Narendran, T.T.: CLOVES: a cluster-and-search heuristic to solve the vehicle routing problem with delivery and pick-up. Eur. J. Oper. Res. 178(3), 699-717

Article *in* European Journal of Operational Research · May 2007

Impact Factor: 2.36 · DOI: 10.1016/j.ejor.2006.01.037 · Source: DBLP

2 authors:

Keshri Ganesh
National Institute of Technology, Silchar
48 PUBLICATIONS **324** CITATIONS

SEE PROFILE

T. T. Narendran
Indian Institute of Technology Madras
79 PUBLICATIONS **1,885** CITATIONS

SEE PROFILE

ARTICLE IN PRESS

Available online at www.sciencedirect.com
SCIENCE DIRECT®

ELSEVIER

European Journal of Operational Research xxx (2006) xxx–xxx

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/ejor

Discrete Optimization

CLOVES: A cluster-and-search heuristic to solve the vehicle routing problem with delivery and pick-up

K. Ganesh, T.T. Narendran *

Department of Management Studies, Indian Institute of Technology Madras, Chennai 600036, India

Received 15 October 2004; accepted 11 January 2006

Abstract

This paper addresses the vehicle routing problem with sequence-constrained delivery and pick-up (VRPDP). We propose a multi-phase constructive heuristic that clusters nodes based on proximity, orients them along a route using *shrink-wrap algorithm* and allots vehicles using *generalized assignment procedure*. We employ genetic algorithm for an intensive final search. Trials on a large number of test-problems have yielded encouraging results.

Keywords: Distribution; Logistics; Vehicle routing problem; Clustering; Genetic algorithm

1. Introduction

The classical vehicle routing problem (VRP) involves routing (Golden and Assad, 1988) a fleet of vehicles, each visiting a set of nodes such that every node is visited exactly once and by exactly one vehicle, with the objective of minimizing the total distance traveled by all the vehicles. It has been investigated exhaustively and has been proved to be *np*-hard (Lenstra and Rinnooy Kan, 1976). The classical VRP has variants depending on tasks performed and on some constraints. We address a problem with a mix of nodes that require *pure delivery*, *pure pick-up* or both *delivery and pick-up*. Such a problem exists in blood banks in a public health care system in which the logistics involves distribution of blood from the bank to a string of hospitals, collection of blood from blood camps and, sometimes, collection as well as distribution at a few hospitals. For operational reasons, collection from camps will have to be done at the end of a trip. This imposes the constraint that nodes that involve only delivery and those that involve both delivery and pick-up must be visited ahead of nodes that require only pick-up. The problem is to be addressed under the conditions that all vehicles must return to the node of origin (depot) and that every node will be visited exactly once. We seek to develop a unified heuristic that addresses the VRPDP, solves it within reasonable limits of accuracy and computational time and is also

* Corresponding author.
E-mail addresses: kog@iitm.ac.in (K. Ganesh), ttn@iitm.ac.in (T.T. Narendran).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2006.01.037

applicable to the classical VRP and VRP with back-hauls (VRPB). When embedded in the planner's software, it can be a valuable tool towards providing service of high quality at low cost. The remainder of the paper is organized as follows: Section 2 presents a review of literature. Section 3 contains a description of the methodology. Section 4 provides the results and comparison while Section 5 presents the conclusions.

2. Literature summary

The combined delivery and pick-up problem has the following variants:

1. Vehicle routing problem with backhauls (VRPB): This has delivery and pick-up with the condition that all deliveries must be completed before commencing pick-up.
2. Vehicle routing problem with mixed delivery and pick-up (VRPMD): This has delivery alone at some nodes and pick-up alone at some nodes with no constraint on sequence or on multiple visits by a vehicle to a node.
3. Vehicle routing with simultaneous delivery and pick-up (VRPSDP): This has delivery and pick-up at every node of visit.

Table 1 presents a summary of the existing literature on these three variants of the VRP. We consider a hybrid version which combines the features of VRPB and VRPSDP. It has some nodes exclusive for *delivery* or *pick-up* while others require simultaneous *delivery and pick-up*. The VRP with this feature has not been considered in literature. The sequence is *pure delivery*, *delivery and pick-up* and *pure pick-up*. As suggested by Fisher and Jaikumar (1981) and Toth and Vigo (1999), we develop a constructive heuristic to provide good initial solutions to Genetic Algorithm and thereby accelerate its convergence.

Table 1
Literature summary

Variants	Methodology	Author and Year
VRPB	Extensions of savings algorithm (Clarke and Wright, 1964)	Deif and Bodin (1984)
	List processing heuristic, set covering heuristic using Lagrangian relaxation	Yano et al. (1987)
	Space-filling curves for clustering and routing followed by improvement procedures	Goetschalckx and Jacobs-Blecha (1989)
	Clustering, improvement procedure for solving TSP	Min et al. (1992)
	Joint clustering for pick-up and delivery nodes followed by assignment	Jacobs-Blecha and Goetschalckx (1993)
	Clustering, solving TSPs, assignment, finally constructing routes	Anily (1996)
	Insert both pickups and deliveries one by one, then use improvement procedures	Thangiah et al. (1996)
	Separate clustering, matching clusters, solving TSPs, improvement procedures	Toth and Vigo (1996)
	Exact method (Lagrangian branch-and-bound procedure)	Toth and Vigo (1997)
	Create a minimal spanning tree, transform to TSP	Gendreau et al. (1997)
	Clustering, solving TSP, improvement procedures	Toth and Vigo (1999)
	Initial solution based on saving-insertion or saving-assignment, plus reactive tabu search	Osman and Wassan (2002)
VRPMD	VRP for line-haul; backhauls inserted using stop-based criterion	Golden et al. (1985)
	VRP for line-haul; backhauls inserted using load-based criterion	Casco et al. (1988)
	Solving TSPs, then re-inserting depot	Mosheiov (1994)
	Creation of minimum spanning tree; transformation to a TSP	Anily and Mosheiov (1994)
	VRP for line-haul nodes, backhauls inserted in clusters or one by one	Salhi and Nagy (1999)
	VRP for line-haul nodes, backhauls brought in by insertion heuristic	Wade and Salhi (2002)
	Initial solution based on constructive heuristic, followed by improvement procedures	Nagy and Salhi (2005)
VRPSDP	Clustering; solving TSPs; penalization of infeasible arcs; re-solving TSPs	Min (1989)
	Solving as TSP; finding pickup and delivery order on TSP-tour	Gendreau et al. (1999)
	Insertion procedure	Dethloff (2001)
	Initial solution based on constructive heuristic; improvement procedures	Nagy and Salhi (2005)

3. Problem description: VRPDP

A fleet of V vehicles, each of capacity Q, originating from a single, common depot must serve a set of N nodes. V could either be an input specified by the user or a parameter that is evaluated at different settings so that its best value could be ascertained. Each node i is characterized by its geographical location and its delivery and pick-up requirements d_i and p_i. Each vehicle must leave from and return to the node of origin. A solution is a set of, at most, V routes such that all the nodes are visited exactly once without violating the constraint on capacity. The problem consists of determining a *route* for each vehicle so as to serve a set of nodes such that the total distance traversed is minimal. Each vehicle makes exactly one trip. The load carried by a vehicle between any pair of adjacent nodes on the route must not exceed capacity. If a node requires both delivery and pick-up, delivery precedes pick-up. In every route, *delivery* and *delivery and pick-up* nodes should be visited before the *pick-up* nodes. No route should comprise *pick-up* nodes exclusively.

4. Approach for solving the VRPDP

We propose a four-stage algorithm, ***CLOVES***

- to **CL**uster nodes, based on proximity, and determine the set of nodes that each vehicle will visit,
- to **O**rient the nodes in each cluster so that a route is identified for each vehicle,
- to assign **VE**hicles to the routes in conformance with the restrictions on sequence and on maximum loads and
- to use this constructed solution as the input to **S**earch for the best solution using GA.

4.1. Clustering

Clustering is a classification technique to group a set of objects into clusters such that the objects in the same cluster are similar in some sense and those in different clusters are dissimilar in the same sense. Proximity or distance measures are often used as the basis for clustering the objects (Anderberg, 1973).

4.1.1. Description

There are three classes of nodes, viz., delivery nodes (d), delivery and pick-up nodes (dp) and pick-up nodes (p), where $d = \{1,2,\ldots,D\}$, $dp = \{1,2,\ldots,DP\}$, $p = \{1,2,\ldots,P\}$ and $N = \{d \cup dp \cup p\}$. The steps are as follows:

(1) Classify the N nodes into D, DP and P nodes.
(2) Construct the distance matrix for each set using the Euclidean coordinates.
(3) Choose the number of initial seeds (Z) in the distance matrix from Row 0 (distances from depot to nodes). Take roughly half the number of seeds as nodes which are farthest from the depot and the rest from the nodes that are nearest to the depot.
Z_d, Z_{dp}, and Z_p are the numbers of seeds for the sets D, DP and P
K_d, K_{dp} and K_p, are the numbers of clusters for the sets D, DP and P
(Z_d, Z_{dp}, Z_pK_d, K_{dp} and $K_p \geqslant 1$)
Z_{di}, Z_{dpi}, Z_{pi} ($i = 1,2,\ldots,k$) are the seeds for the sets D, DP, P, respectively
(4) Form the initial clusters using the chosen seeds
Assign point X_w, $w = 1,2,\ldots,d$ to cluster C_{du}, $u \in \{1,2,\ldots,K_d\}$ for the set D.
iff $\|X_w - Z_u\| < \|X_w - Z_g\|$, $g = 1,2,\ldots,K_d$, and $u \# g$
Resolve ties arbitrarily.
Assign point X_y, $y = 1,2,\ldots,dp$ to cluster C_{dpv}, $v \in \{1,2,\ldots,K_{dp}\}$ for the set DP.
iff $\|X_y - Z_v\| < \|X_y - Z_g\|$, $g = 1,2,\ldots,K_{dp}$, and $v \# g$
Resolve ties arbitrarily.
Assign point X_i, $i = 1,2,\ldots,p$ to cluster C_{pj}, $j \in \{1,2,\ldots,K_p\}$ for the set P.
iff $\|X_i - Z_j\| < \|X_i - Z_g\|$, $g = 1,2,\ldots,K_p$, and $j \# g$
Resolve ties arbitrarily.

If any seed fails to gather a cluster around it, consider the next farthest or nearest node as the seed, as is appropriate, and repeat the process.

(5) If the total load of any cluster exceeds the vehicle capacity Q, re-compute the seeds $Z^*_{dw}, Z^*_{dpy}, Z^*_{pi}$ for the sets D, DP and P as follows:

$$Z^*_{dw} = \frac{1}{n_w} \sum_{X_u = C_w} X_u, \quad w = 1, 2, \ldots, K_d;$$

$$Z^*_{dpy} = \frac{1}{n_y} \sum_{X_v = C_y} X_y, \quad y = 1, 2, \ldots, K_{dp};$$

$$Z^*_{pi} = \frac{1}{n_i} \sum_{X_j = C_i} X_j, \quad i = 1, 2, \ldots, K_p;$$

n_w, n_y and n_i are the numbers of elements belonging to clusters C_w, C_y and C_i.
If $Z_{dw} = Z^*_{dw}$, $w = 1, 2, \ldots, K_d$ for the set D, then go to Step 6. Else, repeat steps 2–4.
If $Z_{dpy} = Z^*_{dpy}$, $y = 1, 2, \ldots, K_{dp}$ for the set DP, then go to Step 6. Else, repeat steps 2–4.
If $Z_{pi} = Z^*_{pi}$, $i = 1, 2, \ldots, K_p$ for the set P, then go to Step 6. Else, repeat steps 2–4.

(6) If the total load of any cluster still exceeds the vehicle capacity Q, remove nodes, as required, from overloaded clusters and reassign them to the next nearest seed such that the vehicle capacity Q is not exceeded in any cluster.

4.2. Orientation of nodes

In each feasible cluster obtained, we orient the nodes along a route using the *Shrink-Wrap* Algorithm (Lawler et al., 1985; Sofge et al., 2002). The nodes are mapped on polar coordinates, sorted by angle, then by distance (θ first, then ρ) and arranged in ascending order. This gives the route to be traversed within each cluster. Briefly, the steps are:

(1) Convert the coordinates of all the nodes to polar coordinates (θ, ρ).
(2) Sort the list of nodes in ascending order of θ. If the θ of two nodes are equal, then sort by ρ.
(3) Join the nodes in the sorted list in the same sequence to obtain the route for each cluster.

4.3. Vehicle allocation

Here, we allocate vehicles to the routes identified for each cluster. The number of vehicles available is known a priori (Petch and Salhi, 2003). Vehicles must visit nodes in the order, *delivery*, *delivery and pick-up* and *pick-up*. Routes that involve only pick-up are not allowed. For allotting vehicles to the routes identified for each cluster so as to minimize the total cost, we use the generalized assignment procedure (GAP), which addresses the capacity constraints (Fisher and Jaikumar, 1981). The application of GAP to the current problem is explained here.

4.3.1. Extended GAP

GAP is used to find the minimum cost assignment of v vehicles to n clusters such that each vehicle is assigned to exactly one cluster, subject to its available capacity. While applying GAP, each cluster is treated as a node. Since the number of clusters depends upon the number of vehicles, the problem will be relatively small and can be solved by the extended GAP in reasonable computing time. The extended GAP is similar to the heuristic of Fisher and Jaikumar (1981) for VRP.

The problem can be formulated as follows:

Notations

$v = \{1, 2, \ldots, V\}$ a set of vehicles
$n = \{1, 2, \ldots, N\}$ be a set of clusters
C_n is the cost of assigning a vehicle to cluster n; $\forall n \in N$

u_n the maximum load that will have to be carried in cluster n
t_v remaining capacity of each partially loaded vehicle v
$X_{vn} = 1$ if vehicle v assigned to cluster n
$=0$ otherwise.

The mathematical formulation of the GAP is

$$\text{Minimize} \quad \sum_{v\in V}\sum_{n\in N} C_n X_{vn} \tag{4.3.1.1}$$

$$\text{Subject to} \quad \sum_{v\in V} X_{vn} = 1 \quad \text{for } n = 1,\ldots,N, \tag{4.3.1.2}$$

$$\sum_{n\in N} u_n X_{vn} \leqslant t_v \quad \text{for } v = 1,\ldots,V, \tag{4.3.1.3}$$

$$X_{vn} \in \{0,1\} \quad \text{for } v = 1,\ldots,V \text{ and } n = 1,\ldots,N, \tag{4.3.1.4}$$

(4.3.1.2) ensures that each cluster is assigned to exactly one vehicle and (4.3.1.3) ensures that the maximum load in a cluster does not exceed the capacity of the vehicle assigned to that cluster.

4.3.2. Procedure

(1) Treat each cluster as a node.
(2) Find the coordinates of each cluster's centroid.
(3) Construct the matrix of distances between every pair of clusters.
(4) Find the cumulative delivery and pick-up load for each cluster.
(5) From the delivery clusters, choose that which is nearest to the depot.
(6) Check for load feasibility.
(7) Complete the route by including *delivery and pick-up* clusters next, followed by *pick-up* clusters.
(8) If the route is not load feasible, increase the number of vehicles and go to step (5).
(9) Check the distance from the depot to the first and last node of each delivery cluster and choose the nearer of the two as the first point of visit for a route.
(10) Find the sequence of nodes for each vehicle using nearest neighbour principle and Stop.

Fig. 1 shows an illustrative example, in which two vehicles are routed through a sequence of clusters. Vehicle 1 goes to *D* cluster 1, *DP* cluster 4 and *P* cluster 6 while vehicle 2 covers *D* cluster 2, *DP* cluster 3 and *P* cluster 5 in that order.

4.4. Intensive search using GA

The solution obtained by the three phases of the proposed constructive heuristic is used as the input to the proposed GA (Goldberg, 1989). We expect the initial population of structured solutions to evolve to high-quality solutions within a relatively small number of generations.

4.4.1. Description of GA

In the application of GA, we seek to design efficient genetic operators for crossover and for mutation. The proposed GA uses a suitable two-point crossover operator and an insertion operator for mutation. The flow chart for the proposed GA is shown in Fig. 2.

4.4.2. Description of proposed GA

(1) For VRPDP, a string (chromosome) is a solution set that shows the order in which nodes are visited, category of nodes, viz., *delivery*, *delivery and pick-up* or *pick-up* and the ordinal number of the vehicle assigned.

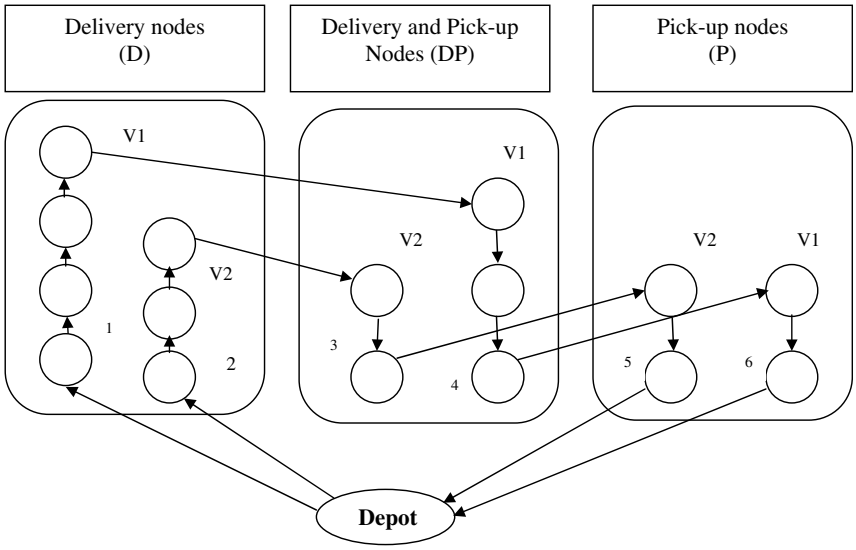

Fig. 1. Vehicle allocation by extended GAP.

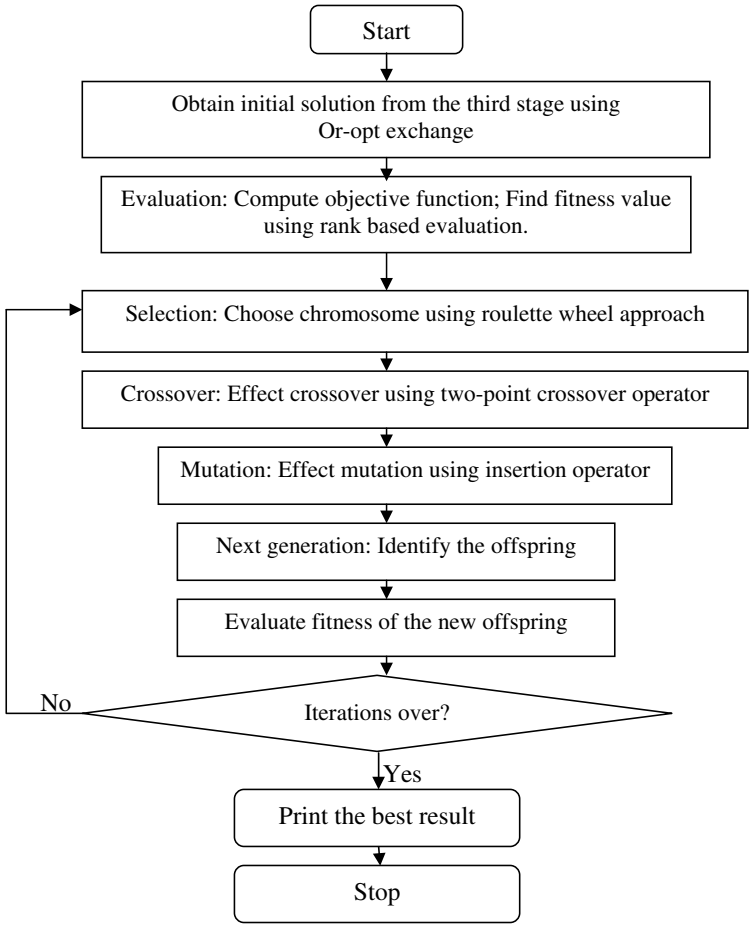

Fig. 2. Flow chart for proposed GA.

Hence we represent the solution in the form of X_{iv} where X denotes the node ($X \in N$)
i indicates D, DP or P
v indicates vehicle, ($v \in V$)
A sample string is shown:

$1_{D\ 1}3_{D\ 1}4_{D\ 1}6_{DP\ 1}5_{D\ 1}8_{DP\ 1}9_{DP\ 2}7_{P\ 2}2_{P\ 2}$

(2) Initialize: Generate the initial population (*popsize*) using Or-opt exchange (Or, 1976; Taillard et al., 1997) from the structured initial string obtained by the route construction heuristic up to the required population size by satisfying the string feasibility. Disallow strings that are exact replicas of the existing members of the population.
String feasibility
The removal of one node from its original position and insertion in another position in the same string should be load feasible (Casco et al., 1988; Dethloff, 2001).

Notations
RDC_i remaining delivery capacity prior to the insertion of node i
RPC_i remaining pick-up capacity prior to the insertion of node i
SR_i immediate successor of node i
PR_i immediate predecessor of node i
U_i remaining load on vehicle after servicing node i
N $\{1,2,\ldots,s,\ldots,i\}$

For the insertion of node s after node i, check for the following feasibility condition:
$d_s \leqslant RDC\ (PR_i)$,
$p_s \leqslant RPC\ (SR_i)$,
where

$$RDC_0 = Q - \sum_{i=1}^{N} d_i,$$

$$RDC_i = \min[RDC(PR_i), Q - U_i], \quad \text{where } i = 1,\ldots,N,$$

$$RPC(PR_0) = Q - \sum_{i=1}^{N} p_i,$$

$$RPC_i = \min[RPC(SR_i), Q - U_i], \quad \text{where } i = 1,\ldots,N.$$

(3) Evaluate fitness function: Evaluate each string by the total distance traveled through all the routes. The following procedure of rank-based evaluation is proposed to assess the merit of the string (Gen and Liu, 1995).
(a) Calculate the objective function $g(p)$ (Eq. (3.1.1)).
(b) Identify the string with the maximum value of objective function, $g_{\max}$.
(c) Transform the objective function to a score function $f(p)$ as follows:

$$f(p) = g_{\max} - g(p) \quad \text{where } g(p) \leqslant g_{\max}.$$

(d) Rank(k_j) the strings in ascending order of the score function $f(p)$.
(e) To each string s_j assign a value

$$eval(s_j) = a(1-a)^{k_j - 1},$$

where $a[a \in (0,1)]$ must be assigned by the user.
(f) Compute the cumulative probabilities q_i for each string

$$q_i = \sum_{j=1}^{i} eval(s_j).$$

Provide a higher chance to strings with good fitness value to be selected as parents for producing the next generation.

(4) Select best strings: Use Holland's proportionate selection, (Holland, 1975) known as roulette wheel selection, for creating a new population for the next generation. Every string carries a probability of being selected at any time, which is proportional to the normalized reciprocal of its fitness value.
The selection process is based on obtaining *popsize* random numbers and selecting a single string for a new population as shown below:
(a) Generate a random number r in $[0, 1]$.
(b) Select the jth string $s_j (1 \leqslant j \leqslant popsize)$ such that $(q_{j-1} \leqslant r < q_j)$.
(c) Repeat steps (a) and (b) *popsize* times and obtain that many strings.

(5) Crossover: Apply the crossover operator to each of the $N/2$ pairs of strings in the mating pool with a crossover probability P_c. Use a two point crossover; select a pair of crossing points randomly, along the length of the first parent string. Based on the string length, generate two crossover points randomly to select a segment in another parent between these crossover points. Generate the offspring by arranging the elements of the selected segment in this parent according to the order in which they appear in the other parent, with the order of the remaining elements being the same as in the first node; check it for feasibility. Exchange the role of these parents to generate another offspring. We implement two versions of the operator, labeled C1 and C2. In the first version (Fig. 3), the nodes outside the two selected points are copied into the proto child and the remaining nodes are copied from the second parent in the order of their appearance. In Fig. 3, the two crossing points are the nodes 5 and 7. In the second version of the operator (Fig. 4), we copy the sequence of the nodes between the selected crossing points into the child and take the remaining nodes from the second parent in the order of their appearance. Here is an illustration of the crossover operator:

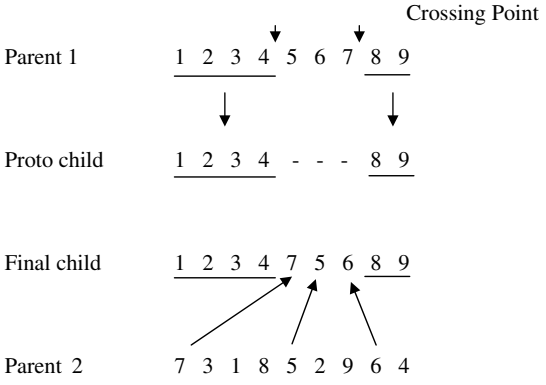

Fig. 3. Two point crossover (C1).

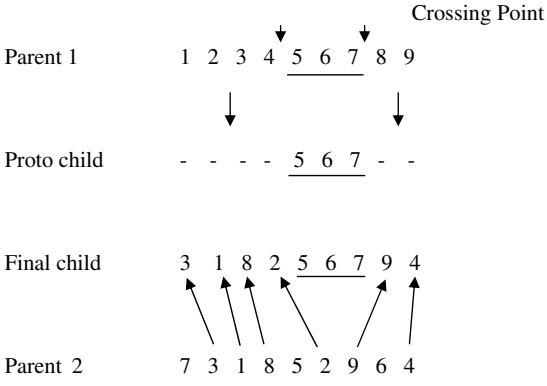

Fig. 4. Two point crossover (C2).

Select two strings randomly from the population and denote them as parent 1 and parent 2. Consider two random crossover points: iv and vii. Exchange the selected segment between two parents to produce offspring as shown in Figs. 3 and 4.

(6) Mutate: The mutation operator makes random changes to one or more nodes of the string. The neighbourhood solution is altered by an insertion-mutation operator (Gen and Cheng, 1997). Mutation is done with a small probability called Mutation Probability (P_m) in order to obtain new strings that are included if found possible.
(7) Termination: Stop, if the same solution repeats or if the maximum number of generations has been reached; else, go to Step 3.

5. Results and analysis

CLOVES algorithm is coded in C and run on a PC Pentium IV 1.70 GHz processor. There is no benchmark problem available in the literature for VRPDP. VRP does not consider *pick-up* or *delivery and pick-up* nodes; VRPB considers *delivery* nodes which precede pick-up nodes, but does not include concurrent *deliv-*

Table 2
Comparison of ***CLOVES*** with best solution value of VRP data-sets: Christofides and Eilon (1969)

Problem number	No. of nodes	Optimal solution value (OS)[a]	Constructed initial solution	% Deviation from OS	***CLOVES*** solution	% Deviation from OS	CPU time (seconds)	***CLOVES*** best solution	% Deviation from OS
E-n13-k4	12	247	260	5.26	**247**	0	0.32	**247**	0
E-n22-k4	21	375	410	9.33	**375**	0	0.41	**375**	0
E-n23-k3	22	569	602	5.80	**569**	0	0.45	**569**	0
E-n30-k3	29	534	566	5.99	**534**	0	0.63	**534**	0
E-n31-k7	30	379	409	7.92	**379**	0	0.68	**379**	0
E-n33-k4	32	835	865	3.59	**835**	0	0.96	**835**	0
E-n51-k5	50	521	630	20.92	**521**	0	1.82	**521**	0
E-n76-k7	75	682	853	25.07	694	1.76	7.71	690	1.17
E-n76-k8	75	735	956	30.07	740	0.68	6.96	738	0.41
E-n76-k10	75	830	963	16.02	867	4.46	2.42	867	4.46
E-n76-k14	75	1021	1294	26.74	1032	1.08	8.43	1032	1.08
E-n101-k8	100	815	969	18.90	830	1.84	6.52	830	1.84
E-n101-k14	100	1071	1354	26.42	1099	2.61	7.16	1099	2.61
Average relative percentage deviation				15.54		0.96			0.89

[a] http://www.branchandcut.org/VRP.

Table 3
Comparison of ***CLOVES*** with best solution value of VRP data-sets: Christofides et al. (1979)

Problem number	No. of nodes	Best-known solution value (BS)[a]	Constructed initial solution	% Deviation from BS	***CLOVES*** solution	% Deviation from BS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from BS
C1	50	524.61	531.90	1.39	**524.61**	0	0.21	**524.61**	0
C2	75	835.26	902.04	8.00	**835.26**	0	6.23	**835.26**	0
C3	100	826.14	879.43	6.45	**826.14**	0	9.63	**826.14**	0
C4	150	1028.42	1147.41	11.57	1030.46	0.20	44.84	1029.56	0.11
C5	199	1291.45	1473.4	14.09	1304.37	1.00	159.43	1298.79	0.57
C11	120	1042.11	1049.43	0.70	**1042.11**	0	6.73	**1042.11**	0
C12	100	819.56	824.42	0.59	**819.56**	0	0.89	**819.56**	0
Average relative percentage deviation				6.11		0.17			0.10

[a] Gendreau et al. (1998).

Table 4
Comparison of ***CLOVES*** with best solution value of VRP data-sets Augerat et al. (1995)

Problem number	No. of nodes	Optimal solution value (OS)[a]	Constructed initial solution	% Deviation from OS	***CLOVES*** solution	% Deviation from OS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from OS
A-n32-k5	31	784	822	4.85	784	0	3.21	784	0
A-n33-k5	32	661	684	3.48	661	0	4.21	661	0
A-n33-k6	32	742	768	3.50	742	0	3.67	742	0
A-n34-k5	33	778	814	4.63	778	0	4.23	778	0
A-n36-k5	35	799	829	3.75	799	0	4.12	799	0
A-n37-k5	36	669	710	6.13	669	0	5.21	669	0
A-n37-k6	36	949	989	4.21	949	0	5.02	949	0
A-n38-k5	37	730	785	7.53	730	0	5.23	730	0
A-n39-k5	38	822	884	7.54	822	0	5.73	822	0
A-n39-k6	38	831	897	7.94	831	0	4.23	831	0
A-n44-k6	43	937	998	6.51	937	0	5.56	937	0
A-n45-k6	44	944	1010	6.99	944	0	4.87	944	0
A-n45-k7	44	1146	1230	7.33	1146	0	6.23	1146	0
A-n46-k7	45	914	1016	11.16	914	0	6.56	914	0
A-n48-k7	47	1073	1197	11.56	1073	0	5.44	1073	0
A-n53-k7	52	1010	1295	28.22	1017	0.69	7.54	1017	0.69
A-n54-k7	53	1167	1363	16.80	1201	2.91	5.21	1172	0.43
A-n55-k9	54	1073	1254	16.87	1081	0.75	7.32	1073	0
A-n60-k9	59	1354	1740	28.51	1403	3.62	11.21	1358	0.3
A-n61-k9	60	1034	1334	29.01	1113	7.64	4.52	1038	0.39
A-n62-k8	61	1288	1656	28.57	1321	2.56	7.52	1288	0
A-n63-k9	62	1616	2149	32.98	1662	2.85	10.32	1627	0.68
A-n63-k10	63	1314	1943	47.87	1332	1.37	10.77	1322	0.61
A-n64-k9	63	1401	1786	27.48	1430	2.07	7.64	1410	0.64
A-n65-k9	64	1174	1577	34.33	1230	4.77	6.53	1177	0.26
A-n69-k9	68	1159	1491	28.65	1199	3.45	5.62	1163	0.35
A-n80-k10	79	1763	2215	25.64	1786	1.3	10.23	1780	0.96
B-n31-k5	30	672	734	9.23	672	0	4.23	672	0
B-n34-k5	33	788	878	11.42	788	0	3.43	788	0
B-n35-k5	34	955	1070	12.04	955	0	5.38	955	0
B-n38-k6	37	805	895	11.18	805	0	4.73	805	0
B-n39-k5	38	549	654	19.13	549	0	2.45	549	0
B-n41-k6	40	829	960	15.80	829	0	3.78	829	0
B-n43-k6	42	742	890	19.95	742	0	4.81	742	0
B-n44-k7	43	909	1190	30.91	909	0	5.75	909	0
B-n45-k5	44	751	990	31.82	751	0	6.24	751	0
B-n45-k6	44	678	964	42.18	678	0	6.18	678	0
B-n50-k7	49	741	1082	46.02	741	0	5.94	741	0
B-n50-k8	49	1312	1458	11.13	1322	0.76	6.55	1318	0.46
B-n51-k7	50	1032	1168	13.18	1057	2.42	3.42	1032	0
B-n52-k7	51	747	1005	34.54	747	0	4.12	747	0
B-n56-k7	55	707	941	33.10	748	5.8	12.23	710	0.42
B-n57-k7	56	1153	1701	47.53	1236	7.2	4.56	1193	3.47
B-n57-k9	56	1598	2126	33.04	1614	1	15.32	1599	0.06
B-n63-k10	62	1496	2033	35.90	1541	3.01	11.21	1510	0.94
B-n64-k9	63	861	1370	59.12	877	1.86	13.21	864	0.35
B-n67-k10	66	1032	1481	43.51	1040	0.78	15.62	1037	0.48
B-n68-k9	67	1272	1558	22.48	1317	3.54	8.21	1275	0.24
B-n78-k10	77	1221	1527	25.06	1287	5.41	11.28	1260	3.19
P-n16-k8	15	450	510	13.33	450	0	3.43	450	0
P-n19-k2	18	212	254	19.81	212	0	3.21	212	0
P-n20-k2	19	216	246	13.89	216	0	2.45	216	0
P-n21-k2	20	211	261	23.70	211	0	2.89	211	0
P-n22-k2	21	216	280	29.63	216	0	3.45	216	0
P-n22-k8	21	603	693	14.93	603	0	3.76	603	0

(*continued on next page*)

Table 4 (*continued*)

Problem number	No. of nodes	Optimal solution value (OS)[a]	Constructed initial solution	% Deviation from OS	***CLOVES*** solution	% Deviation from OS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from OS
P-n23-k8	22	529	630	19.09	529	0	4.32	529	0
P-n40-k5	39	458	570	24.45	458	0	7.32	458	0
P-n45-k5	44	510	590	15.69	510	0	6.65	510	0
P-n50-k7	49	554	684	23.47	554	0	8.21	554	0
P-n50-k8	49	631	867	37.40	650	3.01	11.21	643	1.9
P-n50-k10	49	696	880	26.44	696	0	8.34	696	0
P-n51-k10	50	741	891	20.24	741	0	7.48	741	0
P-n55-k7	54	568	687	20.95	568	0	7.86	568	0
P-n55-k10	54	694	889	28.10	703	1.3	9.21	698	0.58
P-n55-k15	54	989	1198	21.13	989	0	8.25	989	0
P-n60-k10	59	744	955	28.36	764	2.69	4.56	744	0
P-n60-k15	59	968	1245	28.62	1008	4.13	2.38	968	0
P-n65-k10	64	792	922	16.41	809	2.15	5.12	800	1.01
P-n70-k10	69	827	1057	27.81	876	5.93	7.23	827	0
P-n76-k4	75	593	734	23.78	593	0	6.43	593	0
P-n76-k5	75	627	793	26.48	627	0	7.99	627	0
P-n101-k4	100	681	788	15.71	693	1.76	8.21	687	0.88
Average relative percentage deviation				21.87		1.21			0.27

[a] http://www.branchandcut.org/VRP.

ery and pick-up nodes. Hence, for the purpose of comparison, the proposed methodology is tested on VRP data-sets, VRRDP derived from VRP data-sets and VRPB data-sets.

5.1. Performance of CLOVES

The number of nodes, the best-known solution value reported in the literature, the solution obtained by the construction heuristic, the solution and the CPU time for CLOVES and the best solution obtained by CLOVES are reported for all data-sets. A statistic called Relative percentage Deviation (RD) is calculated for each solution as follows:

$$\mathrm{RD} = \frac{(\text{Obtained solution} - \text{Optimal or Best known solution})}{\text{Optimal or Best known solution}} * 100.$$

An average of the RD's is then calculated for the best solutions and presented in last row of each table. Computing times, though reported, are not used for comparison owing to possible variations in the configurations of hardware and software used.

Table 5
Comparison of ***CLOVES*** for VRPDP data-sets derived from VRP data-sets

Problem number	No. of nodes	Lower bound (LB)[a]	Constructed initial solution	% Deviation from LB	***CLOVES*** solution	% Deviation from LB	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from LB
Z1	50	524.61	601.34	14.63	603.11	14.9635	0.23	603.11	14.9635
Z2	75	835.26	985.54	17.99	872.86	4.501592	17.23	872.86	4.501592
Z3	100	826.14	994.73	20.41	923.74	11.81398	19.63	923.74	11.81398
Z4	150	1028.42	1198.21	16.51	1178.13	14.51839	124.84	1177.45	14.49116
Z5	199	1291.45	1476.32	14.31	1484.07	14.38073	179.43	1477.01	14.36835
Z11	120	1042.11	1267.34	21.61	1072.01	2.869179	7.43	1070.23	2.698372
Z12	100	819.56	992.87	21.15	873.16	0.122017	1.91	873.08	6.530333
Average relative percentage deviation				18.09		10.02			9.91

[a] Gendreau et al. (1998).

Table 6
Comparison of ***CLOVES*** with best solution value of VRPB data-sets: Goetschalckx and Jacobs-Blecha (1989) using a real-valued cost matrix

Problem number	No. of nodes	Best-known solution value (BS)[a]	Constructed initial solution	% Deviation from BS	***CLOVES*** solution	% Deviation from BS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from BS
A1	25	229885.65	269043.35	17.03	229885.65	0	20.82	229885.65	0
A2	25	180119.21	215498.79	19.64	180119.21	0	24.52	180119.21	0
A3	25	163405.38	182510.62	11.69	163405.38	0	18.50	163405.38	0
A4	25	155796.41	197540.59	26.79	155796.41	0	17.59	155796.41	0
B1	30	239080.15	276818.85	15.78	239080.15	0	47.28	239080.15	0
B2	30	198047.77	246210.23	24.32	198047.77	0	36.29	198047.77	0
B3	30	169372.29	205802.71	21.51	169372.29	0	17.42	169372.29	0
C1	40	250556.77	315724.23	26.01	251502.50	0.377451	120.81	251502.50	0.377451
C2	40	215020.23	251247.77	16.85	215020.23	0	68.71	215020.23	0
C3	40	199345.96	204746.04	2.71	199345.96	0	33.53	199345.96	0
C4	40	195366.63	228197.37	16.80	195366.63	0	29.86	195366.63	0
D1	38	322530.13	357408.87	10.81	322530.13	0	107.14	322530.13	0
D2	38	316708.86	337234.14	6.48	316708.86	0	113.00	316708.86	0
D3	38	239478.63	266258.37	11.18	239478.63	0	74.01	239478.63	0
D4	38	205831.94	226167.06	9.88	205881.56	0.024107	62.35	205881.56	0.024107
E1	45	238879.58	296152.42	23.98	238879.58	0	136.57	238879.58	0
E2	45	212263.11	280137.89	31.98	212263.11	0	78.25	212263.11	0
E3	45	206659.17	269169.83	30.25	206659.17	0	76.29	206659.17	0
F1	60	264299.60	290315.40	9.84	264299.60	0	190.13	264299.60	0
F2	60	265653.47	336390.53	26.63	265653.47	0	193.14	265653.47	0
F3	60	241120.77	300311.23	24.55	241120.77	0	143.54	241120.77	0
F4	60	233861.85	294197.15	25.80	233861.85	0	132.61	233861.85	0
G1	57	306305.40	373821.60	22.04	306305.40	0	231.80	306305.40	0
G2	57	245440.99	297112.01	21.05	245440.99	0	207.47	245440.99	0
G3	57	229507.48	275023.52	19.83	229507.48	0	217.11	229507.48	0
G4	57	235251.47	297011.53	26.25	235251.47	0	186.30	235251.47	0
G5	57	221730.35	289113.65	30.39	221730.35	0	148.52	221730.35	0
G6	57	213457.45	259111.55	21.39	213457.45	0	189.35	213457.45	0
H1	68	268933.06	338112.94	25.72	268933.06	0	170.88	268933.06	0
H2	68	253365.50	313645.50	23.79	253365.50	0	247.09	253365.50	0
H3	68	247449.04	301342.96	21.78	247449.04	0	268.59	247449.04	0
H4	68	250220.77	291608.23	16.54	250220.77	0	288.60	250220.77	0
H5	68	246121.31	279871.69	13.71	246121.31	0	240.54	246121.31	0
H6	68	249135.32	291981.68	17.20	249135.32	0	196.63	249135.32	0
I1	90	351606.91	447451.09	27.26	351905.01	0.084782	304.07	351606.91	0
I2	90	309955.04	391123.96	26.19	311649.29	0.546612	299.13	309955.04	0
I3	90	294507.38	337231.62	14.51	294507.38	0	297.35	294507.38	0
I4	90	295999.65	368732.35	24.57	296903.67	0.305413	346.67	295999.65	0
I5	90	302524.33	372785.67	23.23	302529.34	0.001656	297.04	302524.33	0
J1	94	335593.42	341124.58	1.65	336975.93	0.41196	336.64	336165.45	0.170453
J2	94	310800.53	361098.47	16.18	311077.00	0.088954	360.68	**310417.21**[b]	−0.123333
J3	94	279219.21	331643.79	18.78	281458.02	0.801811	319.14	279219.21	0
J4	94	296773.38	348714.62	17.50	296773.38	0	325.78	296773.38	0
K1	113	395546.40	441532.60	11.63	397769.29	0.56198	558.88	395546.40	0
K2	113	363214.24	438771.76	20.80	365217.96	0.551663	578.11	363214.24	0
K3	113	366222.05	451210.95	23.21	367767.39	0.421968	616.89	366222.05	0
K4	113	349038.84	411982.16	18.03	350899.52	0.533087	576.46	349038.84	0
L1	150	426017.86	584527.14	37.21	426659.89	0.150705	898.18	426017.86	0
L2	150	402245.17	539122.83	34.03	402245.17	0	792.50	402245.17	0
L3	150	403886.22	435851.78	7.91	406873.02	0.739515	832.02	403886.22	0
L4	150	384844.01	498594.99	29.56	385133.49	0.07522	787.92	385133.49	0.07522
L5	150	388061.69	431388.31	11.16	388211.73	0.038664	835.22	388211.73	0.038664
M1	125	400860.79	491884.21	22.71	401255.14	0.098376	693.18	401255.14	0.098376

(*continued on next page*)

Table 6 (*continued*)

Problem number	No. of nodes	Best-known solution value (BS)[a]	Constructed initial solution	% Deviation from BS	***CLOVES*** solution	% Deviation from BS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from BS
M2	125	398908.71	428122.29	7.32	399239.75	0.082986	645.91	**397655.71**[b]	−0.314107
M3	125	377352.81	441123.19	16.90	378716.96	0.361505	672.32	378716.96	0.361505
M4	125	348624.42	428124.58	22.80	349471.24	0.242903	639.81	349471.24	0.242903
N1	150	408926.40	609121.60	48.96	409588.92	0.162014	797.09	409588.92	0.162014
N2	150	409280.16	458123.84	11.93	409280.16	0	825.63	409280.16	0
N3	150	396167.85	499201.15	26.01	398275.45	0.531997	798.61	398275.45	0.531997
N4	150	397753.86	499921.14	25.69	397810.09	0.014137	844.80	397810.09	0.014137
N5	150	376431.84	431295.16	14.57	376431.84	0	873.94	376431.84	0
N6	150	377665.19	431211.81	14.18	383264.53	1.48262	880.57	383264.53	1.48262
Average relative percentage deviation				20.08		0.140195			0.0506776

[a] Osman and Wassan (2002).
[b] ***CLOVES*** solution better than the existing best-known solution.

5.1.1. VRP data-sets of Christofides and Eilon (1969)

These are denoted as *W-nY-kZ*, where *W* is equal to instance series *E*, *Y*, the number of nodes and *Z*, the number of routes. The results are shown in Table 2.

5.1.2. VRP data-sets of Christofides et al. (1979)

The test problems which have been taken from Christofides et al. (1979) are identified with their original number, prefixed, with letter *C*. The problems *C*1 to *C*5, *C*11 and *C*12 without maximum route length constraint are taken. The results are shown in Table 3.

5.1.3. VRP data-sets of Augerat et al. (1995)

These instances are denoted as *X-nY-kZ*, where *X* is equal to instance series (i.e., *A*, *B*, *P*), *Y*, the number on nodes and *Z*, the number of routes. The results are shown in Table 4.

All the VRP data-sets are available at http://www.branchandcut.org/VRP, maintained by T.Ralphs (LehighUniversity, Bethlehem, PA).

5.1.4. Derived VRPDP data-sets from Christofides et al. (1979)

The next set of data is taken from literature pertaining to VRP. Here, the best solutions reported for VRP are taken as lower bounds, while we superimpose the VRPDP features on this set of problems and the data-sets are named as Z1 to Z5, Z11 and Z12. The results are shown in Table 5.

5.1.5. VRPB data-sets of Goetschalckx and Jacobs-Blecha (1989)

The Goetschalckx and Jacobs-Blecha (1989) problems have long existed as benchmarks among data-sets for VRPB. In the test problems, the total number of nodes range from 25 to 150 with 14 different groups. In each group, the capacity and the fleet-sizes vary. In each group, line-haul percentages of 50%, 66%, and 80% are considered. The problem input data and the solution values are available in the internet website http://www.isye.gatech.edu/people/faculty/Marc_Goetschalckx/cali/Lineback/EvalOnly/Lhbhcase.txt.

In the literature, the Euclidean distances between pairs of nodes are computed in two different ways. In one method, the Euclidean distances are shown in a real-valued cost matrix; the solution is rounded off to the nearest integer. In the other method, the Euclidean distances are multiplied by 10 and rounded off to the nearest integer and presented as an integer-valued cost matrix. Finally, this solution is divided by 10 and rounded off to the nearest integer. Both forms are used for comparing the results. Table 6 reports the results obtained by ***CLOVES*** using real-valued cost matrices, while Table 7 reports those obtained by ***CLOVES*** using integer-valued cost matrices. An illustrated example from this data-set is given in Appendix.

Table 7
Comparison of ***CLOVES*** with best solution value of VRPB data-sets: Goetschalckx and Jacobs-Blecha (1989) using an integer-valued cost matrix

Problem number	No. of nodes	Best-known solution value (BS)[a]	Constructed initial solution	% Deviation from BS	***CLOVES*** solution	% Deviation from BS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from BS
A1	25	229884	269045	17.04	229884	0	19.40	229884	0
A2	25	180117	215461	19.62	180117	0	20.90	180117	0
A3	25	163403	182533	11.71	163403	0	17.91	163403	0
A4	25	155795	197512	26.78	155795	0	16.42	155795	0
B1	30	239077	276862	15.80	239077	0	40.30	239077	0
B2	30	198045	246142	24.29	198045	0	35.82	198045	0
B3	30	169368	205907	21.57	169368	0	16.42	169368	0
C1	40	250557	315624	25.97	250557	0	102.98	250557	0
C2	40	215019	251349	16.90	215019	0	67.16	215019	0
C3	40	199344	203988	2.33	199344	0	31.34	199344	0
C4	40	195365	227899	16.65	195365	0	26.87	195365	0
D1	38	322533	357301	10.78	322533	0	105.97	322533	0
D2	38	316711	336724	6.32	316711	0	110.45	316711	0
D3	38	239482	265143	10.72	239482	0	71.64	239482	0
D4	38	205834	225442	9.53	205834	0	58.21	205834	0
E1	45	238880	295874	23.86	238880	0	116.42	238880	0
E2	45	212262	280469	32.13	212262	0	73.13	212262	0
E3	45	206658	269571	30.44	206658	0	68.66	206658	0
F1	60	263175	291960	10.94	264300	0.427472	188.06	263175	0
F2	60	265214	339160	27.88	265654	0.165904	191.04	265214	0
F3	60	241121	301401	25.00	241121	0	140.30	241121	0
F4	60	233861	295298	26.27	233861	0	128.36	233861	0
G1	57	306304	373917	22.07	307274	0.316679	216.41	306304	0
G2	57	245441	296362	20.75	245441	0	195.52	245441	0
G3	57	229506	276105	20.30	230170	0.289317	185.07	229506	0
G4	57	232646	300734	29.27	235251	1.119727	182.09	232646	0
G5	57	221731	285013	28.54	221731	0	138.80	221731	0
G6	57	213457	259292	21.47	213457	0	185.07	213457	0
H1	68	268933	334948	24.55	270525	0.591969	159.70	268933	0
H2	68	253366	324654	28.14	253366	0	222.38	253366	0
H3	68	247449	307935	24.44	247449	0	265.67	247449	0
H4	68	250221	295409	18.06	250221	0	282.08	250221	0
H5	68	246121	277267	12.65	246121	0	232.83	246121	0
H6	68	249136	295409	18.57	249136	0	183.58	249136	0
I1	90	351609	442718	25.91	356381	1.357189	286.56	351609	0
I2	90	309957	394715	27.35	313917	1.277597	289.55	309957	0
I3	90	294509	339974	15.44	297318	0.953791	277.61	294509	0
I4	90	295988	365146	23.37	295988	0	295.52	295988	0
I5	90	302525	370397	22.44	302708	0.060491	277.61	302525	0
J1	94	335590	345564	2.97	335590	0	302.98	**335340**[b]	−0.0745
J2	94	310798	362398	16.60	310798	0	307.46	**310417**[b]	−0.12259
J3	94	279220	338654	21.29	282535	1.187236	311.93	279220	0
J4	94	296774	342746	15.49	298284	0.508805	304.47	296774	0
K1	113	395544	449316	13.59	395544	0	546.26	**394524**[b]	−0.25787
K2	113	363213	433663	19.40	363231	0.004956	540.29	**362657**[b]	−0.15308
K3	113	366222	455587	24.40	371322	1.392598	555.21	366222	0
K4	113	349037	419529	20.20	359642	3.03836	543.27	349037	0
L1	150	426021	588725	38.19	426021	0	765.65	426021	0
L2	150	402246	531033	32.02	402246	0	774.61	402246	0
L3	150	403806	433942	7.46	403806	0	777.59	403806	0
L4	150	384343	497026	29.32	384343	0	770.13	384343	0
L5	150	388060	434378	11.94	394576	1.679122	780.58	394576	1.679122
M1	125	400858	496587	23.88	400858	0	623.87	**399747**[b]	−0.27716
M2	125	398902	423778	6.24	398902	0	631.33	**398857**[b]	−0.01128
M3	125	377352	449393	19.09	383448	1.615468	628.34	377352	0

(*continued on next page*)

Table 7 (*continued*)

Problem number	No. of nodes	Best-known solution value (BS)[a]	Constructed initial solution	% Deviation from BS	***CLOVES*** solution	% Deviation from BS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from BS
M4	125	348624	423075	21.36	356311	2.204954	625.36	348624	0
N1	150	408921	601419	47.07	408921	0	779.09	408921	0
N2	150	409275	454998	11.17	409275	0	771.62	409275	0
N3	150	396162	497150	25.49	396162	0	780.58	396162	0
N4	150	397748	495167	24.49	410694	3.254825	789.53	410694	3.254825
N5	150	376426	448301	19.09	389349	3.433078	786.55	376426	0
N6	150	377660	444603	17.73	377660	0	792.52	**377408**[b]	−0.06673
Average relative percentage deviation				20.39		0.401283			0.064044

[a] Osman and Wassan (2002).
[b] ***CLOVES*** solution better than the existing best-known solution.

Table 8
Comparison of ***CLOVES*** with Best Solution value of VRPB data-sets: Toth and Vigo (1996)

Problem number	No. of nodes	Best-known solution value (BS)[a]	Constructed initial solution	% Deviation from BS	***CLOVES*** solution	% Deviation from BS	CPU time (seconds)	***CLOVES*** Best solution	% Deviation from BS
eil22_50	21	371	492	32.61	371	0	15.83	371	0
eil22_66	21	366	493	34.70	366	0	17.06	366	0
eil22_80	21	375	526	40.27	375	0	14.62	375	0
eil23_50	22	682[b]	778	14.08	682	0	13.40	682	0
eil23_66	22	649[b]	802	23.57	649	0	32.89	649	0
eil23_80	22	623[b]	681	9.31	623	0	29.24	623	0
eil30_50	29	501	728	45.31	503	0.399202	13.40	501	0
eil30_66	29	537	802	49.35	537	0	84.05	537	0
eil30_80	29	514	778	51.36	517	0.583658	54.82	514	0
eil33_50	32	738	946	28.18	739	0.135501	25.58	738	0
eil33_66	32	750	1157	54.27	750	0	21.93	750	0
eil33_80	32	748	1081	44.52	761	1.737968	86.49	748	0
eil51_50	50	559	696	24.51	562	0.536673	90.15	559	0
eil51_66	50	548	656	19.71	548	0	58.47	548	0
eil51_80	50	570	703	23.33	574	0.701754	47.51	570	0
eilA76_50	75	739	922	24.76	743	0.541272	95.02	739	0
eilA76_66	75	768	1057	37.63	780	1.5625	59.69	768	0
eilA76_80	75	811	972	19.85	833	2.7127	56.04	811	0
eilB76_50	75	809	1162	43.63	825	1.97775	153.49	809	0
eilB76_66	75	873	1228	40.66	891	2.061856	155.93	873	0
eilB76_80	75	931	1155	24.06	947	1.718582	114.51	947	1.718582
eilC76_50	75	713	934	31.00	715	0.280505	104.77	713	0
eilC76_66	75	734	966	31.61	742	1.089918	176.63	734	0
eilC76_80	75	736	956	29.89	753	2.309783	159.58	736	0
eilD76_50	75	690	860	24.64	690	0	151.05	690	0
eilD76_66	75	715	898	25.59	717	0.27972	148.62	715	0
eilD76_80	75	694	884	27.38	710	2.305476	113.29	710	2.305476
eilA101_50	100	842	1021	21.26	847	0.593824	151.05	842	0
eilA101_66	100	852	1125	32.04	867	1.760563	130.35	867	1.760563
eilA101_80	100	875	1104	26.17	900	2.857143	181.51	875	0
eilB101_50	100	936	1214	29.70	952	1.709402	216.84	936	0
eilB101_66	100	998	1318	32.06	1040	4.208417	230.23	1040	4.208417
eilB101_80	100	1021	1332	30.46	1059	3.721841	190.04	1059	3.721841
Average relative percentage deviation				31.14		1.084424			0.415602

[a] Osman and Wassan (2002).
[b] Toth and Vigo (1999).

5.1.6. VRPB data-sets of Toth and Vigo (1996)

Toth and Vigo (1996) generated 33 data-sets from 11 classical instances found in VRP literature. The VRPB instances range from 21 to 100 nodes. For each VRP data-set, three VRPB instances are generated with line-haul percentages of 50, 66 and 80, respectively. The results are shown in Table 8.

The average deviation for each data-set and the average deviation from the best-known solutions clearly indicate that ***CLOVES*** has performed well on all the data-sets. It consistently obtains good quality solutions at reasonable computing times. Solutions better than the existing best are indicated by a (#) symbol in the tables.

6. Conclusion

This study has addressed the vehicle routing problem with delivery and pick-up (VRPDP) with an imposed sequence of pure delivery, delivery and pick-up and pure pick-up. For this *np*-hard problem, we have developed a four-stage algorithm called ***CLOVES***, which constructs a good initial solution and uses it to achieve rapid convergence in the final intensive search.

In the absence of published data-sets that incorporate VRPDP features, we have used published data-sets, some incorporating pick-ups (VRPB) and some with only delivery (VRP), for comparison. As can be seen from the results, ***CLOVES*** promises to be a useful tool for certain classes of VRPs. The study can be extended, in future, to consider constraints on maximum route length and on maximum duration of each trip.

Appendix

Illustration of ***CLOVES*** for an integer-valued data-set (K1) of Goetschalckx and Jacobs-Blecha (1989).

Data-set: K1

Phase 1: Clustering of Nodes

Delivery Clusters:

R1: (44, 67, 50, 26, 43, 10, 56, 72)
R2: (35, 55, 24, 11, 36, 61, 46, 21)
R3: (27, 66, 25, 17, 7, 31, 38)
R4: (39, 37, 47, 6, 5, 22, 32, 8)
R5: (58, 69, 63, 2, 73, 74, 34)
R6: (70, 60, 13, 29, 19, 64)
R7: (12, 53, 45, 71, 14, 33, 42, 20, 62)
R8: (4, 3, 49, 65, 75, 9, 40)
R9: (23, 15, 54, 28, 59, 18, 51, 52)
R10: (48, 16, 68, 41, 57, 1, 30)

Pick-up Clusters:

R1: (30)
R2: (10, 37, 3, 5, 29, 12, 32)
R3: (7, 31, 11, 6)
R4: (1, 17, 15, 33)
R5: (9, 36)
R6: (27, 24, 22, 4)
R7: (18, 8, 13, 26)
R8: (25, 20, 21, 23, 14)
R9: (16, 34)
R10: (2, 35, 19, 28, 38)

Phase 2: Orientation of Nodes to a Route
Delivery Clusters:

R1: (10-50-67-43-72-44-56-26)
R2: (46-21-61-36-11-24-55-35)
R3: (66-17-25-27-31-38-7)
R4: (32-8-22-47-5-6-37-39)
R5: (2-63-73-69-58-74-34)
R6: (60-70-13-29-64-19)
R7: (12-53-45-71-14-33-62-42-20)
R8: (4-3-49-65-75-40-9)
R9: (23-15-52-51-54-18-59-28)
R10: (30-16-48-1-68-57-41)

Pick-up Clusters:

R1: (30)
R2: (10-12-37-29-5-3-32)
R3: (11-6-31-7)
R4: (1-17-15-33)
R5: (9-36)
R6: (4-22-24-27)
R7: (26-13-18-8)
R8: (14-23-20-21-25)
R9: (16-34)
R10: (2-35-19-28-38)

Phase 3: Vehicle Allocation – Solution obtained by Constructive Heuristic

R1: (10-50-67-43-72-44-56-26--30)
R2: (46-21-61-36-11-24-55-35--32-3-5-29-37-12-10)
R3: (66-17-25-27-31-38-7--7-31-6-11)
R4: (32-8-22-47-5-6-37-39--33-15-17-1)
R5: (34-74-58-69-73-63-1--9-36)
R6: (19-64-29-13-70-60--4-22-24-27)
R7: (20-42-62-33-14-71-45-53-12--26-13-18-8)
R8: (4-3-49-65-75-40-9--25-21-20-23-14)
R9: (23-15-52-51-54-18-59-28--34-16)
R10: (41-57-68-1-48-16-30--2-35-19-28-38)

Objective Function Value (Total Distance Traveled) = 449316
Phase 4: Intensive Search using GA

R1: (44-67-50-26-43-10-56-72--30)
R2: (35-55-24-11-36-61-46-21--10-37-3-5-29-12-32)
R3: (27-66-17-25-7-31-38--7-31-11-6)
R4: (39-37-47-6-5-22-32-8--1-17-15-33)
R5: (58-69-63-2-73-74-34--9-36)
R6: (70-60-13-29-19-64--27-24-22-4)
R7: (12-53-45-71-14-33-42-20-62--18-8-13-26)
R8: (4-3-49-65-75-40-9--25-20-21-23-14)

R9: (23-15-54-28-59-18-51-52--16-34)
R10: (48-16-68-41-57-1-30--2-35-19-28-38)

Objective Function Value = 394 524.
Percent improvement over constructive heuristic = 13.88%.
Objective Function Value for existing best solution = 395 544.

References

Anderberg, M.R., 1973. Cluster Analysis for Applications. Academic press, New York.
Anily, Sh., 1996. The vehicle-routing problem with delivery and back-haul options. Naval Research Logistics 43, 415–434.
Anily, Sh., Mosheiov, G., 1994. The traveling salesman problem with delivery and backhauls. Operations Research Letters 16, 11–18.
Augerat, P., Belenguer, J.M., Benavent, E., Corber'an, A., Naddef, D., Rinaldi, G. 1995. Computational Results with a Branch and Cut Code for the Capacitated Vehicle Routing Problem. Research Report 949-M, Universit'e Joseph Fourier, Grenoble, France.
Casco, D.O., Golden, B.L., Wasil, E.A., 1988. Vehicle routing with backhauls: Models, algorithms, and case studies. In: Golden, B.L., Assad, A.A. (Eds.), Vehicle Routing: Methods and Studies. Elsevier, Amsterdam, pp. 127–147.
Christofides, N., Eilon, S., 1969. Expected distances in distribution problems. Operational Research Quarterly 20 (4), 437–443.
Christofides, N., Mingozzi, A., Toth, P., 1979. The vehicle routing problem. In: Christofides, N., Mingozzi, A., Toth, P., Sandi, C. (Eds.), Combinatorial Optimization. Wiley, New York, pp. 315–338.
Clarke, G., Wright, J., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–581.
Deif, I., Bodin, L. 1984. Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauling. In: Kidder, A. (Eds.), Proceedings of the Babson Conference on Software Uses in Transportation and Logistic Management, Babson Park, pp. 75–96.
Dethloff, J., 2001. Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up. OR Specktrum 23, 79–96.
Fisher, M., Jaikumar, R., 1981. A generalized assignment heuristic for vehicle routing. Networks 11, 109–124.
Gen, M., Cheng, R., 1997. Genetic Algorithms and Engineering Design. Wiley Interscience.
Gen, M., Liu, B. 1995. A genetic algorithm for nonlinear goal programming. Technical Report, ISE95-5, Ashikaga Institute of Technology, Ashikaga, Japan.
Gendreau, M., Laporte, G., Hertz, A., 1997. An approximation algorithm for the traveling salesman problem with backhauls. Operations Research 45, 639–641.
Gendreau, M., Laporte, G., Potvin, J-Y. 1998. Metaheuristics for the vehicle routing problem. GERAD research report G-98-52, Montreal, Canada.
Gendreau, M., Laporte, G., Vigo, D., 1999. Heuristics for the travelling salesman problem with pickup and delivery. Computers and Operations Research 26, 699–714.
Goetschalckx, M., Jacobs-Blecha, Ch., 1989. The vehicle routing problem with backhauls. European Journal of Operational Research 42, 39–51.
Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman.
Golden, B.L., Assad, A.A., 1988. Vehicle Routing: Methods and StudiesStudies in Management Science and Systems, vol. 16. Elsevier Science Publication, Amsterdam, Netherlands.
Golden, B.L., Baker, E.K., Alfaro, J.L., Schaffer, J.R. 1985. The Vehicle routing problem with backhauling: Two approaches. Working paper MS/S 85-017, University of Maryland, College Park.
Holland, J.H., 1975. Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor.
Jacobs-Blecha, Ch., Goetschalckx, M. 1993. The vehicle routing problem with backhauls: Properties and solution algorithms. Materials Handling Research Centre Technical Report MHRC-TR-88-13, Georgia Institute of Technology, Atlanta.
Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., Shmoys, D.B., 1985. The Traveling Salesman Problem. Wiley, New York.
Lenstra, J.K., Rinnooy Kan, A.H.G., 1976. On general routing problems. Networks 6, 273–280.
Min, H., 1989. The multiple vehicle routing problem with simultaneous delivery and pick-up points. Transportation Research A 23A, 377–386.
Min, H., Current, J., Schilling, D., 1992. The multiple depot vehicle routing problem with backhauling. Journal of Business Logistics 13, 259–288.
Mosheiov, G., 1994. The travelling salesman problem with pick-up and delivery. European Journal of Operational Research 79, 299–310.
Nagy, G., Salhi, S., 2005. Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. European Journal of Operational Research 162, 126–141.
Or, I. 1976. Traveling salesman type combinatorial problems and their relation to the logistics of blood banking. Ph.D. Dissertation. Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL.
Osman, I.H., Wassan, N.A., 2002. A reactive tabu search metaheuristic for the vehicle routing problem with backhauls. Journal of Scheduling 5, 263–285.
Petch, R.J., Salhi, S., 2003. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. Discrete Applied Mathematics 133 (1–3), 69–92.

Salhi, S., Nagy, G., 1999. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. Journal of the Operational Research Society 50, 1034–1042.

Sofge, D., Schultz, A., Jong, K.D., 2002. Evolutionary computational approaches to solving the multiple traveling salesman problem using a neighborhood attractor schema. Lecture Notes in Computer Science, vol. 2279. Springer-Verlag, Heidelberg, pp. 153–162.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F., Potvin, J., 1997. A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation Science 31, 170–186.

Thangiah, S.R., Sun, T., Potvin, J-Y., 1996. Heuristic approaches to vehicle routing with backhauls and time windows. Computers and Operations Research 23, 1043–1057.

Toth, P., Vigo, D., 1996. A heuristic algorithm for the vehicle routing problem with backhauls. In: Bianco, L., Toth, P. (Eds.), Advanced Methods in Transportation Analysis. Springer, Berlin, pp. 585–608.

Toth, P., Vigo, D., 1997. An exact algorithm for the vehicle routing problem with backhauls. Transportation Science 31, 372–385.

Toth, P., Vigo, D., 1999. A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls. European Journal of Operational Research 113, 528–543.

Wade, A.C., Salhi, S., 2002. An investigation into a new class of vehicle routing problem with backhauls. Omega 30, 479–487.

Yano, C., Chan, T., Richter, L., Cutler, T., Murty, K., McGettigan, G., 1987. Vehicle routing at quality stores. Interfaces 17, 52–63.