# A Hybrid Genetic Algorithm for the Capacitated Vehicle Routing Problem

Jean Berger and Mohamed Barkaoui

Defence Research and Development Canada - Valcartier, Decision Support Technology Section
2459 Pie-XI Blvd. North, Val-Bélair, PQ, Canada, G3J 1X5
jean.berger@drdc-rddc.gc.ca

**Abstract.** Recently proved successful for variants of the vehicle routing problem (VRP) involving time windows, genetic algorithms have not yet shown to compete or challenge current best search techniques in solving the classical capacitated VRP. In this paper, a hybrid genetic algorithm to address the capacitated vehicle routing problem is proposed. The basic scheme consists in concurrently evolving two populations of solutions to minimize total traveled distance using genetic operators combining variations of key concepts inspired from routing techniques and search strategies used for a time-variant of the problem to further provide search guidance while balancing intensification and diversification. Results from a computational experiment over common benchmark problems report the proposed approach to be very competitive with the best-known methods.

## 1 Introduction

In the classical vehicle routing problem (VRP) [1], customers with known demands and service time are visited by a homogeneous fleet of vehicles with limited capacity and initially located at a central depot. Routes are assumed to start and end at the depot. The objective is to minimize total traveled distance, such that each customer is serviced exactly once (by a single vehicle), total load on any vehicle associated with a given route does not exceed vehicle capacity, and route duration combining travel and service time, is bounded to a preset limit.

A variety of algorithms including exact methods and efficient heuristics have already been proposed for VRP. For a survey on the capacitated Vehicle Routing Problem and variants see Toth and Vigo [1]. The authors present both exact and heuristic methods developed for the VRP and its main variants, focusing on issues common to VRP. Overview of classical heuristics and metaheuristics may also be found in Laporte et al. [2], and Gendreau et al. [3,4] respectively.

Tabu search techniques [5,6] and (hybrid) genetic algorithms represent some of the most efficient metaheuristics to address VRP and/or its variants. The basic idea in tabu search is to allow selection of worse solutions once a local optimum has been reached. Different memory structures are then used to prevent repeating the same solutions (cycling), and to diversify and intensify the search. Genetic algorithms [7–9]

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2723, pp. 646–656, 2003.

are adaptive heuristic search methods that mimic evolution through natural selection. They work by combining selection, recombination and mutation operations. The selection pressure drives the population toward better solutions while recombination uses genes of selected parents to produce offspring that will form the next generation. Mutation is used to escape from local minima. Hybrid genetic algorithms combine the above scheme with heuristic methods to further improve solution quality. Tabu search heuristics have proved so far the most successful technique for the capacitated VRP [2], [3], [10], [11]. Alternatively, despite its relative success reported for the traveling salesman problem (see Gendreau et al., [3]) and variants of the vehicle routing problem (VRP) involving time windows [3], [12-21], genetic algorithms have not yet shown to compete with tabu search techniques in solving the capacitated VRP. Limited work using genetic-based techniques for the classical capacitated VRP reports mitigated success so far. As recently proposed procedures match the performance of well-known classical methods [22], others fail to report comparative performance with the best well-known routing techniques, while sometime demonstrating prohibitive run-time to obtain modest solution quality [15], [23]. It is nonetheless believed that genetic-based methods targeted to the classical capacitated VRP have not yet been fully exploited.

In this paper, a competitive hybrid genetic algorithm (HGA-VRP) to address the classical capacitated vehicle routing problem is proposed for the first time. It consists in concurrently evolving two populations of solutions subject to periodic migration in order to minimize total traveled distance using genetic operators combining variations of key concepts inspired from routing techniques and search strategies used for a time-variant of the problem to further provide search guidance while balancing intensification and diversification. A computational experiment conducted on common benchmark problems shows the proposed hybrid genetic approach to be competitive with the best-published methods.

The paper is outlined as follows. Section 2 introduces the main concepts of the proposed hybrid genetic algorithm. Basic principles and features of the algorithm are first introduced. Then, the selection scheme, recombination and mutation operators are presented. Concepts derived from well-known heuristics such as large neighborhood search [24], route neighborhood-based two-stage metaheuristic [25] and $\lambda$-interchange mechanism [26] are briefly outlined. Section 3 presents the results of a computational experiment to assess the value of the proposed approach and reports a comparative performance analysis to alternate methods. Finally, some conclusions and future research directions are presented in Section 4.

## 2 Hybrid Genetic Approach

### 2.1 General Description

The proposed HGA-VRP algorithm mainly relies on the basic principles of genetic algorithms, disregarding explicit solution encoding issues for problem representation. Genetic operators are simply applied to a population of solutions rather than a population of encoded solutions (chromosomes). We refer to these solutions as solution individuals.

Emphasizing genetic diversity, our approach consists in concurrently evolving two populations of solutions ($Pop_1$, $Pop_2$) while exchanging a certain number of individuals (migration) at the end of a new generation. Exclusively formed of feasible solution individuals, populations are evolved to minimize total traveled distance using genetic operators based upon variations of known routing methods. Whenever a new best solution emerges, a post-processing procedure (RC_M) aimed at reordering customers is applied to further improve its solution quality. The RC_M mutation operator is introduced in Section 2.3. The evolutionary process is repeated until a predefined stopping condition is met. The proposed technique is significantly different from the algorithm presented by Berger and Barkaoui [14] in many respects, including the introduction of new and more efficient operators and its application to a problem variant.

The proposed steady-state genetic algorithm resorts to overlapping populations to ensure population replacement for $Pop_1$ and $Pop_2$. At first, new individuals are generated and added to population $Pop_p$ ( $p = 1, 2$ ). The process continues until the overlapping population outnumbers the initial population by $n_p$. Then, the $n_p$ worst individuals are eliminated to maintain population size using the following individual evaluation:

$$Eval_i = d_i / \max(d_m, d_i) . \tag{1}$$

where

$d_i$ = total traveled distance related to individual *i,*

$d_m$ = average total traveled distance over the individuals forming the initial populations.

The lower the evaluation value the better the individual score (minimization problem). An elitist scheme is also assumed, meaning that the best solution ever computed from a previous generation is automatically replicated and inserted as a member of the next generation.

The general algorithm is specified as follows:

*Initialization*
*Repeat*
  $p$=1
  *Repeat* {evolve population $Pop_p$ - new generation}
    *For* $j$ =1..$n_p$ *do*
      Select two parents from $Pop_p$
      Generate a new solution $S_j$ using recombination and mutation operators associated with $Pop_p$
      Add $S_j$ to $Pop_p$
    *end for*
    Remove from $Pop_p$ the $n_p$ worst individuals using the evaluation function (1)
    $p$=$p$+1
  *Until* (all populations $Pop_p$ have been visited)
  *if* (new best feasible solution) *then* apply RC_M on best solution {cust. reordering}
  Population migration {local best solutions exchange across populations}
*Until* (convergence criteria *or* max number of generations)

The initialization phase involves the generation of initial populations $Pop_1$ and $Pop_2$ using a random procedure to construct feasible solution individuals. Solutions are generated using a sequential insertion heuristic in which customers are inserted in random order at randomly chosen insertion positions within routes. This strategy is fast and simple while ensuring unbiased solution generation. Migration consists in exchanging local best individuals from one population to another. Convergence is assumed to occur either when solution quality fails to significantly improve over a consecutive number of generations or, after a maximum number of generations.

### 2.2 Selection

The selection process consists in choosing two individuals (parent solutions) within the population for mating purposes. The selection procedure is stochastic and biased toward the best solutions using a roulette-wheel scheme [9]. In this scheme, the probability to select an individual is proportional to its fitness value. Individual fitness for both populations $Pop_1$ and $Pop_2$ is computed as follows:

$$fitness_i = d_i \ . \tag{2}$$

The notation is the same as in Equation (1). Better individuals show a shorter total traveled distance (minimization problem).

### 2.3 Genetic Operators

The proposed genetic operators incorporate and combine key feature variations of efficient routing techniques such as Solomon's insertions heuristic I1 [27], large neighborhood search [24] and the route neighborhood-based two-stage metaheuristic (RNETS) [25] successfully applied for the Vehicle Routing problem with Time Windows [1]. Details on the recombination and mutation operators used are given in the next sections.

**Recombination.** A single recombination operator is considered, namely IB_X($k$). It recombines two parent solutions by removing and reinserting customers exploiting a variant of a well-known customer insertion heuristic in constructing a child solution. The insertion-based IB_X crossover operator creates an offspring by combining, one at a time, $k$ routes ($R_1$) of parent solution $P_1$ with a subset of customers, formed by nearest-neighbor routes ($R_2$) in parent solution $P_2$. The neighborhood $R_2$ includes the routes of $P_2$ whose centroid is located within a certain range of $r_1 \in R_1$ (centroid). A route centroid corresponds to a virtual site whose coordinates refer to the average position of its specific routed customers. The related range corresponds to the average distance separating $r_1$ from the routes defining $P_2$. The routes of $R_1$ are selected either randomly, with a probability proportional to the number of customers characterizing a tour or based on average distance separating consecutive customers over a route.

A stochastic removal procedure is first carried out to remove from $r_1$, customers likely to be migrated to alternate routes. Targeted customers are either selected according to waiting times, distance separating them from their immediate neighbors,

or randomly. Then, using a modified insertion heuristic inspired from Solomon [27] a feasible child tour is constructed, expanding the altered route $r_1$ by inserting customer visit candidates derived from the nearest-neighbor routes $R_2$ defined earlier. The proposed insertion technique consists in adding a stochastic feature to the standard customer insertion heuristic I1 [27], by selecting randomly the next customer visit over the three best candidates with a bias toward the best. Once the construction of the child route is completed, and reinsertion is no longer possible, a new route construction cycle is initiated. The overall process is repeated for the $k$ routes of $R_1$. Finally, the child inherits the remaining "diminished" routes (if any) of $P_1$. If unvisited customers still remain, additional routes are built using a nearest-neighbor procedure.

The whole process is then iterated once more to generate a second child by interchanging the roles of $P_1$ and $P_2$. Further details of the operator may be found in Berger and Barkaoui [14].

**Mutation.** A suite of four mutation operators is proposed, namely LNSB_M($d$), EE_M, IEE_M and RC_M($I$). Each mutator is briefly described next.

The LNSB_M ($d$) (large neighborhood search -based) mutation operator relies on the concept of the Large Neighborhood Search (LNS) method proposed by Shaw [24]. The LNS consists in exploring the search space by repeatedly removing related customers and reinserting them using constraint-based tree search (constraint programming). Customer relatedness defines a relationship linking two customers based upon specific properties (e.g. proximity and/or identical route membership), such that when both customers are considered simultaneously for a visit, they can compete with each other for reinsertion creating new opportunities for solution improvement. Therefore, customers close to one another naturally offer interchange opportunities to improve solution quality. Similarly, solution number of tours is more likely to decrease when customers sharing route membership are removed all together. As stated in Shaw [24], a set of related customers is first removed. The reinsertion phase is then initiated.

The proposed customer reinsertion technique differs from the procedure introduced by Shaw [24] resorting to alternate insertion cost functions and, customer visit ordering schemes (variable ordering scheme) to carry out large neighborhood search. Customer visit ordering determines the effective sequence of customers to be consecutively visited while exploring the solution space (search tree expansion). For diversification purposes, two customer reinsertion methods are proposed, one of them being randomly selected (50% probability) on mutator invocation.

The first reinsertion method relies on the insertion cost function prescribed by Solomon's procedure I1 [27] for the VRP with time windows and, a rank-based customer visit ordering scheme. Customer insertion cost is defined by the sum of key contributions referring respectively to traveled distance increase, and delayed service time. As for customer ordering, customers ($\{c\}$) are sorted (*CustOrd*) according to a composite ranking, departing from the myopic scheme originally proposed by Shaw. The ranking is defined as an additive combination of two separate rankings, previously achieved over best insertion costs ($Rank_{Cost}(c)$) on the one hand, and number of feasible insertion positions ($Rank_{|Posl}(c)$) on the other hand:

$$CustOrd \leftarrow Sort \ \left( Rank_{Cost}(c) + Rank_{|Pos|}(c) \right). \tag{3}$$

The smaller the insertion cost (short total distance, traveled time) and the number of positions (opportunities), the better (smaller) the ranking. The next customer to be visited within the search process is selected according to the following expression:

$$customer \leftarrow CustOrd[INTEGER(L \times rand^{D})]. \tag{4}$$

where

$L$ = current number of customers to be inserted,

$rand$ = real number over the interval [0,1] (uniform random number generator),

$D$ = parameter controlling determinism. If $D$=1 then selection is purely random (default: $D$=15).

Customer position selection (value ordering) is then based on insertion cost minimization.

The second reinsertion method involves features of the successful insertion heuristic proposed by Liu and Shen [25], for the VRP with time windows, exploiting the maximization of a regret insertion cost function which concurrently takes into account multiple insertion opportunities (regret cost), to determine customer visit ordering. The regret cost -based customer visit ordering scheme is specified as follows. In the insertion procedure proposed by Liu and Shen [25], route neighborhoods associated to unvisited customers are repeatedly examined for customer insertion. This new route-neighborhood structure relates one or multiple routes to individual customers. In our approach the route neighborhood which differs from the one reported by Liu and Shen [25], is strictly bounded to two tours, comprising routes whose distance separating their centroid from the customer location is minimal. Each feasible customer insertion opportunity is explored over its entire route neighborhood. The next customer visit is selected by maximizing a so-called regret cost function that accounts for multiple route insertion opportunities:

$$\text{Regret Cost} = \sum_{r \in RN(c)} \{C_c(r) - C_c(r^*)\} \tag{5}$$

where

$RN(c)$ = route neighborhood of customer $c$,

$C_c(r)$ = minimum insertion cost of customer $c$ within route $r$ (see [25]),

$C_c(r^*)$ = minimum insertion cost of customer $c$ over its route neighborhood.

For both reinsertion methods, once a customer is selected, search is carried out over its different insertion positions (value ordering) based on insertion cost minimization, exploiting limited discrepancy search [28] as specified in Shaw [24]. However, search tree expansion is achieved using a non-constant discrepancy factor $d$, selected randomly (uniform probability distribution) over the set {1,2}. Remaining unvisited customers (if any) are then inserted in additional routes.

The EE_M (edge exchange) mutator focuses on inter-route improvement. EE_M attempts to shift customers to alternate routes as well as to exchange sets of customers between two routes. It is inspired from the $\lambda$-interchange mechanism of Osman [26], performing reinsertions of customer sets over two neighboring routes. In the proposed

mutation procedure, each customer is explored for reinsertion in its surrounding route neighborhood made up of two tours. Tours are being selected such that the distance separating their centroid from customer location is minimal. Customer exchanges occur as soon as the solution improves, i.e., we use a "first admissible" improving solution strategy. Assuming the notation (x, y) to describe the different sizes of customer sets to be exchanged over two routes, the current operator explores values running over the range (x=1, y=0,1,2). The IEE_M (intra-route edge exchange) mutation operator is similar to EE_M except that customer migration is restricted to the same route.

The RC_M (*I*) (reorder customers) mutation operator is an intensification procedure intended to reduce total traveled distance of feasible solutions by reordering customers within a route. The procedure consists in repeatedly reconstructing a new tour using the sequential insertion heuristic I1 over *I* different sets (e.g. *I*=20) of randomly generated parameter values, returning the best solution generated shall an improved one emerge.

## 3 Computational Results

A computational experiment has been conducted to compare the performance of the proposed algorithm with some of the best techniques designed for VRP. The algorithm has been tested on the well-known VRP benchmark proposed by Christofides et al. [29]. For these instances, travel time separating two customers corresponds to their relative Euclidean distance. Based on the study reported in Cordeau et al. [10], the experiment consisted in performing a single simulation run for each problem instance and reporting on average performance. HGA-VRP has been implemented in C++, using the GAlib genetic algorithm library of Wall [30] and the experiment carried out on a 400 MHz Pentium processor. Solution convergence is assumed to occur when its quality fails to improve by at least 1% over 20 consecutive generations. The parameter values for the investigated algorithm are described below.

In the LNSB_M(d) mutation operator the number of customers considered for elimination runs in the range [15, 21]. The discrepancy factor *d* is randomly chosen over {1,2}.

Parameter values for the proposed genetic operators are defined as follows:

- Population size: 15
- Migration: 5
- Population replacement:
  - Elitism
  - Population overlap per generation: $n_1 = n_2 = 2$
- Recombination:
  - IB_X(*k*=2) (20%)
- Mutation:
  - LNSB_M(*d*) (80%)
  - EE_M (50%), IEE_M (50%)
  - RC_M(*I*=20) - whenever a new best feasible solution is found.

The migration parameter, a feature provided by GAlib, refers to the number of (best) chromosomes exchanged between populations after each generation. Because of limited computational resources, parameter values were determined empirically

over a few intuitively selected combinations, choosing the one that yielded the best average output.

Comparative performance is reported for some of the best-known VRP methods, namely referred to as OS [26], GHL [31], CGL [32], TV [33], WH [34], RR [35], RT [36], TA [37] and BB for HGA-VRP. The results are expressed in terms of total traveled distance. Published competing methods with an average performance gap exceeding about 1% (over all instances) of the best-known result, and/or failing to specify run-time and computational resource characteristics, or reporting prohibitive run-time have been deliberately omitted for comparison purposes. Additional results involving other techniques including classical heuristics may nonetheless be found in Cordeau et al. [10].

**Table 1.** Comparison of selected heuristics for VRP

| Probl. Inst (n) | Perf. Time | OS | GHL | CGL | TV | WH | RR | BB | Best |
|---|---|---|---|---|---|---|---|---|---|
| 1 (50) | Dist | **524.61** | **524.61** | **524.61** | **524.61** | **524.61** | **524.61** | **524.61** | **524.61** |
| | (min) | 1.90 | 6.0 | 4.57 | 0.81 | 20.0 | 1.05 | 2.00 | |
| 2 (75) | Dist | 844 | 835.77 | 835.45 | 838.60 | 835.8 | 835.32 | **835.26** | **835.26** |
| | (min) | 0.84 | 53.8 | 7.27 | 2.21 | 50.0 | 43.38 | 14.33 | |
| 3 (100) | Dist | 838 | 829.45 | 829.44 | 828.56 | 830.7 | 827.53 | 827.39 | **826.14** |
| | (min) | 25.72 | 18.4 | 11.23 | 2.39 | 145.0 | 36.72 | 27.90 | |
| 4 (150) | Dist | 1044.35 | 1036.16 | 1038.44 | 1033.21 | 1038.5 | 1044.35 | 1036.16 | **1028.42** |
| | (min) | 59.33 | 58.8 | 18.72 | 4.51 | 285.0 | 48.47 | 48.98 | |
| 5 (199) | Dist | 1334.55 | 1322.65 | 1305.87 | 1318.25 | 1321.3 | 1334.55 | 1324.06 | **1291.45** |
| | (min) | 54.10 | 90.9 | 28.10 | 7.50 | 480.0 | 77.07 | 55.41 | |
| 6 (50) | Dist | **555.43** | **555.43** | **555.43** | **555.43** | **555.4** | **555.43** | **555.43** | **555.43** |
| | (min) | 2.88 | 13.5 | 4.61 | 0.86 | 30.0 | 2.38 | 2.33 | |
| 7 (75) | Dist | 911.00 | 913.23 | **909.68** | 920.72 | 911.8 | **909.68** | **909.68** | **909.68** |
| | (min) | 17.61 | 54.6 | 7.55 | 2.75 | 45.0 | 82.95 | 10.5 | |
| 8 (100) | Dist | 878.00 | **865.94** | 866.38 | 869.48 | 878.0 | 866.75 | 868.32 | **865.94** |
| | (min) | 49.99 | 25.6 | 11.17 | 2.90 | 165.0 | 18.93 | 5.05 | |
| 9 (150) | Dist | 1184.00 | 1177.76 | 1171.81 | 1173.12 | 1176.5 | 1164.12 | 1169.15 | **1162.55** |
| | (min) | 76.26 | 71.0 | 19.17 | 5.67 | 345.0 | 29.85 | 17.88 | |
| 10(199) | Dist | 1441.00 | 1418.51 | 1415.40 | 1435.74 | 1418.3 | 1420.84 | 1418.79 | **1395.85** |
| | (min) | 76.02 | 99.8 | 29.74 | 9.11 | 535.0 | 42.72 | 43.86 | |
| 11(120) | Dist | 1043.00 | 1073.47 | 1074.13 | 1042.87 | 1043.4 | **1042.11** | 1043.11 | **1042.11** |
| | (min) | 24.07 | 22.2 | 14.15 | 3.18 | 275.0 | 11.23 | 22.43 | |
| 12(100) | Dist | 819.59 | **819.56** | **819.56** | **819.56** | **819.6** | **819.56** | **819.56** | **819.56** |
| | (min) | 14.87 | 16.0 | 10.99 | 1.10 | 95.0 | 1.57 | 7.21 | |
| 13(120) | Dist | 1547.00 | 1573.81 | 1568.91 | 1545.51 | 1548.3 | 1550.17 | 1553.12 | **1541.14** |
| | (min) | 47.23 | 59.2 | 14.53 | 9.34 | 510.0 | 1.95 | 34.91 | |
| 14(100) | Dist | **866.37** | **866.37** | 866.53 | **866.37** | **866.4** | **866.37** | **866.37** | **866.37** |
| | (min) | 19.60 | 65.7 | 10.65 | 1.41 | 140.0 | 24.65 | 4.73 | |
| Average Deviation from Best | | *1.03%* | *0.86%* | *0.69%* | *0.64%* | *0.63%* | *0.55%* | *0.48%* | |
| Average Time (min) | | *33.60* | *46.8* | *13.75* | *3.84* | *222.85* | *24.65* | *21.25* | |

Computational results for all problem data sets are summarized in Table 1. The first column describes the various instances and their related size whereas the second specifies total traveled distance and run-time (in minutes). The following columns refer to particular problem-solving methods. Best-known results are depicted in the last column (Taillard [37] and, Rochat and Taillard [36] for instances 5 and 10). The

last row refers to average run-time and performance deviation from the best-known solutions over all problem instances. Related computer platforms include VAX 8600 for OS, Silicon Graphics 36 MHz for GHL, Sun Ultrasparc 10 (440 MHz) for CGL, Pentium PC 200 MHz for TV, Sun 4/630 MP for WH, Sun Sparc4 IPC for RR, Silicon Graphics 100 MHz for RT, Silicon Graphics 4D/35 for TA and Pentium 400 MHz for BB respectively. Explicit results for RT and TA have been omitted because no run-time was provided. It is worth noticing that reported results for WH includes the best computed solution over five execution runs as well as cumulative run-time.

The results of the experiment do not show any conclusive evidence to support a dominating heuristic over the others. But, solution quality and run-time reported for BB proves the HGA-VRP method to be competitive in comparison to alternate techniques as it mostly matches the performance of best-known heuristic routing procedures. Accordingly, the average solution quality deviation (0.48%) and reasonable run-time obtained certainly shows that hybrid genetic algorithms can be comparable to tabu search techniques.

## 4 Conclusion

A hybrid genetic algorithm (HGA-VRP) to address the classical capacitated vehicle routing problem was presented. Focusing on total traveled distance minimization, HGA-VRP concurrently evolves two populations of solutions in which respective best individuals are mutually exchanged through migration over each generation. Genetic operators were designed to incorporate and combine variations of key concepts emerging from recent promising techniques for a time-variant of the problem, to further emphasize search diversification and intensification. Results from a limited computational experiment showed that HGA-VRP is cost-effective and very competitive in comparison to the best-known VRP metaheuristics.

Future work will be conducted to further improve the proposed algorithm. Existing alternate metaheuristic features and insertion procedures including techniques explicitly designed for the capacitated VRP will be examined to enhance genetic operators while reducing computational cost. Other improvements lie in the introduction of alternate population replacement schemes, fitness models, and an adaptive scheme to dynamically adjust parameters simplifying the configuration procedure in selecting suitable parameters. Application of the approach to other related problems will be explored as well.

## References

1. Toth, P. and D. Vigo (2002), "The Vehicle Routing Problem", *SIAM Monographs Discrete Mathematics and Applications*, edited by P. Toth and D. Vigo, Philadelphia, USA.
2. Laporte, G., M. Gendreau, J.-Y. Potvin and F. Semet (1999), "Classical and Modern Heuristics for the Vehicle Routing Problem", *Les Cahiers du GERAD, G-99-21*, Montreal, Canada.
3. Gendreau, M., G. Laporte and J.-Y. Potvin (1998), "Metaheuristics for the Vehicle Routing: Problem", *Les Cahiers du GERAD, G-98-52*, Montreal, Canada.

4. Gendreau, M., G. Laporte and J.-Y. Potvin (1997), "Vehicle routing: modern heuristics. Local Search in Combinatorial Optimization", eds.: E. Aarts and J.K. Lenstra, 311–336, Wiley:Chichester.
5. Glover, F. (1986), "Future Paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research* 13, 533–549.
6. Glover, F. and M. Laguna (1997), *Tabu Search*, Kluwer Academic Publishers, Boston.
7. Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.
8. Jong De, K. A. (1975), An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. Dissertation, University of Michigan, U.S.A.
9. Goldberg, D.E (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, New York.
10. Cordeau, J.-F., M. Gendreau, G. Laporte, J.-Y. Potvin and F. Semet (2002), "A Guide to Vehicle Routing Heuristics", *Journal of the Operational Research Society* 53, 512-522.
11. Cordeau, J.-F. and G. Laporte (2002), "Tabu Search Heuristics for the Vehicle Routing Problems", *Les Cahiers du GERAD, G-2002-15*, Montreal, Canada.
12. Bräysy, O. and M. Gendreau (2001), "Vehicle Routing Problem with Time Windows, Part II: Metaheuristics", Internal Report STF 42 A01025, SINTEF Applied Mathematics, Department of Optimization, Norway.
13. Dalessandro, S.V., L.S. Ochi and L.M. de A. Drummond (1999), A Parallel Hybrid Evolutionary Metaheuristic for the Period Vehicle Routing Problem. *IPPS/SPDP 1999, 2nd Workshop on Biologically Inspired Solutions to Parallel Processing Problems*, San Juan, Puerto Rico, USA, 183–191.
14. Berger, J. and M. Barkaoui (2000), "An Improved Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows", International ICSC Symposium on Computational Intelligence, part of the International ICSC Congress on Intelligent Systems and Applications (ISA'2000), University of Wollongong, Wollongong, Australia.
15. Machado, P., J. Tavares, F. Pereira and E. Costa (2002), "Vehicle Routing Problem: Doing it the Evolutionary Way", *Proc. of the Genetic and Evolutionary Computation Conference*, New York, USA.
16. Gehring, H. and J. Homberger (2001), "Parallelization of a Two-Phase Metaheuristic for Routing Problems with Time Windows", *Asia-Pacific Journal of Operational Research* 18, 35–47.
17. Tan, K.C., L.H. Lee and K. Ou (2001), "Hybrid Genetic Algorithms in Solving Vehicle Routing Problems with Time Window Constraints", *Asia-Pacific Journal of Operational Research* 18, 121–130.
18. Thangiah, S.R., I.H. Osman, R. Vinayagamoorthy and T. Sun (1995), "Algorithms for the Vehicle Routing Problems with Time Deadlines", *American Journal of Mathematical and Management Sciences* 13, 323–355.
19. Thangiah, S.R. (1995), "Vehicle Routing with Time Windows Using Genetic Algorithms", In *Application Handbook of Genetic Algorithms*: *New Frontiers*, *Volume II*, 253–277, L. Chambers (editor), CRC Press, Boca Raton.
20. Thangiah, S.R. (1995), "An Adaptive Clustering Method using a Geometric Shape for Vehicle Routing Problems with Time Windows", In *Proceedings of the 6th International Conference on Genetic Algorithms*, L.J. Eshelman (editor), 536–543 Morgan Kaufmann, San Francisco.
21. Blanton, J.L. and R.L. Wainwright (1993), "Multiple Vehicle Routing with Time and Capacity Constraints using Genetic Algorithms", In *Proceedings of the 5th International Conference on Genetic Algorithms*, S. Forrest (editor), 452–459 Morgan Kaufmann, San Francisco.
22. Sangheon, H. (2001), "A Genetic Algorithm Approach for the Vehicle Routing Problem", *Journal of Economics*, Osaka University, Japan.

23. Peiris, P. and S.H. Zak (2000), "Solving Vehicle Routing Problem Using Genetic Algorithms", *Annual Research Summary – Part I – Research, Section 1*
24. *Automatic Control***,** School of Electrical and Computer Engineering, Purdue University, http://www.ece.purdue.edu/ECE/Research/ARS/ARS2000/PART_I/Section1/1_19.whtml.
25. Shaw, P. (1998), "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems", In *Principles and Practice of Constraint Programming, Lecture Notes in Computer Science*, M. Maher and J.-F. Puget.(eds.), 417–431, Springer-Verlag, New York.
26. Liu, F.-H. and S.-Y. Shen (1999), "A Route-Neighborhood-based Metaheuristic for Vehicle Routing Problem with Time Windows", *European Journal of Operational Research* 118, 485–504.
27. Osman, I.H. (1993), "Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem", *Annal of Operations Research* 41, 421–451.
28. Solomon, M.M. (1987), "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints", *Operations Research* 35, 254–265.
29. Harvey, W.D. and M.L. Ginsberg (1995), "Limited Discrepancy Search", In Proceedings of the 14th IJCAI, Montreal, Canada.
30. Christofides N., A. Mingozzi and P. Toth (1979), "The Vehicle Routing Problem", in Christofides N., Mingozzi A., Toth P. and Sandi C. (eds). *Combinatorial Optimization,* Wiley, Chichester 315–338.
31. Wall, M. (1995), *GAlib - A C++ Genetic Algorithms Library, version 2.4*. (http://lancet.mit.edu/galib-2.4/), MIT, Boston.
32. Gendreau, M., A. Hertz and G. Laporte (1994), "A Tabu Search Heuristic for the Vehicle Routing: Problem", *Management Science* 40*,* 1276–1290.
33. Cordeau, J.-F., M. Gendreau and G. Laporte (1997), "A Tabu Search Heuristic for the Periodic and Multi-depot Vehicle Routing Problems", *Networks* 30, 105–119.
34. Toth, P. and D. Vigo (1998), "The Granular Tabu Search and its Application to the Vehicle Routing Problem", *Technical Report OR/98/9*, DEIS, University of Bologna, Bologna, Italy.
35. Wark, P. and J. Holt (1994), "A Repeated Matching Heuristic for the Vehicle Routing Problem", *Journal of Operational Research Society* 45, 1156–1167.
36. Rego, C. and C. Roucairol (1996), "A Parallel Tabu Search Algorithm Using Ejection Chains for the Vehicle Routing Problem", In: Osman IH and Kelly JP (eds). *Meta-Heuristics: Theory and Applications*, Kluwer, Boston, 661–675.
37. Rochat, Y. and E.D. Taillard (1995), "Probabilistic Diversification and Intensification in Local Search for Vehicle Routing", *Journal of Heuristics* 1, 147–167.
38. Taillard E.D. (1993), "Parallel Iterative Search Methods for Vehicle Routing Problems", *Networks* 23, 661–673.