s_i^* \) will be applied in inequality (2).

\[
s_{i^*} + t_{i^*j} + s_j + t_{ji^*} \leq R \quad \forall j \in N
\]

where

\[
t_{i^*j} + t_{ji^*} = \min_{i \in H}[t_{ij} + t_{ji}] \quad \forall j \in N.
\]  

A trivial upper bound for \( B \) can easily be set similar to \( k' \) above, as equal to the number of customers in \( N \). However, \( B \) plays a significant role in determining the complexity of the problem. A unit increase in \( B \) implies the inclusion of a new set of dummy depots nodes in the network, resulting in an exponential increase in complexity for
the problem. Thus, the value of $B$ needs to be carefully kept to a minimum. It is proposed for the value of $B$ to be problem dependent and taken as the lesser of $\left[ \frac{n(N)}{\max\{i\in N\}} \right]$ or $\left[ \frac{R}{\min\{s_{i*}+t_{i*j}+s_{j}+t_{j*j}\}} \right]$; namely:

$$B = \min \left\{ \left[ \frac{n(N)}{\max\{i\in N\}} \right], \left[ \frac{R}{\min\{s_{i*}+t_{i*j}+s_{j}+t_{j*j}\}} \right] \right\}. \tag{4}$$

It can be seen that the first term on the right-hand-side of Eq. (4) is constrained by capacity and represents the maximum number of times a vehicle needs to return to depot if it services all customers with maximum load, and the second term is constrained by range and represents the maximum number of times a vehicle needs to return to depot if it services all customers with minimum travel time on a one-per-trip basis. In the event of a large number of customers or a high maximum demand, the value of the first term would become artificially inflated. A more realistic upper bound would thus come from the second term which is limited by range of the vehicle. Conversely, a customer node located close to a depot would similarly inflate the value of the second term, resulting in the first term now giving the more realistic upper bound. These two terms are essentially upper bounds of the term $B$, constrained by capacity and range respectively. The lower of the two values would thus give the upper bound of $B$.

4. Mixed integer programming formulations

Fisher and Jaikumar [13,14] developed a three-index vehicle flow formulation for VRPs with capacity restrictions, time windows and no stopping times. Such formulations use variables to represent the passing of a vehicle on an arc or edge $(i, j)$. A three-index vehicle flow formulation has also been adopted in this paper. Although computationally intensive, the use of such an index has several advantages. It not only allows for differentiation of vehicles but also enables the tracking of individual vehicular service loads, thus serving the dual purpose of effecting capacity constraints and subtour prevention. The formulations are also capable of handling both symmetric and asymmetric problems, providing easy adaptability for sensitivity analysis to be carried out. The usefulness of such an indexing is made explicit in the mixed integer programming formulations below.

4.1. DCVRP_Fix

The mixed integer programming formulation for DCVRP_Fix involves three types of decision variables: $x_{ij,k}$: flow variables equal to 1 if vehicle $k$ travels from node $i$ to $j$ and 0 otherwise, $\forall i, j \in V, k \in K$. $y_k$: flow variables equal to 1 if vehicle $k$ is used and 0 otherwise, $\forall k \in K$. $z_{ik}$: load variables specifying the total load (including that at node $i$) serviced by vehicle $k$ by the time it reaches customer node $i$, $i \in N$ and initialised to 0 at the depot nodes $i$, $i \in H$, $\forall i \in V, k \in K$.

The mixed integer linear programming formulation for DCVRP_Fix is:

(P1) Minimize $\sum_{i \in V} \sum_{j \in V} \sum_{k \in K} x_{ijk}c_{ij} + \sum_{k \in K} y_km$ \tag{5}

subject to

$$\sum_{j \in V} \sum_{k \in K} x_{ijk} = 1 \quad \forall i \in N \tag{6}$$

$$\sum_{i \in H} \sum_{j \in N} x_{ijk} - y_k = 0 \quad \forall k \in K \tag{7}$$

$$\sum_{i \in V} x_{ijk} - \sum_{i \in V} x_{ijk} = 0 \quad \forall j \in V, k \in K \tag{8}$$

$$z_{ik} = 0 \quad \forall i \in H, k \in K \tag{9}$$
 initialise the load and ensure conservation at both customer nodes as well as Flex is: are linearised from the serves a dual purpose central to the robustness of the formulation. Besides assigning values to the load and respectively ensure that the capacity and range of each vehicle is not exceeded. In and are straightforward assignment appear to require the constraints: however serve a dual purpose. When a vehicle is utilized and passes through a depot, these constraints impose non-negativity conditions on the load variables, the network. these constraints force the vehicle route to pass through a depot and in doing so, prevent the formation of subtours in finally, Constraints initialise the load variables, to zero for all vehicles at each depot. It can be seen that constraints (10) and (11) are linearised from the following equation: Eq. (18) serves a dual purpose central to the robustness of the formulation. Besides assigning values to the load variables, at customer nodes, they also act to prevent the formation of subtours. Whenever a vehicle is utilized, these constraints force the vehicle route to pass through a depot and in doing so, prevent the formation of subtours in the network.

Next, Constraints (12) and (13) respectively ensure that the capacity and range of each vehicle is not exceeded. In Constraints (13), this is done by adding up the total service and travel time for each vehicle and ensuring that it falls within the length of one shift. Finally, Constraints (14) and (15) impose binary conditions on the flow variables, and while Constraints (16) impose non-negativity conditions on the load variables, .

4.2. DCVRP_Flex

The mixed integer programming formulation for DCVRP_Flex involves four types of decision variables:

\( x_{ijk} \): flow variables similarly defined as above.

\( y_k \): flow variables similarly defined as above.

\( z_{ik} \): load variables specifying the total load (including that at node \( i \)) serviced by vehicle \( k \) since its last visit to a depot by the time it reaches customer node \( i, i \in N \) and initialised to 0 at the depot nodes \( i, i \in H_0, \ldots, H_B, \forall i \in V', k \in K \).

\( w_{ijk} \): time variables specifying the time of arrival of vehicle \( k \) at node \( i, i \in N, H_1, \ldots, H_B \) and initialised to 0 at the original depot nodes (start nodes) \( i, i \in H_0, \forall i \in V', k \in K \).

The mixed integer linear programming formulation for DCVRP_Flex is:

(P2) Minimize

\[
(19)
\]
subject to

\begin{align}
\sum_{j \in V'} \sum_{k \in K} x_{ijk} &= 1 \quad \forall i \in N \tag{20} \\
\sum_{i \in V'} \sum_{j \neq i} x_{ijk} - \sum_{i \in V'} x_{jik} &= 0 \quad \forall j \in N, \; k \in K \tag{21} \\
\sum_{i \in H_0} \sum_{j \in N} x_{ijk} - y_k &= 0 \quad \forall k \in K \tag{22} \\
\sum_{i \in H_{b-1}} \sum_{j \in N} x_{ijk} - \sum_{i \in H_b} \sum_{j \in N} x_{jik} &= 0 \quad \forall k \in K, \; H_b \in \{H_1, \ldots, H_B\} \tag{23} \\
\sum_{i \in N} \sum_{j \in N} x_{ijk} - \sum_{i \in N} x_{jik} &\geq 0 \quad \forall j \in \{H_1, \ldots, H_{B-1}\}, \; k \in K \tag{24} \\
x_{ijk} &= 0 \quad \forall i \in H_B, \; j \in N, \; k \in K \tag{25} \\
(z_{ik} + d_j - z_{jk}) &\leq M(1 - x_{ijk}) \quad \forall i \in V', \; j \in N, \; k \in K, \; i \neq j \tag{26} \\
(z_{ik} + d_j - z_{jk}) &\geq -M(1 - x_{ijk}) \quad \forall i \in V', \; j \in N, \; k \in K, \; i \neq j \tag{27} \\
(w_{ik} + s_i + t_j - w_{jk}) &\leq M(1 - x_{ijk}) \quad \forall i \in V', \; j \in \{N, H_1, \ldots, H_B\}, \; k \in K, \; i \neq j \tag{28} \\
(w_{ik} + s_i + t_j - w_{jk}) &\geq -M(1 - x_{ijk}) \quad \forall i \in V', \; j \in \{N, H_1, \ldots, H_B\}, \; k \in K, \; i \neq j \tag{29} \\
z_{ik} &\leq Q \quad \forall i \in N, \; k \in K \tag{30} \\
w_{ik} &\leq R \quad \forall i \in \{N, H_1, \ldots, H_B\}, \; k \in K \tag{31} \\
z_{ik} &= 0 \quad \forall i \in \{H_0, \ldots, H_B\}, \; k \in K \tag{32} \\
w_{ik} &= 0 \quad \forall i \in H_0, \; k \in K \tag{33} \\
x_{ijk} \in \{0, 1\} \quad \forall i, \; j \in V', \; k \in K, \; i \neq j \tag{34} \\
y_{ik} \in \{0, 1\} \quad \forall k \in K \tag{35} \\
z_{ik} &\geq 0 \quad \forall i \in N, \; k \in K \tag{36} \\
w_{ik} &\geq 0 \quad \forall i \in \{N, H_1, \ldots, H_B\}, \; k \in K \tag{37}
\end{align}

where \( M \) is a large positive constant with a lower bound value defined as:

\[
\min(M) = \max\{2Q, R + \max\{s_i + t_{ij}\}\}. \tag{38}
\]

\( M \) has to take on the maximum absolute value of the left-hand-side terms in Constraints (10) and (11) and (26)–(29), in order for it to function effectively as a large positive constant in these equations.

The objective function, (19) and Constraints (20) are the same as objective function (5) and Constraints (6) in the previous formulation. Unlike (8) however, Constraints (21) ensure conservation of flow only at customer nodes.

**Characterisation of flow through depots**

The flow of vehicles through depots is characterized by Constraints (22)–(25). Similar to (7), assignment Constraints (22) also serve a dual purpose. When a vehicle is utilized and passes through an original depot node (start node) in \( H_0 \), these constraints assign a non-zero value to the flow variable \( y_k \). The binary nature of \( y_k \) in turn restricts all vehicle routes to only passing through this start node exactly once. Constraints (23) ensure conservation of flow through depots by forcing vehicles leaving a depot node in \( H_{b-1} \) to return to a depot node in \( H_b \). In addition, Constraints (24) ensure conservation of flow at the intermediate depot nodes in \( H_1 \) to \( H_{B-1} \) by restricting vehicles from leaving these nodes unless it has first entered it. Finally, Constraints (25) completes the flow characterization by stemming the flow of vehicles from the final depot nodes in \( H_B \). It may be further noted that although Constraints (22)–(25) appear to require Constraints (39), these can be made implicit as long as matrices \( C' \) and \( T' \) partially satisfy the tight triangular inequality as discussed earlier in Section 3.2.

\[
x_{ijk} = 0 \quad \forall i, \; j \in \{H_0, \ldots, H_B\}, \; k \in K, \; i \neq j. \tag{39}
\]
Subtour elimination

Constraints (26) and (27) are exactly the same as Constraints (10) and (11) in the previous formulation. Nevertheless, it is important to note that they function slightly differently in the prevention of subtours for this formulation. Whenever a vehicle is utilized, these constraints force the vehicle routes to pass through a depot (any depot) and thus act only to prevent the formation of subtours in the trip segments between depots and not for the entire vehicle route. Subtour prevention for the entire route is instead taken care of by Constraints (28) and (29), which are linearised from Constraints (40)

\[ \sum_{i \in V'} \sum_{k \in K} x_{ijk} (w_{ik} + s_i + t_{ij} - w_{jik}) = 0 \quad \forall j \in \{N, H_1, \ldots, H_B\}. \]  

Besides assigning values to the time variables, \( w_{ijk} \) at the customer nodes and dummy depot nodes in \( H_1 \) to \( H_B \), the constraints also serve the dual purpose of forcing vehicle routes to pass through an original depot node in \( H_0 \). In doing so, they prevent the formation of subtours throughout the entire vehicle route.

Next, Constraints (30) which are similar to Constraints (12) in the previous formulation, and Constraints (31) respectively ensure that the capacity and range of each vehicle is not exceeded. Constraints (32) initialise the load variables, \( z_{ik} \) to zero for all vehicles at all depot nodes in \( H_0 \) to \( H_B \), while Constraints (33) initialise the time variables, \( w_{ijk} \) to zero for all vehicles at each of the original depot nodes in \( H_0 \). Constraints (34)–(36) are the same as Constraints (14)–(16) and Constraints (37) impose non-negativity conditions on the time variables, \( w_{ijk} \).

5. Solution algorithms

Network models and mixed integer programming formulations for both DCVRP variants have been developed and presented in the previous sections. While mixed integer programming formulations are key to the derivation of optimal solutions, when combined with network models, they are also very useful in the development of efficient heuristics. In this paper, to accurately reflect the potential benefits of flexible assignment, an analytical comparison of both variants is conducted by solving them to optimality.

Several solution algorithms such as [15] and [16] have been developed for similar symmetric and asymmetric TSP variants respectively. In general, these algorithms proceed by first relaxing the integrality and subtour elimination constraints, and later generating them only when they are discovered to be violated. However, these techniques are not directly applicable to (P1) and (P2) due to the additional term minimizing vehicle utilization in the objective functions. Given the nature of the problems, a removal of subtour constraints and hence capacity and range constraints would result in an unrealistic reduction of vehicle utilization to one. Instead, it is proposed to solve (P1) and (P2) to optimality by applying a mix integer programming, branch and bound algorithm with a node selection strategy to branch on the best bound, i.e. branching is always done on the pending node giving the smallest value to the objective function. Additionally, in determining the value of \( B \), a flexible, problem dependant bounded solution space is redefined and the solution time is further improved, allowing larger problems to be solved.

6. Study area and simulated data set

To present as accurate a picture as possible of the potential benefits in employing flexible assignment, a simulated study area modeled against real world conditions is proposed. Working within the current solution capability of the problem, a rectangular study area (20 km × 10 km) of approximately one-third the land area of Singapore is considered. As with local conditions, this area is serviced by two depots. Four separate cases are considered: (i) symmetric problems with centralized depot location, (ii) symmetric problems with peripheral depot location, (iii) asymmetric problems with centralized depot location, and (iv) asymmetric problems with peripheral depot location.

Let the study area be defined on a 20 × 10 coordinate space. For cases (i) and (iii), the coordinates of the two depots are fixed at (5, 5) and (15, 5); while for cases (ii) and (iv), the coordinates of the two depots are fixed at (0, 5) and (20, 5). The set of customer nodes are randomly generated throughout the study space, while ensuring that matrices \( C \) and \( T \) satisfy the tight triangular inequality, and matrices \( C' \) and \( T' \) partially satisfy the tight triangular inequality (as discussed in Section 3.2), for both symmetric and asymmetric problems. Other parameter values used in this study, calculated and assumed based on current costs/conditions in Singapore are presented in Table 1.
Table 1
Parameter values calculated and assumed in this study

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Assumptions</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_{ij}$</td>
<td>Travel cost from node $i$ to $j$</td>
<td>Rate of cost: $0.10/km</td>
<td>$0.1 \times \text{dist. (km)}$</td>
<td>$\ $</td>
</tr>
<tr>
<td>$t_{ij}$</td>
<td>Travel time from node $i$ to $j$</td>
<td>Average vehicle speed:</td>
<td>$0.02 \times \text{dist. (km)}$</td>
<td>h</td>
</tr>
<tr>
<td>$d_i$</td>
<td>Demand at node $i$</td>
<td>Vehicle capacity: 10 units</td>
<td>0–10</td>
<td>units</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Demand at depot nodes $i, i \in H, H_0, \ldots, H_B$: 0 units</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Demand at customer nodes $i, i \in N$: 1–10 units</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s_i$</td>
<td>Service time at node $i$</td>
<td>Service time at depot nodes $i, i \in H, H_0, \ldots, H_B$: 0.6 h</td>
<td>0.16–0.7</td>
<td>h</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Service time at customer nodes $i, i \in N$: $[0.1 + 0.06(d_i)]$ h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m$</td>
<td>Fixed cost of utilizing one vehicle</td>
<td>$40/\text{shift}$</td>
<td>40</td>
<td>$\ $</td>
</tr>
<tr>
<td>$Q$</td>
<td>Vehicle capacity</td>
<td>–</td>
<td>10</td>
<td>units</td>
</tr>
<tr>
<td>$R$</td>
<td>Vehicle range</td>
<td>Average length of one work shift: 4 hours/shift</td>
<td>4</td>
<td>h</td>
</tr>
</tbody>
</table>

All monetary amounts stated above are in Singapore dollars.

values, (P1) and (P2) are thus applied on the rectangular study area described, using the timeframe of one work shift for each run and compared across all four cases.

7. Computational results

Problems (P1) and (P2) are coded into ILOG OPL Studio, Version 3.5.1 and solved using the ILOG CPLEX Mixed Integer Programming module. All problems are solved on a PC equipped with a Pentium 4, 3.00 GHz CPU and 1.00 GB of RAM. The key limiting factor, restricting the solution of larger problems, where the size of the search tree becomes too large is the computer’s memory limit.

P1 and P2 are applied on all four cases across a range of customers and demands, where solution capability permits. Based on preliminary investigations on the computational requirement for exact solutions, varied scenarios with 2 customers having a range of demands from 2 to 10 units and a range of 2–8 customers, with 2 units of demand each are explored and reported in this paper. For each permutation in the number of customers, six random sets of customer nodes are generated in order to calculate three different sets of Euclidean matrices for each of the four cases. Results from the runs are summarized and presented in Table 2. The definitions of the various column headings are as follows:

- $N$: Total number of customers.
- $D$: Demand per customer (assumed uniform for all customers).
- INST: Identification number for the three random instances generated in each case.
- COST: Total cost incurred per work shift.
- VEH: Total number of vehicles used in optimal solution.
- TIME: Number of CPU seconds.
- RT_DP: Indicator for return depot where ‘S’ means returning to the same depot and ‘D’ means otherwise. (‘S’ for all instances in DCVRP_Fix)
- RL: Total number of times vehicle is reloaded. (0 for all instances in DCVRP_Fix)

Results from Table 2 show the successful generation of cost savings by DCVRP_Flex across all four cases. Compared to DCVRP_Fix, cost savings from DCVRP_Flex range from a minimum of 0% to a maximum of 49.1%. The key features of the results are:

(i) DCVRP_Flex is observed to be more efficient than DCVRP_Fix in its use of vehicles. As the number of customers increases or as demand increases, DCVRP_Fix requires a greater number of vehicles much earlier than DCVRP_Flex. When this happens, DCVRP_Flex generates cost savings ranging from 47.6%–49.1%. This
Table 2
Computational results for DCVRP_Fix and DCVRP_Flex

| N  | D  | INST | Case 1 | | Case 2 | | Case 3 | | Case 4 |
|----|----|------|-------|----------|----------|-------|----------|----------|
|    |    |      | DCVRP_Fix | DCVRP_Flex | DCVRP_Fix | DCVRP_Flex | DCVRP_Fix | DCVRP_Flex |
|    |    |      | COST | VEH | TIME | COST | VEH | RT_DP | RL | TIME | COST | VEH | TIME | COST | VEH | RT_DP | RL | TIME |
| 2  | 2  | 1    | 40.84 | 1 | 0.02 | 40.84 | 1 | S | 0 | 0.03 | 41.43 | 1 | 0.01 | 41.43 | 1 | S | 0 | 0.03 |
| 2  | 17 | 0.02 | 42.00 | 1 | D | 0 | 0.05 | 42.46 | 1 | 0.01 | 42.27 | 1 | D | 0 | 0.02 |
| 3  | 80 | 0.02 | 40.80 | 1 | S | 0 | 0.03 | 41.80 | 1 | 0.02 | 41.80 | 1 | S | 0 | 0.03 |
| Ave|     | 41.27| 1 | 0.02 | 41.27| 1 | 0 | 0.04 | 41.90 | 1 | 0.01 | 41.83 | 1 | 0 | 0.03 |
| 2  | 4  | 2    | 40.84 | 1 | 0.28 | 40.84 | 1 | S | 0 | 0.03 | 41.43 | 1 | 0.02 | 41.43 | 1 | D | 0 | 0.03 |
| 2  | 17 | 0.00 | 42.00 | 1 | D | 0 | 0.03 | 42.46 | 1 | 0.01 | 42.27 | 1 | D | 0 | 0.01 |
| 3  | 80 | 0.00 | 40.80 | 1 | S | 0 | 0.03 | 41.80 | 1 | 0.00 | 41.80 | 1 | S | 0 | 0.03 |
| Ave|     | 41.27| 1 | 0.09 | 41.27| 1 | 0 | 0.03 | 41.90 | 1 | 0.01 | 41.83 | 1 | 0 | 0.02 |
| 2  | 6  | 1    | 80.88 | 2 | 0.01 | 40.88 | 1 | S | 1 | 0.03 | 82.06 | 2 | 0.01 | 42.06 | 1 | S | 1 | 0.03 |
| 2  | 00 | 0.02 | 42.17 | 1 | D | 1 | 0.05 | 83.40 | 2 | 0.02 | 43.40 | 1 | S | 1 | 0.03 |
| Ave|     | 81.43| 2 | 0.01 | 41.48| 1 | 1 | 0.04 | 82.67 | 2 | 0.02 | 42.73 | 1 | 1 | 0.03 |
| 2  | 8  | 1    | 80.88 | 2 | 0.02 | 40.88 | 1 | S | 1 | 0.03 | 82.06 | 2 | 0.02 | 42.06 | 1 | S | 1 | 0.03 |
| 2  | 00 | 0.02 | 42.17 | 1 | D | 1 | 0.05 | 83.40 | 2 | 0.02 | 43.40 | 1 | S | 1 | 0.03 |
| Ave|     | 81.43| 2 | 0.02 | 41.48| 1 | 1 | 0.03 | 82.67 | 2 | 0.02 | 42.73 | 1 | 1 | 0.03 |
| 2  | 10 | 1    | 80.88 | 2 | 0.00 | 40.88 | 1 | S | 1 | 0.02 | 82.06 | 2 | 0.00 | 42.06 | 1 | S | 1 | 0.02 |
| 2  | 00 | 0.02 | 42.17 | 1 | D | 1 | 0.03 | 83.40 | 2 | 0.02 | 43.40 | 1 | S | 1 | 0.02 |
| Ave|     | 81.43| 2 | 0.01 | 41.48| 1 | 1 | 0.02 | 82.67 | 2 | 0.01 | 42.73 | 1 | 1 | 0.02 |
| 4  | 2  | 1    | 42.35 | 1 | 0.30 | 42.35 | 1 | S | 0 | 1.30 | 42.43 | 1 | 0.25 | 42.43 | 1 | S | 0 | 0.88 |
| 2  | 91 | 0.22 | 42.91 | 1 | S | 0 | 1.17 | 42.93 | 1 | 0.19 | 42.93 | 1 | S | 0 | 0.92 |
| 3  | 30 | 0.30 | 42.83 | 1 | D | 0 | 1.36 | 43.74 | 1 | 0.31 | 43.20 | 1 | D | 0 | 1.27 |
| Ave|     | 42.79| 1 | 0.27 | 42.70 | 1 | 0 | 1.28 | 43.03 | 1 | 0.25 | 42.85 | 1 | 1 | 0.02 |
| 6  | 2  | 1    | 83.22 | 2 | 7.32 | 43.76 | 1 | D | 1 | 5.11 | 83.22 | 2 | 10.83 | 43.92 | 1 | D | 1 | 5.17 |
| 2  | 03 | 13.77 | 43.18 | 1 | D | 1 | 4.31 | 83.69 | 2 | 7.36 | 43.10 | 1 | D | 1 | 3.06 |
| 3  | 01 | 13.43 | 42.92 | 1 | D | 1 | 3.95 | 83.76 | 2 | 11.54 | 42.90 | 1 | D | 1 | 5.63 |
| Ave|     | 83.09| 2 | 11.51 | 43.29 | 1 | 1 | 4.46 | 83.56 | 2 | 9.91 | 43.31 | 1 | 1 | 4.62 |
| 8  | 2  | 1    | 84.10 | 2 | 70.46 | 43.42 | 1 | D | 1 | 331.45 | 84.93 | 2 | 657.48 | 44.11 | 1 | D | 1 | 379.53 |
| 2  | 79 | 706.98 | 43.63 | 1 | D | 1 | 164.53 | 84.70 | 2 | 998.43 | 44.35 | 1 | D | 1 | 285.79 |
| 3  | 80 | 803.55 | 43.93 | 1 | D | 1 | 437.28 | 84.32 | 2 | 941.73 | 44.61 | 1 | D | 1 | 193.03 |
| Ave|     | 83.90| 2 | 737.00 | 43.66 | 1 | 1 | 311.09 | 84.65 | 2 | 865.88 | 44.36 | 1 | 1 | 286.12 |

| N  | D  | INST | Case 3 | | Case 4 | | Case 3 | | Case 4 |
|----|----|------|-------|----------|----------|-------|----------|----------|
|    |    |      | DCVRP_Fix | DCVRP_Flex | DCVRP_Fix | DCVRP_Flex | DCVRP_Fix | DCVRP_Flex |
|    |    |      | COST | VEH | TIME | COST | VEH | RT_DP | RL | TIME | COST | VEH | TIME | COST | VEH | RT_DP | RL | TIME |
| 2  | 2  | 1    | 41.18 | 1 | 0.02 | 41.18 | 1 | S | 0 | 0.03 | 41.38 | 1 | 0.01 | 41.38 | 1 | S | 0 | 0.05 |
| 2  | 31 | 0.02 | 41.14 | 1 | D | 0 | 0.02 | 41.46 | 1 | 0.02 | 41.46 | 1 | S | 0 | 0.05 |
| 3  | 00 | 0.02 | 41.00 | 1 | S | 0 | 0.03 | 41.71 | 1 | 0.01 | 41.71 | 1 | S | 0 | 0.03 |
| Ave|     | 41.16| 1 | 0.02 | 41.11 | 1 | 0 | 0.03 | 41.52 | 1 | 0.01 | 41.52 | 1 | 0 | 0.04 |
cost efficiency gained by DCVRP_Flex can be observed to stem from its flexibility in allowing vehicles to reload at any depot. This is true for all four cases.

In cases where reloading is not necessary, i.e. the total number of vehicles used in both DCVRP_Fix and DCVRP_Flex is the same, it is observed that DCVRP_Flex continues to generate cost savings of up to 0.6%. Although relatively smaller than that generated above, these cost savings can become significant where the scale of operations is large. These cost savings occur in instances where customers are positioned such that DCVRP_Flex can capitalize on their locations and decrease vehicle route distances by returning to a different depot. Otherwise, DCVRP_Flex returns a similar total cost as DCVRP_Fix. This implies a win–win situation, where the use of DCVRP_Flex will always return at least a zero, if not positive cost savings. This is again true for all four cases. Nevertheless, it is also important to point out the added complexity in implementing DCVRP_Flex and the necessity to manage issues such as the imbalance of vehicles across depots. Operators would need to weigh these costs against the relatively benefits before implementation.

(ii) A key observation from the above results is the evident similarity in total cost variations across all four cases. The analogous success of DCVRP_Flex emphasizes the adaptability of this relaxation and its potential for success regardless of problem symmetry and depot locations.
Finally, as with the case of the traveling salesman problem [17], it is observed from the number of CPU seconds that asymmetrical problems such as those in cases (iii) and (iv) are much easier to solve. It is also noted that the number of CPU seconds increases exponentially with the problem size, preventing the practical solution of larger problems. Results generated from current problem sizes however appear to indicate that DCVRP_Flex is expected to continue generating similar percentage cost savings with increases in demand or number of customers.

8. Conclusion

Two new DCVRP variants are proposed in this paper to investigate, for the first time, potential benefits of flexible assignment. Network models and mixed integer programming formulations are developed for both problems. Using Singapore as a case study, an analytical comparison of the two variants is conducted, with numerical examples considering both depot locations and problem symmetry. Positive benefits of flexible assignment are established across all cases, with cost savings ranging from a minimum of 0% to a maximum of 49.1%, providing strong motivation for the future research of this relaxation.

The key limitations of flexible assignment lie in its high complexity and the necessity to manage vehicle imbalance across depots. Given the deterministic nature of this problem, a keen observation is made that the start and end depots are interchangeable. This extends the flexibility for vehicles to begin their route at either the start or end depot, acting to a certain extent to mitigate the problem of vehicle imbalance across depots. The high complexity of this problem can next be overcome through the development of more efficient solution algorithms or heuristics, with the help of the network models presented in this paper. It is the authors’ hope that the positive results shown in this paper will spur fellow researchers on to further explore this new and promising area of research.

References