RESEARCH ARTICLE
doi: 10.2306/scienceasia1513-1874.2012.38.307

ScienceAsia 38 (2012): **307–318**

An improved Clarke and Wright savings algorithm for the capacitated vehicle routing problem

Tantikorn Pichpibul[a], **Ruengsak Kawtummachai**[b,*]

[a] School of Manufacturing Systems and Mechanical Engineering, Sirindhorn International Institute of Technology, Thammasat University, Pathumthani 12121 Thailand

[b] Faculty of Business Administration, Panyapiwat Institute of Management, Chaengwattana Road, Nonthaburi 11120 Thailand

*Corresponding author, e-mail: ruengsakkaw@pim.ac.th

Received 1 Aug 2011
Accepted 20 Jun 2012

ABSTRACT: In this paper, we have proposed an algorithm that has been improved from the classical Clarke and Wright savings algorithm (CW) to solve the capacitated vehicle routing problem. The main concept of our proposed algorithm is to hybridize the CW with tournament and roulette wheel selections to determine a new and efficient algorithm. The objective is to find the feasible solutions (or routes) to minimize travelling distances and number of routes. We have tested the proposed algorithm with 84 problem instances and the numerical results indicate that our algorithm outperforms CW and the optimal solution is obtained in 81% of all tested instances (68 out of 84). The average deviation between our solution and the optimal one is always very low (0.14%).

KEYWORDS: heuristics, optimization, tournament selection, roulette wheel selection

INTRODUCTION

The capacitated vehicle routing problem (CVRP) was initially introduced by Dantzig and Ramser[1] in their article on a truck dispatching problem and, consequently, became one of the most important and widely studied problems in the area of combinatorial optimization. Not only is the travelling salesman problem classified as nondeterministic polynomial time (NP) hard[2], but also the bin packing problem is a special case of CVRP. Accordingly, the CVRP has been concluded to be an NP-hard problem[3–5]. The basic concept of CVRP is to find a feasible set of vehicle routes that minimizes the total travelling distance and/or the total number of vehicles used. For each route, the vehicle departs from a given depot and returns to the same depot after completing the service. CVRP involves a single depot, a homogeneous fleet of vehicles, and a set of customers who require delivery of goods from the depot.

Since CVRP was first proposed in 1959[1], it has received much attention from researchers and practitioners. Therefore, numerous approaches and algorithms have also been developed. First, an exact algorithm, which is an algorithm that solves a problem to optimality by computing the distance of every feasible solution and then choosing a solution with minimum distance, was reported. The approach consists of a branch-and-bound algorithm[6], a branch-and-cut algorithm[7–9], and a branch-and-cut-and-price algorithm[10]. In these algorithms, CVRP instances involving more than 100 customers can rarely be solved to optimality due to a huge amount of computation time. Second, a heuristic algorithm, which is an algorithm that should find solutions among all feasible ones, composed of savings algorithm[11], sweep algorithm[12,13], sequential insertion algorithm[14], petal algorithm[15,16], two-phase insertion[17], cluster-first route-second algorithm[18], 2-petals algorithm[19], k-opt heuristic[20], Or-exchanges[21], and λ-interchanges[22]. These algorithms usually find a feasible solution (near optimal) fast and easily but they do not guarantee that the optimal solution will be found. Finally, a metaheuristic algorithm, which is an iterative improvement approach by combining a heuristic algorithm with intelligent ideas for exploring and exploiting the search space, composed of simulated annealing[22], tabu search[23,24], genetic algorithm[25,26], ant colony algorithm[27,28], memetic algorithm[29,30], active-guided evolution strategies[31], honey bees mating optimization algorithm[32], and particle swarm optimization algorithm[33,34]. In these algorithms, a good metaheuristic implementation can provide efficiently near-optimal solutions in a reasonable computation time.

The Clarke and Wright savings algorithm (CW)[11] is the most widely applied heuristic for solving CVRP

due to its simplicity of implementation and efficient calculation speed. CW has also been widely applied as a basis algorithm in many commercial routing packages. But CW without any improvements provides a solution that is far from the optimal one. However, several implementations of CW for CVRP and other types of VRP were also tackled by many researchers; we have summarized their methods as follows. First, Tillman[35] modified the savings formula[11] to solve the multi-depot vehicle routing problem by revising $c_{1,i}$ and $c_{j,1}$ to c_i^k and c_j^k which are cost or distance between nodes i and j to the nearest terminal k as in:

$$s_{i,j}^k = c_i^k + c_j^k - c_{i,j}.$$

Bodin et al[36] modified the savings formula[11] in the case of the Federal Express Corporation's aeroplane scheduling problem by deleting $c_{j,1}$ which is the travelling time from customer j to depot 1 in order to deal with one way flights (either outgoing or incoming aircraft) in:

$$s_{i,j} = c_{1,i} - c_{i,j}.$$

Goetschalckx and Jacobs-Blecha[37] compared their heuristic with the modified CW proposed by Deif and Bodin[38] in a case of vehicle routing problem with backhauls by adjusting the savings calculation to be:

$$s_{i,j} = s_{i,j} - \alpha s_{\max}.$$

Here $s_{\max}$ is an estimate of the maximum savings value and α is a penalty multiplier, $\alpha \in [0.1, 0.3]$. Vigo[39] compared his heuristic with parameterized CW consisting of route shape (λ) parameter[40,41] and weighted (μ) parameter[42] in an asymmetric capacitated vehicle routing problem as in:

$$s_{i,j} = c_{1,i} + c_{j,1} - \lambda c_{ij} + \mu |c_{1,i} - c_{j,1}|.$$

Here, λ is a parameter that controls the relative significance of direct arc between two customers ($\lambda \in [0, 3]$) and μ is the asymmetry between two customers with respect to their distances to the depot ($\mu \in [0, 1]$). Altınel and Öncan[43] introduced a new enhancement of the original CW in parameterized saving consisting of route shape (λ) parameter[40,41] and weighted (μ) parameter[42]. Their new heuristic with savings criterion considers the customer demand parameter (ν), which includes the demand of customers on a vehicle's capacity. Their proposed savings formula is as follows:

$$s_{i,j} = c_{1,i} + c_{j,1} - \lambda c_{ij} + \mu |c_{1,i} - c_{j,1}| + \nu \frac{d_i + d_j}{\bar{d}}.$$

Here d_i is the demand of customer i and $\bar{d}$ is the average demand of all customers. Considering these three independent parameters (λ, μ, ν), Altınel and Öncan[43] used a simple enumerative approach by varying parameter λ in the ranges of $[0.1, 2]$ and parameters μ and ν in the ranges of $[0, 2]$, and using a step size of 0.1. Totally, 8820 different solutions are obtained and the best solution is chosen. Therefore, this approach requires much computing time which can be reduced by using a genetic algorithm[44] and empirically adjusted greedy heuristics[45] to adjust the parameters. Juan et al[46] presented a simulation study in routing via the generalized CW which is a hybrid algorithm that combines the parallel version of CW with Monte Carlo simulation and state-of-the-art random number generators. Finally, CW is widely used to generate initial solutions for further application with other algorithms to get the improvement[47–49].

Due to our summary of the related CW studies as mentioned above, there are few literature published by Gaskell[40], Yellow[41], Paessens[42], Altınel and Öncan[43], and Juan et al[46] in which CW is the standalone algorithm applied to solve CVRP by concerning the modification of inside its procedure. But in those works, only Juan et al[46] successfully reported 50 optimal solutions in real Euclidean distance. Therefore, one of our motivations is to improve the CW by using our new competitive approach which can successfully reach the optimal solutions. In our algorithm, we simply have applied the parallel version of CW[11] combined with our approach adjusted from two genetic operators in the genetic algorithm including tournament and roulette wheel selections. It is shown that our algorithm is very simple to apply for solving CVRP.

In this paper, an improved Clarke and Wright savings algorithm (ICW) for CVRP by using the parallel version of CW combined with our approach is initially proposed. It is able to compete with CW related algorithms and other algorithms in terms of solution quality while solving CVRP.

PROBLEM DEFINITION

The CVRP can be stated as the problem in which vehicles based at a depot are required to serve geographical customers in order to satisfy known customer demands. All vehicles have the same loading capacity. All customers have non-negative demands. Each customer must be visited once by one vehicle. The loading of each vehicle cannot exceed the loading capacity. The objective of CVRP is to minimize the total travelling cost of all vehicles. The model

formulation of CVRP is shown below.

$$\text{Minimize} \quad \sum_{i\in N}\sum_{j\in N}\sum_{v\in V} c_{ij}x_{ij}^{v} \tag{1}$$

$$\text{subject to} \quad \sum_{v\in V} y_i^v = 1 \ \text{ for } i \in N, \tag{2}$$

$$\sum_{i\in N} x_{ij}^v = y_j^v \ \text{ for } j \in N \text{ and } v \in V, \tag{3}$$

$$\sum_{j\in N} x_{ij}^v = y_i^v \ \text{ for } i \in N \text{ and } v \in V, \tag{4}$$

$$\sum_{i\in N} d_i y_i^v \leqslant Q \ \text{ for } v \in V, \tag{5}$$

$$\sum_{i\in N} x_{i1}^v \leqslant 1 \ \text{ for } v \in V, \tag{6}$$

$$\sum_{j\in N} x_{1j}^v \leqslant 1 \ \text{ for } v \in V, \tag{7}$$

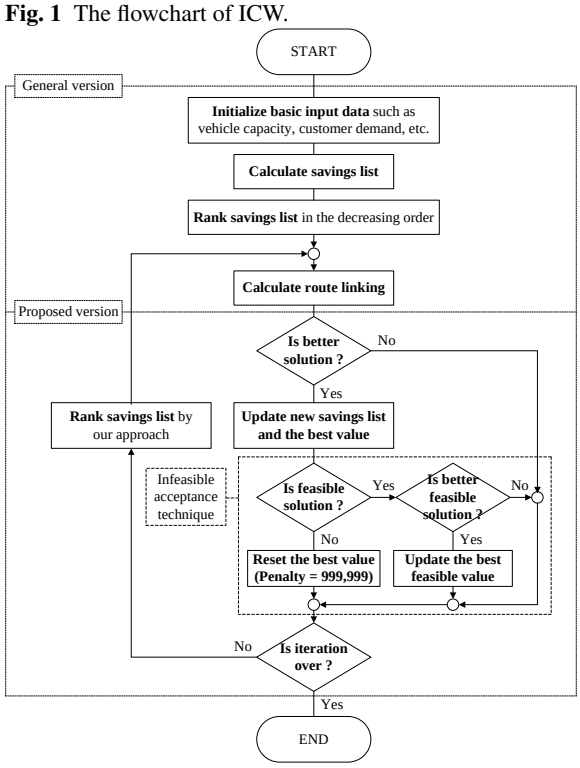
where N is number of customers including depot (depot is assigned to be customer 1), c_{ij} is the travelling cost between customer i and j ($i, j \in N, i \neq j$), V is number of vehicles ($v \in V$), Q is the loading capacity, d_i is the demand associated with each customer i, $x_{ij}^v \in \{0,1\}$ ($i, j \in N$; $v \in V$), $y_i^v \in \{0,1\}$ ($i \in N$; $v \in V$).

In this formulation, the objective function is expressed by (1) which states that the total travelling distance of all vehicles is to be minimized. Eq. (2) represents the constraint that each customer must be visited once by one vehicle, where $y_i^v = 1$ if vehicle v visits customer i, and 0 otherwise. It is guaranteed in (3) and (4) that each customer is visited and left with the same vehicle, where $x_{ij}^v = 1$ if vehicle v travels from customer i to customer j, and 0 otherwise. A constraint in (5) ensures that the total delivery demands of vehicle v do not exceed the vehicle capacity. Eq. (6) and (7) express that vehicle availability should not be exceeded.

METHOD OF APPROACH

In this section, an ICW which combines the advantages of CW with tournament and roulette wheel selections for solving CVRP is proposed. The concept of CW is based on the computation of savings for combining two customers into the same route. There are two versions of CW: sequential and parallel. In the sequential version, only one route is expanded until no more routes can be merged to this route. In contrast, in the parallel version several routes can be constructed in parallel. According to ICW, we implemented the parallel version of CW since it usually generates the better results than the corresponding sequential version[5,47]. Our work in this paper continues that of Pichpibul and Kawtummachai[50,51] which introduced the decision-making software based on the sequential version of CW and genetic algorithm, in which the roulette wheel selection is used as one of our genetic operators. The flowchart of ICW is given in Fig. 1 and is described in the following subsection.

Fig. 1 The flowchart of ICW.



The general Clarke and Wright savings algorithm

In the classical version, we first calculate the distance matrix $(d_{i,j})$ as:

$$d_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}.$$

Here, x_i, y_i and x_j, y_j are the geographical locations of customer i and j. Second, the savings value between customer i and j is calculated:

$$s_{i,j} = d_{1,i} + d_{j,1} - d_{i,j}.$$

Here, $d_{1,i}$ is the travelling distance between depot 1 and customer i. Third, all savings values are sorted in the decreasing order. Beginning with the topmost entry in the list (the largest $s_{i,j}$). Finally, starting from the top of the savings list, CW includes link (i, j) in a route if no route constraints will be violated

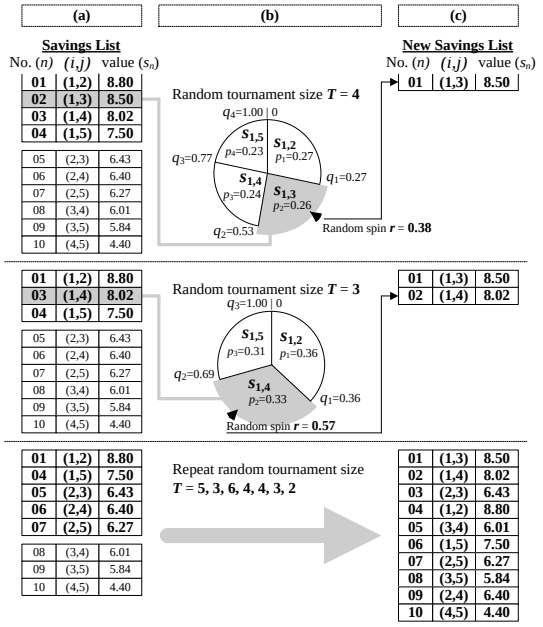
through the inclusion of customer i and j in that route. The route constraints are as follows: (a) Either, neither i nor j have already been assigned to a route. (b) Exactly, one of the two customers (i or j) has already been included in an existing route and that customer is not interior to that route (a customer is interior to a route if it is not adjacent to the depot 1 in the order of traversal of customers). (c) Both customer i and j have already been included in two different existing routes and neither customer is interior to its route. The route linking is repeated to process the next entry in the savings list until no feasible link is possible. In the case of non-routed customers, each is assigned by a route that begins at depot 1, visits the unassigned customer and returns to the same depot 1.

The proposed Clarke and Wright savings algorithm

In the proposed version, ICW is an iterative improvement approach designed to find the global optimum solutions. In contrast to CW which always sorts link (i, j) in the decreasing order to generate standard savings list, our approach arranges link (i, j) by adapting the combination of ideas between tournament and roulette wheel selections to generate a new savings list. Goldberg et al[52] gave an example of the tournament selection which randomly chooses a set of chromosomes and picks out the best one from the set for reproduction. The number of chromosomes in the set is called tournament size. A common tournament size is two which is called binary tournament. Fitness proportionate selection or roulette wheel selection was introduced by Holland[53]. His basic idea is to determine the selection or survival probability for each chromosome proportional to the fitness value. The example of our approach is shown in Fig. 2. The new savings list replaces the previous savings list only if the current solution is better than the previous one. In case that the stopping condition is satisfied, ICW will be terminated. Here, the total number of iterations and the number of consecutive iterations without any improvement in the best found solution are employed as the stopping condition. Notice that, ICW may generate the infeasible solution in which the number of available vehicles is inadequate at some iteration. In this case, the savings list will not be selected to be a new one by our approach. Therefore, it always links routes that belong to the same savings list. In order to avoid this problem, we present the infeasible acceptance technique which resets the best value equal to very large penalty value (999 999) when the better solution found is infeasible.

In Fig. 2, we have illustrated the example of

Fig. 2 The example of our approach to generate the new savings list.



generating the new savings list by our approach in case of five customers. Fig. 2a shows initial savings values, which represents the savings list sorted by the decreasing order in case of the first iteration of ICW, or represents the savings list derived from the previous better one. In Fig. 2b, our approach begins with adapting the tournament selection. The tournament size is a random number between two and six, and a set of savings values is chosen from the savings list. In the general tournament selection approach, the best savings value with minimum distance is picked out from the set. But in contrast to our tournament selection, the savings value is picked out from the set by the roulette wheel selection process. For savings number n with savings value s_n, its selection probability p_n and cumulative probability q_n are calculated as:

$$p_n = s_n / \sum_{i \in T} s_i \quad \text{for } n \in T$$

$$q_n = \sum_{i \in n} p_i \quad \text{for } n \in T$$

Here, T is the tournament size. Therefore, a roulette wheel is created by these probabilities. The selection method starts by spinning the roulette wheel with a random number r from the range between 0 and 1. If $r \leqslant q_1$, then choose the first savings value s_1; otherwise, choose the nth savings value s_n $(2 \leqslant n \leqslant T)$ such that $q_{n-1} < r \leqslant q_n$. This single savings

value is collected into the new savings list represented by Fig. 2c, and is discarded from next tournament selection operation in order to avoid the duplicate savings value. Hence, remaining savings values will be repeatedly executed by tournament selection until the last saving value is determined in the savings list, and that value is automatically collected into the new saving list.

EXPERIMENTAL ANALYSIS

This section presents the numerical experiment of ICW. We use the benchmark problem instances available in the literature to verify the efficiency and effectiveness of our algorithm.

Implementation and hardware

The ICW has been implemented in VISUAL BASIC 6.0 on an Intel Core i7 CPU 860 clocked at 2.80 GHz with 1.99 GB of RAM on a Windows XP platform. In our algorithm, some parameters have to be preset before the execution. For ICW, we have set the number of tournament sizes to be a random number from 3–9 and the total number of iterations equal to 10 000. Moreover, ICW is also terminated after 1000 consecutive iterations without any improvement in the best found solution.

Benchmark instances

In order to test ICW, we used five well-known datasets of CVRP (composed of 84 instances). These considered datasets are symmetric with vehicle capacity constraints and number of available vehicles restrictions. Their characteristics are listed in Table 1. Among 84 problem instances, 70 instances are from Augerat et al[8] referred to as A, B, and P. Next, 11 instances are from Christofides and Eilon[54] referred to as E. The last three instances are taken from Fisher[7] referred to as F. All problem instances have very tight vehicle capacity constraints in which the ratios of demand to capacity calculated by (8) below and shown in Table 1 are close to 1.0, except the E-n23-k3 problem which is equal to 0.75. The standard deviation of customer demands for each problem depicted in Table 1 indicates that the customer demands in Fisher's[7] benchmark problem vary much more than in the other benchmark problems.

$$\text{Ratio of demand to capacity} = \sum_{i \in N} d_i / VQ \qquad (8)$$

Here, d_i is the delivery demand at customer i, V is the number of vehicles, and Q is the vehicle capacity.

The benchmark problem sizes that we emphasized in this paper are classified as small-scale (less than 50 customers) and medium-scale (between 51 and 100 customers). All considered problems here are symmetric with different situations, e.g., uniformly

Table 1 The characteristics for five well-known datasets of CVRP.

Instance	Capacity	RDC[a]	SD[b]
A-n32-k5	100	0.82	7.26
A-n33-k5	100	0.89	6.35
A-n33-k6	100	0.90	11.40
A-n34-k5	100	0.92	7.24
A-n36-k5	100	0.88	6.59
A-n37-k5	100	0.81	8.13
A-n37-k6	100	0.95	11.14
A-n38-k5	100	0.96	6.94
A-n39-k5	100	0.95	8.07
A-n39-k6	100	0.88	11.85
A-n44-k6	100	0.95	6.57
A-n45-k6	100	0.99	6.84
A-n45-k7	100	0.91	6.94
A-n46-k7	100	0.86	7.23
A-n48-k7	100	0.89	6.80
A-n53-k7	100	0.95	8.68
A-n54-k7	100	0.96	7.76
A-n55-k9	100	0.93	9.82
A-n60-k9	100	0.92	9.38
A-n61-k9	100	0.98	9.78
A-n62-k8	100	0.92	7.96
A-n63-k9	100	0.97	7.42
A-n63-k10	100	0.93	9.82
A-n64-k9	100	0.94	9.20
A-n65-k9	100	0.97	7.34
A-n69-k9	100	0.94	8.10
A-n80-k10	100	0.94	7.55
B-n31-k5	100	0.82	6.09
B-n34-k5	100	0.91	15.59
B-n35-k5	100	0.87	7.93
B-n38-k6	100	0.85	7.32
B-n39-k5	100	0.88	6.52
B-n41-k6	100	0.95	5.67
B-n43-k6	100	0.87	6.91
B-n44-k7	100	0.92	10.99
B-n45-k5	100	0.97	7.33
B-n45-k6	100	0.99	6.51
B-n50-k7	100	0.87	10.25
B-n50-k8	100	0.92	10.82
B-n52-k7	100	0.87	6.83
B-n56-k7	100	0.88	6.91
B-n57-k9	100	0.89	7.27
B-n64-k9	100	0.98	9.10
B-n66-k9	100	0.96	6.54
B-n67-k10	100	0.91	7.36
B-n68-k9	100	0.93	7.96
B-n78-k10	100	0.94	7.51

Table 1 (Cont.).

Instance	Capacity	RDC[a]	SD[b]
P-n16-k8	35	0.88	8.42
P-n19-k2	160	0.97	7.71
P-n20-k2	160	0.97	8.15
P-n21-k2	160	0.93	7.42
P-n22-k2	160	0.96	7.31
P-n22-k8	3000	0.94	630.19
P-n23-k8	40	0.98	7.43
P-n40-k5	140	0.88	8.40
P-n45-k5	150	0.92	8.23
P-n50-k7	150	0.91	7.31
P-n50-k8	120	0.99	7.31
P-n50-k10	100	0.95	7.31
P-n51-k10	80	0.97	8.06
P-n55-k7	170	0.88	7.07
P-n55-k8	160	0.81	7.07
P-n55-k10	115	0.91	7.07
P-n60-k10	120	0.95	7.07
P-n60-k15	80	0.95	7.07
P-n65-k10	130	0.94	7.01
P-n70-k10	135	0.97	7.52
P-n76-k4	350	0.97	7.96
P-n76-k5	280	0.97	7.96
P-n101-k4	400	0.91	8.87
E-n22-k4	6000	0.94	630.19
E-n23-k3	4500	0.75	846.61
E-n30-k3	4500	0.94	610.64
E-n33-k4	8000	0.92	809.21
E-n51-k5	160	0.97	8.06
E-n76-k7	220	0.89	7.96
E-n76-k8	180	0.95	7.96
E-n76-k10	140	0.97	7.96
E-n76-k14	100	0.97	7.96
E-n101-k8	200	0.91	8.87
E-n101-k14	112	0.93	8.87
F-n45-k4	2010	0.90	258.55
F-n72-k4	30 000	0.96	3009.05
F-n135-k7	2210	0.95	187.32

[a] Ratio of demand to capacity.
[b] Standard deviation of customer demand.

and not uniformly dispersed customers, clustered and not clustered, with a centred or not centred depot. One of our motivations is to understand and develop a new approach that we can apply to every situation. The details of each problem are explained next. Augerat et al[8] proposed three datasets (A, B, and P). For the instances in dataset A, both customer locations and demands are randomly generated. The customer locations in dataset B are clustered instances. The modified version of other instances is dataset P, in which the problem ranges in size from 16–101 customers including the depot. The fourth dataset was proposed by Christofides and Eilon[54]. The customers are randomly distributed in the plane and the depot is either in the centre or near to it. The problem ranges in size from 22–101 customers including the depot. The fifth dataset was proposed by Fisher[7]. Each instance represents a day of grocery deliveries from the Peterboro and Bramalea, Ontario terminals, respectively, of National Grocers Limited, and the depot is not centred. The problem ranges in size from 45–135 customers including the depot.

Computational results and the comparison analysis of ICW

According to the optimal solutions[9, 10, 55] that have appeared in the literature for five well-known datasets of CVRP, the Euclidean distances, which are the distances between customers rounded to be the closest integer value, are used by following the TSPLIB standard[56]. Most earlier studies also relied on integer distance, except Juan et al[46] who reported their finding using real distance in double precision. Moreover, Battarra et al[44] and Corominas et al[45] both reported only the average percentage improvements of their solutions over CW solutions in each benchmark problem. In contrast, we do not only consider the improvements of our solutions over CW solutions but also show the performance of our solutions by comparing ICW with the algorithms for CVRP (Table 2).

In addition, a few works on integer distance, in which the problems were solved in all instances, were presented by AÖ, GN, H, and NJK. This is, therefore, one of our motivations to show the performance of our results by fair comparison with those results. In order to further evaluate our solution quality, several works on integer distance in which the problems were solved in some instances presented by AK, GGW, NMP, XK, ML, CYW, BHK, GPV, and KS, were compared with our solutions individually. We discuss each benchmark in a separate section where the percentage deviation between obtained solution (obt) and compared solution (com) is shown. Here, the percentage deviation is calculated as $(\text{obt} - \text{com})/\text{com}$. As ICW is a probabilistic algorithm, results can vary from run to run.

Summary results for CVRP instances

The results in Table 3 show that ICW can find high quality solutions in reasonable computation times. Out of the 84 problems, we find the optimal solutions for 68 problems with up to 135 customers. For sixteen medium problems with 50–100 customers, the percentage deviations between our solutions and the

Table 2 The algorithms used to compare with ICW.

Abbreviation	Authors	Year	Algorithm
CW	Clarke and Wright [11]	1964	Clarke and Wright savings algorithm
NMP	Noon et al [57]	1994	TSSP+1 decomposition approach
XK	Xu and Kelly [58]	1996	Network flow-based tabu search heuristic
ML	Mazzeo and Loiseau [59]	2004	Ant colony algorithm
AÖ	Altınel and Öncan [43]	2005	New enhancement of the Clarke and Wright savings heuristic
CYW	Chen et al [60]	2006	Hybrid discrete particle swarm optimization algorithm
GN	Ganesh and Narendran [61]	2007	Cluster-and-search heuristic
AK	Ai and Kachitvichyanukul [33]	2009	Particle swarm optimization and two solution representations
BHK	Bouhafs et al [62]	2010	Hybrid heuristic approach
GPV	Geetha et al [63]	2010	Hybrid particle swarm optimization with genetic operators
GGW	Groër et al [49]	2010	Library of local search heuristic
H	Hinton [64]	2010	Novel techniques
KS	Kim and Son [65]	2010	Probability matrix based particle swarm optimization
NJK	Na et al [66]	2011	Extended sweep algorithm

optimal solutions are very low (0.78% for A-n62-k8, 1.00% for A-n64-k9, 0.51% for A-n80-k10, 0.54% for B-n52-k7, 0.30% for B-n66-k9, 0.71% for B-n68-k9, 1.39% for B-n78-k10, 0.43% for P-n55-k10, 0.68% for P-n76-k4, 0.44% for P-n101-k4, 0.59% for E-n76-k7, 0.95% for E-n76-k8, 1.08% for E-n76-k10, 0.59% for E-n76-k14, 0.49% for E-n101-k8, 1.21% for E-n101-k14). Nevertheless, in those near optimal solutions, there are five problems (A-n80-k10, B-n66-k9, P-n55-k10, E-n76-k10, E-n101-k8) for which our results are better than the other results. Moreover, there are also fourteen problems where our results not only obtained the optimal solutions but also performed better than the other results. Furthermore, our solutions outperform CW and ECW solutions in all directions. These indicate that ICW is extremely effective and efficient to produce high quality solutions for five well-known datasets of CVRP. The important details of our improvement are as follows.

The average percentage deviation between CW solutions and our solutions for benchmark of datasets A, B, P, E, and F are −4.7%, −2.3%, −10.4%, −5.6%, and −2.9%. We have found that CW solutions were improved by the average of 5.5%. This finding shows that dataset P has the highest average deviation and dataset B has the lowest average devia-

Table 3 Comparisons of five well-known datasets with integer results.

Instance	Optimal solution	CW	NMP	XK	ML	AÖ	CYW	GN	AK	BHK	GPV	GGW	H	KS	NJK	ICW	
																Solution	Time (s)
A-n32-k5	784	842	—	—	—	827	—	**784**	—	—	—	—	787	—	810	**784**	1.261
A-n33-k5	661	713	—	—	—	700	**661**	**661**	**661**	—	**661**	—	662	**661**	686	**661**	1.891
A-n33-k6	742	(775)	—	—	—	743	—	**742**	—	—	—	—	**742**	—	743	**742**	1.732
A-n34-k5	778	(810)	—	—	—	793	—	**778**	—	—	—	—	780	—	785	**778**	2.980
A-n36-k5	799	826	—	—	—	806	—	**799**	—	—	—	—	802	—	826	**799**	1.908
A-n37-k5	669	705	—	—	—	708	—	**669**	—	—	—	—	672	—	670	**669**	3.469
A-n37-k6	949	975	—	—	—	974	—	**949**	—	—	—	—	951	—	962	**949**	1.850
A-n38-k5	730	(765)	—	—	—	(751)	—	**730**	—	—	—	—	733	—	749	**730**	5.647
A-n39-k5	822	898	—	—	—	894	—	**822**	—	—	—	—	828	—	—	**822**	2.744
A-n39-k6	831	861	—	—	—	848	—	**831**	—	—	—	—	835	—	856	**831**	2.462
A-n44-k6	937	(974)	—	—	—	(985)	—	**937**	—	—	—	941	938	—	957	**937**	26.148
A-n45-k6	944	(1005)	—	—	—	955	—	**944**	—	—	—	948	**944**	—	991	**944**	16.501
A-n45-k7	1146	1200	—	—	—	1178	—	**1146**	—	—	—	—	**1146**	—	1173	**1146**	12.702
A-n46-k7	914	940	—	—	—	934	**914**	**914**	**914**	—	921	**914**	917	**914**	946	**914**	8.515
A-n48-k7	1073	1110	—	—	—	1102	—	**1073**	—	—	—	**1073**	1074	—	1113	**1073**	29.267
A-n53-k7	1010	1098	—	—	—	1062	—	1017	—	—	—	**1010**	1020	—	—	**1010**	34.585
A-n54-k7	1167	1199	—	—	—	1174	—	1172	—	—	—	—	1171	—	—	***1167**	6.698
A-n55-k9	1073	1098	—	—	—	1102	—	**1073**	—	—	—	—	1074	—	1095	**1073**	26.444
A-n60-k9	1354	1416	—	—	—	1372	**1354**	1358	1355	—	1368	—	1367	**1354**	1420	**1354**	68.984
A-n61-k9	1034	(1099)	—	—	—	(1045)	—	1038	—	—	—	—	1040	—	1100	***1034**	50.631
A-n62-k8	1288	1346	—	—	—	1341	—	**1288**	—	—	—	—	1316	—	1359	1298	14.495
A-n63-k9	1616	(1684)	—	—	—	1648	—	1627	—	—	—	—	1636	—	1712	***1616**	12.579
A-n63-k10	1314	1352	—	—	—	1356	—	1322	—	—	—	—	1322	—	1386	***1314**	39.647
A-n64-k9	1401	(1489)	—	—	—	1441	—	1410	—	—	—	—	1424	—	1499	1415	41.513
A-n65-k9	1174	(1230)	—	—	—	1194	—	1177	—	—	—	—	1189	—	1223	***1174**	14.670
A-n69-k9	1159	1206	—	—	—	1179	—	1163	—	—	—	—	1174	—	1207	***1159**	95.858
A-n80-k10	1763	1859	—	—	—	1810	—	1780	—	—	—	—	1794	—	1866	*1772	97.353

Table 3 (Cont.).

Instance	Optimal solution	CW	NMP	XK	ML	AÖ	CYW	GN	AK	BHK	GPV	GGW	H	KS	NJK	ICW	
																Solution	Time (s)
B-n31-k5	672	677	—	—	—	673	—	**672**	—	—	—	—	676	—	677	**672**	1.267
B-n34-k5	788	794	—	—	—	788	—	**788**	—	—	—	—	789	—	802	**788**	2.393
B-n35-k5	955	978	—	—	—	975	**955**	**955**	**955**	—	**955**	—	956	**955**	962	**955**	3.076
B-n38-k6	805	837	—	—	—	820	—	**805**	—	—	—	—	807	—	817	**805**	10.180
B-n39-k5	549	564	—	—	—	552	—	**549**	—	—	—	—	553	—	575	**549**	2.685
B-n41-k6	829	(896)	—	—	—	869	—	**829**	—	—	—	839	834	—	843	**829**	9.615
B-n43-k6	742	777	—	—	—	752	—	**742**	—	—	—	**742**	746	—	746	**742**	6.544
B-n44-k7	909	936	—	—	—	932	—	**909**	—	—	—	**909**	914	—	942	**909**	3.182
B-n45-k5	751	754	—	—	—	751	**751**	**751**	**751**	—	754	—	753	**751**	797	**751**	5.207
B-n45-k6	678	(723)	—	—	—	(742)	—	**678**	—	—	—	—	681	—	732	**678**	22.088
B-n50-k7	741	745	—	—	—	746	—	**741**	—	—	—	**741**	744	—	779	**741**	3.605
B-n50-k8	1312	1356	—	—	—	1381	—	1318	—	—	—	—	1317	—	1349	***1312**	16.914
B-n52-k7	747	761	—	—	—	754	—	**747**	—	—	—	**747**	749	—	758	751	19.436
B-n56-k7	707	727	—	—	—	718	—	710	—	—	—	**707**	712	—	726	**707**	9.141
B-n57-k9	1598	1652	—	—	—	1616	—	1599	—	—	—	—	1605	—	1642	***1598**	16.936
B-n64-k9	861	(915)	—	—	—	(924)	—	864	—	—	—	884	870	—	1161	***861**	26.186
B-n66-k9	1316	(1419)	—	—	—	1354	—	—	—	—	—	—	1328	—	1363	*1320	71.090
B-n67-k10	1032	(1095)	—	—	—	1107	—	1037	—	—	—	—	1041	—	1080	***1032**	56.309
B-n68-k9	1272	1311	—	—	—	1309	**1272**	1275	1274	—	1281	—	1287	1275	1308	1281	22.333
B-n78-k10	1221	1259	—	—	—	1255	1239	1260	1223	—	—	—	1227	1223	1268	1238	162.953
P-n16-k8	450	(478)	—	—	—	(472)	—	**450**	—	—	—	—	451	—	513	**450**	0.544
P-n19-k2	212	237	—	—	—	(219)	—	**212**	—	—	—	—	**212**	—	219	**212**	0.780
P-n20-k2	216	234	—	—	—	247	—	**216**	—	218	—	—	217	—	217	**216**	0.655
P-n21-k2	211	236	—	—	—	233	—	**211**	—	—	—	—	212	—	**211**	**211**	0.665
P-n22-k2	216	240	—	—	—	234	—	**216**	—	—	—	—	217	—	**216**	**216**	0.714
P-n22-k8	603	(591)	—	—	—	(590)	—	**603**	—	—	—	—	(588)	—	(560)	**603**	5.417
P-n23-k8	529	(537)	—	—	—	(537)	—	**529**	—	—	—	—	531	—	554	**529**	8.450
P-n40-k5	458	516	—	—	—	484	—	**458**	—	—	—	—	461	—	467	**458**	2.425
P-n45-k5	510	569	—	—	—	519	—	**510**	—	**510**	—	—	512	—	—	**510**	18.135
P-n50-k7	554	593	—	—	—	578	—	**554**	—	—	—	—	559	—	—	**554**	4.500
P-n50-k8	631	(670)	—	—	—	(644)	—	643	—	—	—	—	637	—	—	***631**	13.196
P-n50-k10	696	(735)	—	—	—	(708)	—	**696**	—	—	—	—	700	—	—	**696**	7.392
P-n51-k10	741	(786)	—	—	—	754	—	**741**	—	743	—	—	749	—	—	**741**	21.023
P-n55-k7	568	617	—	—	—	586	—	**568**	—	—	—	—	573	—	—	**568**	10.422
P-n55-k8	576	628	—	—	—	589	—	—	—	—	—	—	580	—	—	***576**	8.646
P-n55-k10	694	(734)	—	—	—	709	—	698	—	—	—	—	699	—	—	*697	7.940
P-n60-k10	744	814	—	—	—	799	—	**744**	—	—	—	—	752	—	—	**744**	46.106
P-n60-k15	968	(1013)	—	—	—	1003	—	**968**	—	—	—	—	980	—	—	**968**	10.809
P-n65-k10	792	848	—	—	—	—	—	800	—	—	—	—	805	—	—	***792**	77.789
P-n70-k10	827	(892)	—	—	—	848	—	**827**	—	—	—	—	845	—	—	**827**	61.272
P-n76-k4	593	684	—	—	—	(638)	602	**593**	594	—	610	—	608	594	612	597	75.559
P-n76-k5	627	705	—	—	—	678	—	**627**	—	—	—	—	639	—	—	**627**	17.570
P-n101-k4	681	754	—	—	—	708	694	687	683	—	—	—	697	683	715	684	98.591
E-n22-k4	375	388	—	—	—	**375**	—	**375**	—	—	—	—	**375**	—	**375**	**375**	0.474
E-n23-k3	569	621	—	—	—	574	—	**569**	—	—	—	—	(568)	—	**569**	**569**	0.501
E-n30-k3	534	(532)	—	—	—	—	**534**	**534**	**534**	—	**534**	—	(505)	**534**	543	**534**	5.726
E-n33-k4	835	841	—	—	—	841	—	**835**	—	—	—	—	837	—	852	**835**	1.603
E-n51-k5	521	(582)	—	—	**521**	—	528	**521**	**521**	—	522	**521**	524	**521**	532	**521**	9.931
E-n76-k7	682	733	690	685	—	703	688	690	**682**	—	—	—	696	687	703	686	53.178
E-n76-k8	735	787	739	736	—	772	—	738	—	—	—	—	743	—	746	742	87.349
E-n76-k10	830	903	—	—	877	(854)	—	867	—	—	—	—	846	—	907	*839	70.115
E-n76-k14	1021	(1048)	1042	1023	—	(1038)	—	1032	—	—	—	—	1034	—	1072	1027	50.207
E-n101-k8	817	879	—	—	845	853	—	830	—	—	—	—	831	—	850	*821	198.335
E-n101-k14	1071	1130	—	1080	—	1120	—	1099	—	—	—	—	1102	—	1152	1084	135.074
F-n45-k4	721	737	—	—	—	—	—	—	—	—	—	—	723	—	750	***721**	3.856
F-n72-k4	237	(253)	—	—	—	—	244	—	**237**	—	253	—	241	**237**	241	**237**	75.613
F-n135-k7	1162	1201	—	—	—	—	1215	—	**1162**	—	—	—	1169	1170	1221	**1162**	325.277

Bold indicates the optimal solution was obtained.

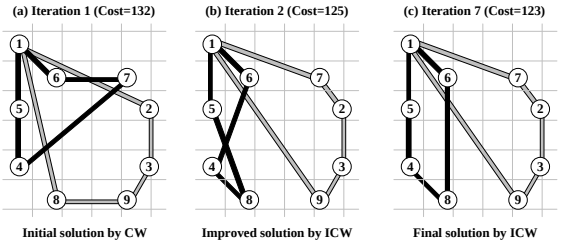
* indicates the obtained solution beats the others.

(number) indicates the infeasible solution (the number of available vehicles is inadequate).

tion. The greatest improvements of these values are, respectively, presented in top three instances including P-n76-k4 (−14.6%), P-n40-k5 (−12.7%), and P-n76-k5 (−12.4%). We can conclude that the problems which have the features like clustered customers can be solved by CW better than the problems which have the features of uniformly and not uniformly dispersed customers. In addition, the results show that ICW can significantly solve both above-mentioned problems to obtain optimal or near optimal solutions.

Another finding from Table 3 is the infeasible solutions produced by CW and ECW in which the number of available vehicles is inadequate. This finding is referred to Vigo[39] that CW does not allow for the control of the number of routes of final solution. The solution found for a given instance can, in fact, require more than k routes to serve all the customers, hence being infeasible.

Fig. 3 The example of our approach works.



The advantages and performance analysis of ICW

This paper successfully demonstrates the clear advantages of ICW which can improve the most recent CW solutions (Gaskell[40], Yellow[41], Paessens[42], Altınel and Öncan[43], Battarra et al[44], Corominas et al[45]) for five well-known datasets of CVRP. Moreover, we observe that ICW also obtains optimal solutions in some instances. We attribute this success to two key factors. First, the savings list ranking based on tournament and roulette wheel selections provides a powerful classification mechanism. Second, the new savings list replacement, when the solution obtained from current savings list is better than the previous one, provides an effective enhancement to locate a new better solution. In order to illustrate the works of our algorithm, we give an example in Fig. 3 which is a case of eight customers with two vehicles and each vehicle has a capacity of four units. ICW can start with any feasible or infeasible solution. We use CW to produce an initial solution as shown in Fig. 3a, and then our approach inside the ICW mechanism is executed to improve the solution which occurs in two cases composed of intra- and inter-route neighbourhoods as shown in Fig. 3b and 3c. In particular, some vehicles in 28 problems have more than ten customers as follows: 11 customers (A-n37-k5, A-n38-k5, A-n53-k7, B-n39-k5, B-n50-k7, B-n52-k7, B-n56-k7, B-n66-k9, B-n68-k9, B-n78-k10, P-n21-k2, P-n22-k2, P-n45-k5, P-n55-k7, E-n51-k5), 13 customers (A-n64-k9), 14 customers (A-n62-k8, A-n80-k10, E-n23-k3, E-n30-k3, E-n33-k4, E-n76-k7), 16 customers (P-n76-k5), 19 customers (F-n45-k4), 22 customers (P-n76-k4), 24 customers (F-n72-k4), 29 customers (P-n101-k4), and 41 customers (F-n135-k7). It is shown that our algorithm exhaustively explored the solution spaces, and can obtain the optimal or near optimal solutions alone without any local search method like 2-opt, 3-opt, and others.

In Table 4, we have reported the experiment of our results by ICW in every 2500-iteration that influences our approach types of behaviour. In our algorithm, each iteration is finished in just a few milliseconds. As expected, the largest number of improvements is found at the beginning (1–2500 iterations). In some

Table 4 The experiment of our results by ICW.

Instance	2.5 K	5 K	7.5 K	10 K	F	L
A-n32-k5	3	0	0	0	2	13
A-n33-k5	4	0	0	0	4	31
A-n33-k6	6	0	0	0	2	170
A-n34-k5	9	0	0	0	2	799
A-n36-k5	3	0	0	0	28	169
A-n37-k5	5	0	0	0	12	1053
A-n37-k6	1	0	0	0	7	7
A-n38-k5	8	0	0	0	18	2081
A-n39-k5	7	0	0	0	2	374
A-n39-k6	8	0	0	0	2	270
A-n44-k6	4	2	3	1	4	9490
A-n45-k6	14	1	0	0	101	4820
A-n45-k7	11	2	0	0	3	3360
A-n46-k7	5	0	0	0	87	1878
A-n48-k7	5	1	5	1	17	7602
A-n53-k7	9	2	1	0	2	6787
A-n54-k7	8	0	0	0	2	392
A-n55-k9	4	6	0	0	97	4441
A-n60-k9	7	2	4	2	3	9522
A-n61-k9	10	1	2	0	3	5647
A-n62-k8	11	0	0	0	7	909
A-n63-k9	8	7	0	0	2	4045
A-n63-k10	5	0	0	0	3	630
A-n64-k9	8	8	0	0	4	3995
A-n65-k9	18	0	0	0	10	815
A-n69-k9	12	1	1	2	23	8899
A-n80-k10	15	6	0	0	2	4464
B-n31-k5	1	0	0	0	29	29
B-n34-k5	2	0	0	0	361	643
B-n35-k5	6	0	0	0	6	1038
B-n38-k6	5	11	0	0	10	4475
B-n39-k5	5	0	0	0	26	339
B-n41-k6	10	4	0	0	4	3186
B-n43-k6	6	0	0	0	2	1514
B-n44-k7	4	0	0	0	5	161
B-n45-k5	4	0	0	0	96	925
B-n45-k6	5	5	0	0	37	4961
B-n50-k7	3	0	0	0	6	35
B-n50-k8	12	1	0	0	40	3305
B-n52-k7	9	2	0	0	4	3644
B-n56-k7	4	0	0	0	25	722
B-n57-k9	14	0	0	0	14	1966
B-n64-k9	14	0	0	0	4	490
B-n66-k9	15	6	1	0	2	6778
B-n67-k10	8	1	0	0	4	4966
B-n68-k9	18	0	0	0	10	1151
B-n78-k10	7	0	5	5	85	9547

Table 4 (Cont.).

Instance	2.5 K	5 K	7.5 K	10 K	F	L
P-n16-k8	2	0	0	0	6	7
P-n19-k2	4	0	0	0	4	224
P-n20-k2	2	0	0	0	10	16
P-n21-k2	2	0	0	0	4	8
P-n22-k2	5	0	0	0	5	31
P-n22-k8	11	0	0	0	2	12
P-n23-k8	4	0	0	0	17	72
P-n40-k5	7	0	0	0	2	198
P-n45-k5	10	1	1	0	2	5890
P-n50-k7	6	0	0	0	5	285
P-n50-k8	9	0	0	0	3	1866
P-n50-k10	8	0	0	0	2	986
P-n51-k10	8	5	0	0	8	4456
P-n55-k7	13	0	0	0	2	1278
P-n55-k8	7	0	0	0	4	878
P-n55-k10	10	0	0	0	6	701
P-n60-k10	15	4	1	0	2	6572
P-n60-k15	13	0	0	0	2	745
P-n65-k10	9	4	1	2	3	9444
P-n70-k10	9	4	1	0	2	5131
P-n76-k4	8	3	0	0	15	4822
P-n76-k5	10	0	0	0	5	343
P-n101-k4	14	1	0	0	3	3181
E-n22-k4	1	0	0	0	2	2
E-n23-k3	1	0	0	0	2	2
E-n30-k3	20	0	0	0	3	1946
E-n33-k4	4	0	0	0	21	803
E-n51-k5	13	2	0	0	3	2988
E-n76-k7	10	2	1	0	22	5104
E-n76-k8	11	2	3	2	13	9185
E-n76-k10	15	7	3	0	2	6961
E-n76-k14	8	1	0	0	97	4102
E-n101-k8	13	3	1	0	10	7467
E-n101-k14	9	3	0	0	9	4708
F-n45-k4	9	0	0	0	3	401
F-n72-k4	6	2	2	0	110	7269
F-n135-k7	17	2	7	1	5	9833

2.5 K: Number of improvements between 1 and 2500 iterations. 5 K: Number of improvements between 2501 and 5000 iterations. 7.5 K: Number of improvements between 5001 and 7500 iterations. 10 K: Number of improvements between 7501 and 10 000 iterations.
F: Iteration number of the first improvement.
L: Iteration number of the last improvement.

problems, improvements are obtained up to 10 000 iterations. Moreover, ICW can provide optimal solution in just some 10 iterations according to E-n22-k4, E-n23-k3, A-n37-k6, P-n16-k8, P-n21-k2, P-n22-k8, A-n32-k5, P-n20-k2, B-n31-k5, A-n33-k5, P-n22-k2, B-n50-k7, and P-n23-k8 problems (2, 2, 7, 7, 8, 12, 13, 16, 29, 31, 31, 35, and 72 iterations, respectively), which takes only a few seconds to run.

CONCLUSIONS

In this paper, we presented a new approach called the improved Clarke and Wright savings algorithm (ICW) to solve the capacitated vehicle routing problem (CVRP). It combines the Clarke and Wright savings algorithm with tournament and roulette wheel selection operators. We also performed experiments using five well-known datasets of CVRP (composed of 84 instances) obtained from literature and compared them with the optimal solutions. Our algorithm is competitive or outperforms recent heuristics and metaheuristics on integer distance. Moreover, the numerical results show that our algorithm also gets small average percentage deviations when compared with the optimal one (0.14%). Furthermore, our solutions outperform Clarke and Wright[11]'s solutions and Altınel and Öncan[43]'s solution in all directions and generate the optimal solutions in 81% of all instances (68 out of 84).

Acknowledgements: The first author expresses gratitude to Panyapiwat Institute of Management, Thailand for providing the financial support for his Ph.D. study in Sirindhorn International Institute of Technology, Thammasat University, Thailand.

REFERENCES

1. Dantzig GB, Ramser JH (1959) The truck dispatching problem. *Manag Sci* **6**, 80–91.
2. Miller RE, Thatcher JW (1972) *Complexity of Computer Computations*, Plenum Press, New York.
3. Garey MR, Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, New York.
4. Lenstra JK, Rinnooy KA (1981) Complexity of vehicle routing and scheduling problems. *Networks* **11**, 221–7.
5. Toth P, Vigo D (2002) *The Vehicle Routing Problem*, SIAM monographs on discrete mathematics and applications, SIAM Publishing, Philadelphia, PA.
6. Christofides N, Mingozzi A, Toth P (1981) Exact algorithm for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Math Program* **20**, 255–82.
7. Fisher ML (1994) Optimal solution of vehicle routing problems using minimum K-trees. *Oper Res* **42**, 626–42.
8. Augerat P, Belenguer J, Benavent E, Corbern A, Naddef D, Rinaldi G (1995) Computational results with a branch and cut code for the capacitated vehicle routing problem, Research Report 949-M, Universite Joseph Fourier, Grenoble, France.

9. Lysgaard J, Letchford AN, Eglese RW (2004) A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Math Program* **100**, 423–45.
10. Fukasawa R, Longo H, Lysgaard J, Poggi de Aragão M, Reis M, Uchoa E, Werneck RF (2006) Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Math Program* **106**, 491–511.
11. Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Oper Res* **12**, 568–81.
12. Wren A, Holliday A (1972) Computer scheduling of vehicles form one or more depots to a number of delivery points. *J Oper Res Soc* **23**, 333–44.
13. Gillett BE, Miller LR (1974) A heuristic algorithm for the vehicle-dispatch problem. *Oper Res* **22**, 340–9.
14. Mole RH, Jameson SR (1976) A sequential route-building algorithm employing a generalized savings criterion. *J Oper Res Soc* **27**, 503–11.
15. Foster BA, Ryan DM (1976) An integer programming approach to the vehicle scheduling problem. *J Oper Res Soc* **27**, 367–84.
16. Ryan DM, Hjorring C, Glover F (1993) Extensions of the petal method for vehicle routing. *J Oper Res Soc* **44**, 289–96.
17. 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 315–38.
18. Fisher ML, Jaikumar R (1981) A generalized assignment heuristic for vehicle routing. *Networks* **11**, 109–24.
19. Renaud J, Boctor FF, Laporte G (1996) An improved petal heuristic for the vehicle routing problem. *J Oper Res Soc* **47**, 329–36.
20. Lin S (1965) Computer solutions of the travelling salesman problem. *Bell Syst Tech J* **44**, 2245–69.
21. Or I (1976) Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking. PhD thesis, Department of Industrial Engineering and Management Science, North-western Univ, Evanston, IL.
22. Osman IH (1993) Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Ann Oper Res* **41**, 421–51.
23. Gendreau M, Hertz A, Laporte G (1994) A tabu search heuristic for the vehicle routing problem. *Manag Sci* **40**, 1276–90.
24. Rochat Y, Taillard ÉD (1995) Probabilistic diversification and intensification in local search for vehicle routing. *J Heuristics* **1**, 147–67.
25. Bean JC (1994) Genetic algorithms and random keys for the sequencing and optimization. *Informs J Comput* **6**, 154–60.
26. Baker BM, Ayechew MA (2003) A genetic algorithm for the vehicle routing problem. *Comput Oper Res* **30**, 787–800.
27. Dorigo M, Di Caro G, Gambardella LM (1999) Ant algorithms for discrete optimization. *Artif Life* **5**, 137–72.
28. Bin Y, Zhong-Zhen Y, Baozhen Y (2009) An improved ant colony optimization for vehicle routing problem. *Eur J Oper Res* **196**, 171–6.
29. Prins C (2004) A simple and effective evolutionary algorithm for the vehicle routing problem. *Comput Oper Res* **31**, 1985–2002.
30. Moscato P, Cotta C (2003) A gentle introduction to memetic algorithms. In: Glover F, Kochen-berger GA (eds) *Handbook of Metaheuristics*. Kluwer Academic, Boston, pp 105–44.
31. Mester D, Bräysy O (2007) Active-guided evolution strategies for large-scale capacitated vehicle routing problems. *Comput Oper Res* **34**, 2964–75.
32. Marinakis Y, Marinaki M, Dounias G (2008) Honey bees mating optimization algorithm for the vehicle routing problem. *Nat Comput* **129**, 5–27.
33. Ai TJ, Kachitvichyanukul V (2009) Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem. *Comput Ind Eng* **56**, 380–7.
34. Marinakis Y, Marinaki M (2010) A hybrid genetic - particle swarm optimization algorithm for the vehicle routing problem. *Expert Syst Appl* **37**, 1446–55.
35. Tillman FA (1969) The multiple terminal delivery problem with probabilistic demands. *Transport Sci* **3**, 192–204.
36. Raff S (1983) Routing and scheduling of vehicles and crews: The state of the art. *Comput Oper Res* **10**, 63–211.
37. Goetschalckx M, Jacobs-Blecha C (1989) The vehicle routing problem with backhauls. *Eur J Oper Res* **42**, 39–51.
38. Deif I, Bodin LD (1984) Extension of the Clarke-Wright algorithm for solving the vehicle routing problem with backhauling. In: Kidder AE (ed) Proceedings of the Babson Conference on Software Uses in Transportation and Logistics Management, Babson Park, MA, pp 75–96.
39. Vigo D (1996) A heuristic algorithm for the asymmetric capacitated vehicle routing problem. *Eur J Oper Res* **89**, 108–26.
40. Gaskell TJ (1967) Bases for vehicle fleet scheduling. *J Oper Res Soc* **18**, 281–95.
41. Yellow P (1970) A computational modification to the savings method of vehicle scheduling. *J Oper Res Soc* **21**, 281–3.
42. Paessens H (1988) The savings algorithm for the vehicle routing problem. *Eur J Oper Res* **34**, 336–44.
43. Altınel İK, Öncan T (2005) A new enhancement of the Clarke and Wright savings heuristic for the capacitated vehicle routing problem. *J Oper Res Soc* **56**, 954–61.
44. Battarra M, Golden B, Vigo D (2008) Tuning a parametric Clarke-Wright heuristic via a genetic algorithm. *J Oper Res Soc* **59**, 1568–72.
45. Corominas A, García-Villoria A, Pastor R (2010) Fine-tuning a parametric Clarke and Wright heuristic by

means of EAGH (empirically adjusted greedy heuristics). *J Oper Res Soc* **61**, 1309–14.
46. Juan AA, Faulin J, Ruiz R, Barrios B, Caballé S (2010) The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem. *Appl Soft Comput* **10**, 215–24.
47. Laporte G, Gendreau M, Potvin J, Semet F (2000) Classical and modern heuristics for the vehicle routing problem. *Int Trans Oper Res* **7**, 285–300.
48. Chen P, Huang H, Dong X (2010) Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem. *Expert Syst Appl* **37**, 1620–7.
49. Groër C, Golden B, Wasil E (2010) A library of local search heuristics for the vehicle routing problem. *Math Prog Comp* **2**, 79–101.
50. Pichpibul T, Kawtummachai R (2008) Delivery routing optimization for an inbound-outbound logistic case. In: Proceedings of the 9th International Conference on Industrial Management, pp 286–92.
51. Pichpibul T, Kawtummachai R (2009) An implementation of a two-phase metaheuristic approach for the multi-depot vehicle routing optimization with simultaneous delivery and pickup. In: Proceedings of the International Symposium on Scheduling, pp 30–5.
52. Goldberg D, Korb B, Deb K (1989) Messy genetic algorithms: motivation, analysis, and first results. *Comp Systs* **3**, 493–530.
53. Holland J (1975) *Adaptation in Natural and Artificial Systems*, Univ of Michigan Press, Ann Arbor.
54. Christofides N, Eilon S (1969) An algorithm for the vehicle dispatching problem. *J Oper Res Soc* **20**, 309–18.
55. Baldacci R, Christofides N, Mingozzi A (2008) An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Math Program* **115**, 351–85.
56. Reinelt G (1991) TSPLIB: A travelling salesman problem library. *ORSA J Comput* **3**, 376–84.
57. Noon CE, Mittenthal J, Pillai R (1994) A TSSP+1 decomposition approach for the capacity-constrained vehicle routing problem. *Eur J Oper Res* **79**, 524–36.
58. Xu J, Kelly JP (1996) A network flow–based tabu search heuristic for the vehicle routing problem. *Transport Sci* **30**, 379–93.
59. Mazzeo S, Loiseau I (2004) An ant colony algorithm for the capacitated vehicle routing. *Electron Notes Discrete Math* **18**, 181–6.
60. Chen A, Yang G, Wu Z (2006) Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. *J Zhejiang Univ Sci A* **7**, 607–14.
61. Ganesh K, Narendran TT (2007) CLOVES: A cluster-and-search heuristic to solve the vehicle routing problem with delivery and pick-up. *Eur J Oper Res* **178**, 699–717.
62. Bouhafs L, Hajjam A, Koukam A (2010) A hybrid heuristic approach to solve the capacitated vehicle routing problem. *J Artif Intell Theor Appl* **1**, 31–4.
63. Geetha S, Poonthalir G, Vanathi PT (2010) A hybrid particle swarm optimization with genetic operators for vehicle routing problem. *J Adv Inform Tech* **1**, 181–8.
64. Hinton TG (2010) The vehicle routing problem including a range of novel techniques for its solution. PhD thesis, Department of Computer Science, Faculty of Engineering, Univ of Bristol, Bristol, UK.
65. Kim BI, Son SJ (2010) A probability matrix based particle swarm optimization for the capacitated vehicle routing problem. *J Intell Manuf* **23**, 1119–26.
66. Na B, Jun Y, Kim BI (2011) Some extensions to the sweep algorithm. *Int J Adv Manuf Tech* **56**, 1057–67.