Decision Making in Manufacturing and Services
Vol. 3 • 2009 • No. 1–2 • pp. 87–100

Sequential Simulated Annealing for the Vehicle Routing Problem with Time Windows

Marcin Woch*, Piotr Łebkowski*

Abstract. This article presents a new simulated annealing algorithm that provides very high quality solutions to the vehicle routing problem. The aim of described algorithm is to solve the vehicle routing problem with time windows. The tests were carried out with use of some well known instances of the problem defined by M. Solomon. The empirical evidence indicates that simulated annealing can be successfully applied to bi-criterion optimization problems.

Keywords: simulated annealing, vehicle routing problem with time windows, bi-criterion optimization

Mathematics Subject Classification: 90B06, 90C59

Revised: 17 November 2009

1. INTRODUCTION

The aim of a typical Vehicle Routing Problem (VRP) is to design least cost routes from one central depot to a set of geographically scattered points, so called clients. Each point on the routes has to be visited only once by exactly one vehicle. All routes start and end at the central depot and the total demand of all the clients on one route must not exceed vehicle capacity.

The Vehicle Routing Problem with Time Windows (VRPTW) is a generalization of the VRP involving the additional constraint that every customer must be served within a given time window. Time window requirement does not prevent any vehicle from arriving before the allowed start of service at a customer location. The fleet of vehicles is homogenous. The primary objective is minimization of the number of routes, total travelled distance and total duration of routes are considered as secondary aim.

Thus the VRPTW is a bi-criterion optimization problem. The practical application of the VRPTW includes school bus routing, deliveries of goods to department stores, newspaper, mail distribution, maintenance services, etc. The previous works on the VRPTW can be split into two categories: exact optimization and approximate

* Department of Operations Research and Information Technology, AGH University of Science and Technology, Krakow, Poland. E-mail: mwoch@antei.com.pl, plebkows@zarz.agh.edu.pl

algorithms (heuristics). Surveys on exact methods are described in works Desrochers, Desrosiers and Solomon (1992), Halse (1992), Fisher, Jorsten and Madsen (1997), Kohl and Madsen (1997), Kohl *et al.* (1999).

In the second category works by Cordeau *et al.* (2001) and Bräysy and Gendreau (2001a and 2001b) can be cited. Other works worth mentioning describe Tabu Search (Taillard *et al.* 1997, Cordeau, Laporte and Mericier 2000), simulated annealing (Chiang and Russel 1996), genetic algorithms (Berger, Barkaoui and Braysy 2000), evolution strategies (Homberger and Gehring 1999), local search (Shaw 1997, Li, Lim and Huang 2001, Ibaraki *et al.* 2001), constraint programming (Shaw 1998) and ant colony systems (Gambardella, Taillard and Agazzi 1999). The methods described by these papers have been able to solve some of the Solomon (1987) tests sized up to 100 customers. In particular simulated annealing algorithms have been among the most suitable approaches to tackle the VRPTW.

This paper presents a simulated annealing algorithm to solve the VRPTW. The objective is to find the best possible solutions to some Solomon (1987) instances.

In section 2 the VRPTW is formulated. Section 3 describes a sequential annealing algorithm. Section 4 presents the empirical results and the algorithm description, section 5 presents computational results and section 6 concludes the work. It can be concluded that the results obtained with our algorithm are competitive with the other approaches results. For fourteen instances we found the best solutions known so far for three instances we managed to improve current best results (see Table 1 and Appendix).

2. PROBLEM DESCRIPTION

The VRPTW can be described as follows. Given a fleet of homogenous vehicles housed at a central depot and a set of customers requiring goods delivery (or pick up). A set of feasible routes starting from and terminating at the depot must be constructed so that each customer is visited exactly once, and the total distance travelled by the vehicles is minimized.

Each feasible route must also satisfy three types of constraints:

- the total load on a route cannot exceed the capacity of the vehicle servicing the route.
- the time of beginning of service at each customer location must occur before the upper bound of its time window. A vehicle is allowed to arrive before the lower bound but in such a case the vehicle must wait.
- each vehicle must begin and terminate its route within given boundaries defined by a central depot's time window.

Two variants of this problem may be defined by introducing either hard or soft time windows at the customer locations. In the latter scheme the time of beginning of service at any given customer location can occur after the time window's upper bound (although the time window at the central depot must still be strictly satisfied). The variant with soft time windows reduces to the hard one, by adding a large penalty to

the objective value when the time window constraints are not satisfied. Our problem-solving methodology addresses problems with hard time windows.

The formal definition of VRPTW is as follows (Bräysy and Gendreau, 2001). The VRPTW is defined on a graph (N, A). The node set N is composed of the set of customers C, and the central depot which is denoted as nodes 0 and $n+1$, where n represents the total number of customers $|C| = \{1, 2, \ldots, n\}$.

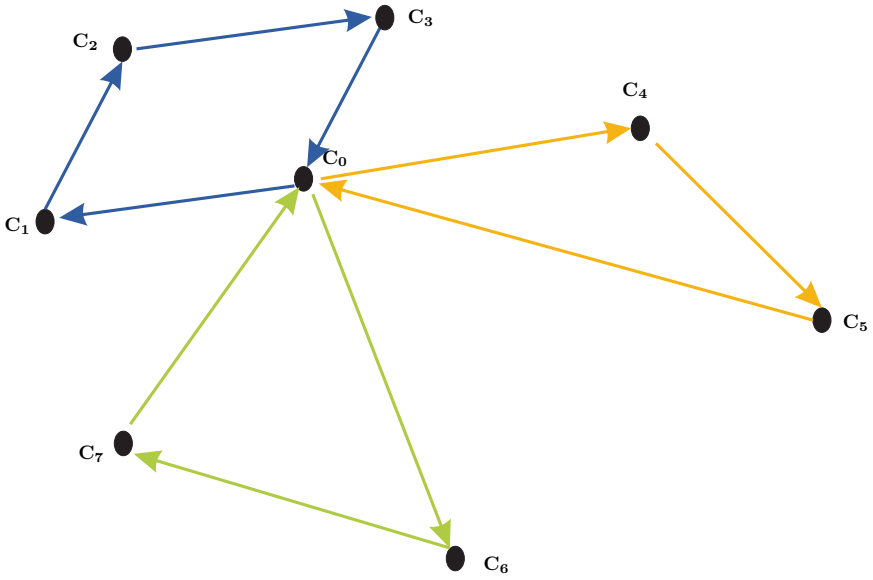

Fig. 1. *Graphical representation of delivery problem*

Table 1. *Data parameters*

(N, A)	–	a graph representing the VRPTW
N	–	node set composed of customers and a depot
A	–	arc set representing connection between nodes
C	=	$1, 2, \ldots, n$ – set of customers
C_0	–	central depot
c_{ij}	–	travel cost between customer C_i and C_j
t_{ij}	–	travel time between customer C_i and C_j
V	–	set of homogenous vehicles
q	–	capacity of a vehicle
d_i	–	demand of customer C_i
$[a_i, b_i]$	–	time window of customer C_i
$[a_0, b_0]$	–	depot's time window

The arc set A represents possible connections between the nodes. All routes (arcs) start at central depot – node 0 and end at node $n+1$. A cost c_{ij} and travel time t_{ij} are associated with each arc $(i,j) \in A$ of the network. The travel time t_{ij} also includes a service time at customer i.

The set of homogenous vehicles is denoted by V. Each vehicle has a given capacity q and each customer a demand $d_i, i \in C$. At each customer, the start of the service must be within a given time interval named a time window, $[a_i, b_i], i \in C$. Vehicles must also leave the depot within the time window $[a_0, b_0]$ and return during its time window $[a_{n+1}, b_{n+1}]$. A vehicle is permitted to arrive before the opening of the time window, and wait at no cost until service becomes possible, but it is not permitted to arrive after the latest time window (hard time windows version of the problem).

Table 2. *Variables*

X_{ij}^k	=	$\forall (i,j) \in A, \forall k \in V = 1$ when vehicle k goes from node i to j, 0 otherwise
S_i^k	–	$\forall i \in N, \forall k \in V$ – time where vehicle k starts service at customer $i, i \in C$

The model contains two types of decision variables. The decision variable X_{ij}^k (defined $\forall (i,j) \in A, \forall k \in V$) is equal 1 if vehicle k drives from node i to node j, and 0 otherwise. The decision variable S_i^k (defined $\forall i \in N, \forall k \in V$) denotes the time vehicle $k, k \in V$, starts service at customer $i, i \in C$. If vehicle k does not service customer i, S_i^k has no meaning. We may assume that $S_0^k = 0, \forall k$, and S_{n+1}^k denotes the arrival time of vehicle k at the depot. The objective is to design a set of minimal cost routes, one for each vehicle, such that all customers are serviced exactly once. Hence, split deliveries are not allowed. The routes must be feasible with respect to the capacity of the vehicles and the time windows of the customers serviced. The VRPTW can be formulated as a mixed integer program:

$$\text{minimize} \sum_{k \in V} \sum_{(i,j) \in A} c_{ij} X_{ij}^k \tag{1}$$

subject to

$$\sum_{k \in V} \sum_{j \in N} X_{ij}^k = 1, \qquad \forall i \in C \tag{2}$$

$$\sum_{i \in C} d_i \sum_{j \in N} X_{ij}^k \leqslant q, \qquad \forall k \in V \tag{3}$$

$$\sum_{j \in N} X_{0j}^k = 1, \qquad \forall k \in V \tag{4}$$

$$\sum_{i \in N} X_{ih}^k - \sum_{j \in N} X_{hj}^k = 0, \qquad \forall h \in C, \ \forall k \in V \tag{5}$$

$$\sum_{i \in N} X_{i,n+1}^k = 1, \qquad \forall k \in V \tag{6}$$

$$X_{ij}^k(S_i^k + t_{ij} - S_j^k) \leqslant 0, \qquad \forall (i,j) \in A, \ \ \forall k \in V \tag{7}$$

$$a_i \leqslant S_i^k \leqslant b_i, \qquad \forall i \in N, \ \ \forall k \in V \tag{8}$$

$$X_{ij}^k \in \{0,1\}, \qquad \forall (i,j) \in A, \ \ \forall k \in V \tag{9}$$

The objective function (1) represents the costs that should be minimized. Constraint (2) ensures that each customer is assigned to exactly one vehicle, and constraint (3) states that no vehicle can service more customers than its capacity. Constraints (4), (5) and (6) are the flow constraints requiring that each vehicle k leaves node 0 once, leaves node $h, h \in C$ if and only if it enters that node, and returns to node $n+1$. Note that constraint (6) is redundant, but is maintained in the model to underline the network structure.

The arc $(0, n+1)$ is included in the network, to allow empty tours. More precisely, we permit an unrestricted number of vehicles, but a cost c_v is put on each vehicle used. This is done by setting $c_{0,n+1} = -c_v$. The value of c_v is sufficiently large to primarily minimize the number of vehicles and secondarily minimize travel costs. Constraint set (7) states that vehicle k cannot arrive at j before $S_i^k + t_{ij}$ if it travels from i to j. Constraint (8) ensures that all time windows are respected and (9) are of integrality conditions.

Lenstra and Rinnooy Kan (1981) proved that the VRP and the VRPTW are NP--hard combinatorial optimization problems and the best possible solutions are found by heuristic algorithms.

3. SIMULATED ANNEALING

The algorithm of simulated annealing was first introduced by Metropolis *et al.* (1953), and then used to solve optimization problems by Kirkpatrick, Gellat and Vecchi (1983), and Cerny (1985).

Simulated annealing is a generalization of a Monte Carlo method for examining the equations of state and frozen states of systems (Metropolis *et al.* 1953). The concept is based on the manner in which metals re-crystallize in the process of annealing. In an annealing process a disordered melt, which starts at high temperature is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered and approaches a "frozen" ground state at a minimal temperature. Hence the process can be thought of as an adiabatic approach to the lowest energy state. If the initial temperature of the system is too low or cooling is done either too slowly or too fast the system may become quenched forming defects or freezing out in unstable states (i.e. trapped in a local minimum energy state).

The original Metropolis scheme was that an initial state of a thermodynamic system was chosen at energy E and temperature T, holding T constant the initial configuration is perturbed and the change in energy $dE = E_i - E_{i-1}$ is calculated. If the change in energy is negative the new configuration is accepted. If the change in energy is positive it is accepted with a probability given by the Boltzmann factor:

$$P = e^{-(dE/T)} \tag{10}$$

This process is repeated sufficient times to give good sampling statistics for the current temperature, and then the temperature is decreased and the entire process repeated until a frozen state is achieved at $T = 0$.

The state of the thermodynamic system is analogical to the current solution of a combinatorial problem (Kirkpatrick *et al.* 1983, Cerny 1985), the energy equation for the thermodynamic system is analogical to the objective function, and ground state is analogous to the global minimum. Avoidance of confinement in a local minimum depends on the annealing schedule, the choice of initial temperature, the number of iterations, epoch length and also on temperature change function.

A comprehensive and detailed description of the subject can be found in Reeves (1995). Concluding, the simulated annealing algorithm performs the search by sampling the neighbourhood randomly. It attempts to avoid becoming prematurely trapped in a local optimum by sometimes accepting an inferior solution. The level of this acceptance depends on the magnitude of the increase in solution cost and on the search time to date (Czech, 2001).

The theoretical behaviour of simulated annealing can be modelled using a non-homogeneous Markov chain in which the probability of moving from solution (state) i to solution j depends on i, j and the current value of the temperature of annealing. The results concerning optimal convergence of such chains can be found in Aarts and Laarhoven (1987), Aarts and Korst (1989), and Hajek (1988).

In the worst case scenario, time complexity $T(n)$ of the sequential simulated annealing algorithm for the VRPTW is

$$T(n) \leqslant an3 = O(n3) \tag{11}$$

where a is the number of cooling stages (Czech, 2001).

4. ALGORITHM DESCRIPTION

The creation of initial solution is performed in the beginning of the simulated annealing algorithm. In current work we used our own method and its algorithm is:

```
1. Order all clients ascending by time window
2. Repeat until client list is empty
   2.1. For i = 1 to MT_CLIENT
      2.1.1. For each route
        2.1.1.1. Find best insertion place for current client on the route
        2.1.1.2. If new route is feasible and total cost < old total cost
                 Save the route and best insertion place
      2.1.2. If the best place for the client is found
             Insert the client in to the best place
      2.1.3. Otherwise
             Insert the client in to the beginning of the new route
      2.1.4. Delete current client from the list
```

```
   2.2. For i = Last client number down to LT_CLIENT
      2.2.1. For each route
        2.2.1.1. Find best insertion place for current client on the route
        2.2.1.2. If new route is feasible and total cost < old total cost
                 Save the route and best insertion place
      2.2.2. If the best place for the client is found
             Insert the client in to the best place
      2.2.3. Otherwise
             Insert the client in to the beginning of the new route
      2.2.4. Delete current client from the list
3. Save found solution
```

In first step all the clients are ordered by time window, the smaller time window the higher position in the list. Thanks to this, the most "problematic" clients are processed first. Time window is calculated according to the formula: Due Date – Ready Time – Service Time.

This method in step 2.1 takes into consideration the clients with the smallest time windows. The number of such clients in single iteration is defined by parameter MT_CLIENT. It is unlikely too many clients with small time windows be packed into one route. That is why in step 2.2 clients with wider time windows are tried to be inserted into the existing routes. The number of clients with wider time window considered in single iteration is defined by parameter LT_CLIENT.

After many experiments we realised that best results are obtained for MT_CLIENT equal 5 and LT_CLIENT equal 2. Matching clients with small time windows to the clients with wider time window allow reducing number of routes in the initial step of simulated annealing. Steps 2.1 and 2.2 are repeated until all clients are inserted on routes. Procedure to find best insertion place of a current client on the route is as follows (Czech 2001):

```
1. For each client C on the current route
   1.1. Insert client C behind current client on the route
   1.2. If the new route is feasible and
        new cost < old cost then
        Save best place of client C
2. Save found solution
```

This function inserts current client behind every client on the route. Then a check is performed if the route is feasible. If so, then algorithm compares costs of previous and new found solutions. Best insertion place is saved provided new solution cost is smaller than previously.

Transfer function is one of basic elements which ensure efficiency of the simulated annealing algorithm. Ability of an algorithm to accept a worse solution and escape local minima depends on annealing parameters as well as a transfer function. This feature is especially important for small temperature values. In current paper own transfer function is presented. The main advantage of described function is its speed and ability to thorough exploration of search space.

The algorithm is:

```
1. Randomly select max. CNUMBER clients from the client list, to do that:
   1.1.  Randomly select route R1
   1.2.  Randomly select client C on the route R1
   1.3.  Find feasible neighbourhood of client C using
         the Increment Radius Method
   1.4.  Delete selected clients from the list
   1.5.  If only one client has been found then randomly
         select another route R2
   1.6.  If possible insert client C in to route R2
   1.7.  Otherwise try to replace either client on route R2
         with the client C. Mark found client C2 to deletion.
   1.8.  Repeat steps 1.3 - 1.8 for client C2
   1.9.  If no. of selected client < CNUMBER
         return to step 1.1
   1.10. If no. of selected clients > CNUMBER
         terminate transfer function
      return to previous feasible solution
2. With probability P order all the routes ascending
   by no. of clients on each route
3. For each selected client C
   3.1.  For each route R
      3.1.1.  Find best place of client C on route R
      3.1.2.  If new distance < old distance
              Remember a route and an insertion position of
              client C on route R
   3.2.  If the best place on route R found for client C
         Insert client C on to route R
   3.3.  Otherwise
      3.3.1. Create a new route
      3.3.2. Insert client C at the beginning of the new route
4. For each route
   4.1.  Try to improve the route replacing clients
         positions
5. With probability 0.75 order all the routes by no. of
   clients ascending
6. Improve all the routes
```

The major part of described algorithm relies on grouping clients by their distance from the current client (step 1.3). The idea derives from "first cluster then route" algorithm described by Sariklis and Powell (2000).

This method is composed of predefined number of iterations. After many experiments we decided to set maximum number of iterations within step 1.3 to 20. In each iteration, distance (so called radius) is calculated based on equation:

$$r_i = ar_{i-1} \tag{12}$$

where i denotes current iteration, a is a parameter which we decided to set to 1.15.

In next step a list of clients which are located in a distance less or equal radius r is created. Such clients are marked to deletion from routes based on probabilit P which is inversely proportional to the radius. This procedure was called by us Increment Radius Method.

Increment Radius Method "prefers" clients which are located near current client. This ensures the groups (clusters) of clients are moved between routes.

Parameter CNUMBER denotes maximum number of clients which can be removed from the routes in order to find another location for them. The experiments showed that best result were achieved when CNUMER was equal 10–15% of all clients.

In step 4 each single route is being improved. Each client from current route is removed and a new better place for it on same route is searched for. The search starts from first client on the route.

In routes improvement step (step 6) the best place for each client is searched for. In order to achieve this, a client is removed from its current position and new feasible insertion place is looked for. If new place is better than old one, new routes are accepted. In step 5 routes are ordered with probability 0.75 by number of clients on route. This allows routes with small number of clients to be removed from the list if new best places are found for their clients.

In order to achieve good optimization results the simulated annealing metaheuristic has to be adjusted to a specific problem. With respect to the VRPTW following parameters have to be taken into consideration:

a. Initial temperature of annealing T0. If T0 is too high then almost all new solutions are accepted and the search produces a series of random solutions. When T0 is too low, very few movements are allowed which reduces the scope of the search. After a series of experiments we decided to set constant initial temperature T0 = 100.
b. Cooling schedule. The most popular temperature reduction function was described by Laarhoven and Aarts (1987) and it bases on geometric reduction $T_i = rT_{i-1}$. Osman (1993) proposed to increase a temperature when the solution had not been improved after the certain number of steps. In presented algorithm we decided to set parameter r = 0.965 and when current solution is not improved after 20% of all the iterations the temperature is adjusted by factor = 1.1.
c. Number of annealing steps executed in each temperature. It is usually related to the size of a solution neighbourhood. In our computational experiments we adopted method proposed by Czech (2001) and the number of annealing steps between n2 and 10n2 were tried out.
d. Termination condition. We experimented with constant number of iterations between 200 - 600 were tried out.
e. Solution Cost. For bi-criterion optimization problems the most frequently used solution cost is calculated according to the equation:

$$Cost = aR + bD \tag{13}$$

This model (equation (1)) is very simple to be implemented and we also decided to adopt this model. Many authors recommend to set constant parameters $a = 10\ 000$

and $b = 1$ and we adopted this as well. Setting parameter $a >> b$ ensures that a primary objective is reduction of number of the vehicles. Secondary objective is to reduce the total travelled distance. R is a number of routes within a solution, D denotes the total distance of all routes.

5. COMPUTATIONAL EXPERIMENTS

The purpose of this study was to create a new algorithm based on the simulated annealing heuristic for solving the vehicle routing problems with time windows. Computational testing of the proposed method was carried out with some test problems of Solomon (1987) (Table 3).

Table 3. *Experimental results*

Problem Instance	Best Published Solution		Best Computed Solutions	
	No. of Vehicles	Distance	No. of Vehicles	Distance
RC101	14	1,696.94	14	1,696.94
RC102	12	1,554.75	12	1,554.75
RC103	11	1,261.67	11	1,261.67
RC104	10	1,135.48	10	1,135.48
RC105	13	1,629.44	13	1,631.18
RC106	11	1,424.73	11	1,424.73
RC107	11	1,230.48	11	1,230.48
RC108	10	1,139.82	10	1,139.82
R201	4	1,252.37	4	1,252.37
R202	3	1,191.70	3	1,191.70
R203	3	939.54	3	939.50
R204	2	825.52	2	825.52
R205	3	994.42	3	994.42
R206	3	906.14	3	906.14
R207	2	890.61	2	890.61
R208	2	726.75	2	726.82
R209	3	909.16	3	909.16
R210	3	939.34	3	939.37
R211	2	892.71	2	885.71

The simulated annealing algorithm presented in this paper, proved its effectiveness to solve the vehicle routing problem with time windows. By making use of it excellent solutions to some instances of Solomon (1987) benchmark set have been found. This ensures that the simulated annealing can be applied with success to bicriterion optimization problems.

6. CONCLUSION

The simulated annealing algorithm described in this paper was implemented using C\AL language which is built-in in Microsoft Dynamics NAV system. The tests were carried out on the Intel 2Duo T9400 2.53 GHz computer with 4GB RAM with the problem instances published by Solomon (1987).

The Solomon tests are composed of 56 instances. Each of them contains one central depot and 100 customers. The locations are defined in Cartesian coordinate systems. It is assumed that the travel times t_{ij} are equal the corresponding Euclidean distances d_{ij} between the customer i and j locations.

The test problems are grouped into six types. The simplest sets C1 and C2 contain clustered clients in terms of geographical distribution. In sets RC1 and RC2 it is semi-clustered with a mix of randomly distributed and clustered customers whereas the customers' locations in problem sets R1 and R2 are generated randomly in a given area according to a uniform distribution. The clients from problem sets R1, C1 and RC1 have narrow time windows. Thus few clients per route are allowed. Problem sets R2, C2 and RC2 have wider windows and more clients per route are allowed.

The objective of our work was to find as good solutions as possible to some Solomon (1987) benchmark problems by using the simulated annealing. For this purpose the problem instances RC101..RC108 and R201..R211 were selected. Table 3 contains the test results obtained. The best known solutions to the instances are marked in bold (the details regarding the best solutions are given in the Appendix). It can be seen (Table 3) that using the simulated annealing we were able to find new best solution to the problem instances R203 and R211. For fourteen instances we found the best solutions known so far. Only for three instances from these groups we found worse solutions as compared to the best known.

Generally it can be concluded that the results obtained with our algorithm are competitive with the other approaches results. We believe that further research on presented algorithm may enhance the quality of solutions to the VRPTW achieved in the present work.

APPENDIX – BEST SOLUTIONS

Problem instance: R203. Distance: 939.50. Vehicles: 3.

route 1:
27-52-18-45-46-36-64-11-62-88-31-30-76-3-79-78-9-35-34-29-68-12-26-13-58-0;
route 2:
94-95-97-92-98-37-42-57-15-43-14-44-38-86-16-85-59-99-96-6-84-83-8-48-47-49-19-10-63-90-32-66-20-70-7-82-17-61-91-100-93-5-60-89-0;
route 3:
50-33-81-65-71-51-1-69-39-67-23-72-73-21-40-53-87-2-41-22-75-56-74-4-55-25-54-24-80-77-28-0.

Problem instance: R211. Distance: 885.71. Vehicles: 2.

route 1:
95-59-92-98-42-15-2-21-73-72-39-67-23-75-22-41-57-87-94-6-53-40-12-76-29-79-33-81-9-71-65-66-20-51-35-34-78-3-77-68-80-24-54-55-25-4-56-74-58-0;
route 2:
28-27-52-69-31-30-63-64-11-19-62-88-7-82-18-83-84-5-99-85-61-16-44-14-38-86-17-45-8-46-48-47-36-49-90-32-10-70-1-50-26-13-97-96-37-43-100-91-93-60-89-0.

REFERENCES

Aarts, E.H.L., Korst, J.H.M.(1989). *Simulated annealing and Boltzmann machines.* Wiley, Chichester.

Aarts E.H.L., van Laarhoven P.J.M. (1987). *Simulated annealing: Theory and applications.* Wiley, New York.

Berger, J., Barkaoui, M. (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.

Bräysy, O., Gendreau, M. (2001). *Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms.* Working Paper, SINTEF Applied Mathematics, Department of Optimisation, Norway.

Bräysy, O., Gendreau, M. (2001). *Vehicle Routing Problem with Time Windows, Part II: Metaheuristics.*, Working Paper, SINTEF Applied Mathematics, Department of Optimisation, Norway.

Cerny, V.(1985). *A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm*, J. of Optimization Theory and Applic. 45, pp. 41–55.

Chiang, W.-C., Russell, R.A. (1996). *Simulated annealing metaheuristics for the vehicle routing problem with time windows.* Annals of Operations Research 63, 3-27.

Cordeau, J.F., Laporte, G., Mericier, A.,(2000) *A unified tabu search heuristic for vehicle routing problems with time windows.* Technical report CRT-00-03, Centre for Research on Transportation, Montreal, Canada.

Cordeau, J.F., Desaulniers, G., Desrosiers, J., Solomon, M.M., Soumis, F. (2001). *The VRP with Time Windows. To appear in The Vehicle Routing Problem*, Chapter 7, P. Toth and D. Vigo (eds), SIAM Monographs on Discrete Mathematics and Applications.

Cordeau, J.F., Laporte, G., Mercier, A. (2001). *A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows*, Journal of the Operational Research Society 52, pp. 928–936.

Cortes, J., Bullo, F. (2009). *Nonsmooth Coordination and Geometric Optimization via Distributed Dynamical Systems.* SIAM Review, 51(1).

Czech, Z.J. (2001). *Parallel simulated annealing for the delivery problem.* Proc. of the 9th EuromicroWorkshop on Parallel and Distributed Processing, Mantova, Italy, 219-2261.

Desrochers, M., Desrosiers, J., Solomon, M. (1992). *A New optimization algorithm for the vehicle routing problem with time windows.* Oper. Res. 40, 342-354, 1992.

Fisher, M.L., Jornsten, K.O., Madsen, O.B.G. (1997). *Vehicle routing with time windows – two optimization algorithms.* Opns. Res. 45, pp. 488–498.

Gambardella, L.M., Taillard, E., Agazzi, G. (1999). *MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows.* In: New Ideas in Optimization, D. Corne, M. Dorigo and F. Glover (eds.), pp. 63–76, McGraw-Hill, London.

Gehring, H. Homberger, J. (1999). *A Parallel Hybrid Evolutionary Metaheuristic for the Vehicle Routing Problem with Time Windows.* Proceedings of EUROGEN99 – Short Course on Evolutionary Algorithms in Engineering and Computer Science, Reports of the Department of Mathematical Information Technology Series. No. A 2/1999, University of Jyväskylä, Finland, pp. 57–64.

Hajek, B. (1988). *Cooling schedules for optimal annealing.* MOR 13, pp. 311–329.

Halse, K. (1992). *Modeling and solving complex vehicle routing problems.* PhD dissertation no. 60, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark, DK-2800 Lyngby, Denmark.

Homberger, J., Gehring, H. (1999). *Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows.* INFOR 37, pp. 297–318.

Ibaraki, T., Kubo, M., Masuda, T., Uno, T., Yagiura, M. (2001). *Effective local search algorithms for the vehicle routing problem with general time windows.* Working paper, Department of Applied Mathematics and Physics, Kyoto University, Japan.

Kirkpatrick, S., Gellat, C.D., Vecchi, M.P. (1983). *Optimization by simulated annealing*, Science 220, pp. 671–680.

Kohl, N., Madsen O.B.G., (1997). *An optimization algorithm for the vehicle routing problem with time windows based on lagrangean relaxation*, Opns. Res. 45, pp. 395–406.

Lenstra J., Rinnooy, K.A. (1981). *Complexity of vehicle routing and scheduling problems*, Networks 11, pp. 221–227.

Li, H., Lim, A., Huang, J. (2001). *Local search with annealing-like restarts to solve the VRPTW.* Working paper, Department of Computer Science, National University of Singapore.

Osman, I.H. (1993). *Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem.* Annals of Operations Research 41, pp. 421–451.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E. (1953). *Equation of state calculation by fast computing machines.* Journ. of Chem. Phys. 21, pp. 1087–1091.

Reeves C.R. (1993). *Modern Heuristics Techniques for Combinatorial Problems.* Halsted Press, New York.

Sariklis, D., Powell, S. (2000). *A Heuristic Method for the Open Vehicle Routing Problem.* Journal of the Operational Research Society, 51, pp. 564–573. ,

Shaw, P. (1997). *A new local search algorithm providing high quality solutions to vehicle routing problems.* Working paper, University of Strathclyde, Glasgow, Scotland.

Shaw, P. (1998). *Using constraint programming and local search methods to solve vehicle routing problems.* [in:] Principles and Practice of Constraint Programming – CP98, Lecture Notes in Computer Science, M. Maher and J.-F. Puget (Eds), Springer-Verlag, New York, pp. 417–431.

Solomon, M.M. (1987). *Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints*, Operations Research 35, pp. 254–265.

Taillard, É., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.-Y. (1997). *A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows.* Transportation Science 31, pp. 170–186.

Tan, K.C., Lee, L.H., Ou, K. (2001). *Hybrid Genetic Algorithms in Solving Vehicle Routing Problems with Time Window Constraints.* Asia-Pacific Journal of Operational Research 18, pp. 121–130.

Tang, J., Pan, Z., Fung, R., Lau, H. (2009). *Vehicle Routing Problem with Fuzzy Time Windows.* In: Fuzzy Sets and Systems Vol. 160(5), pp. 683–695, Amsterdam.

Woch, M. (2004). *Rozwiązanie problemu dostaw z oknami czasowymi za pomocą symulowanego wyżarzania.* STUDIA INFORMATICA Vol. 25, No. 2 (58), s. 67–81, Gliwice.

Woch, M. (2008). *Zarządzanie kosztami zapasów w systemie zintegrowanym klasy ERP Microsoft Dynamics NAV 4.0.* [w:] R. Knosala [red.], Komputerowo Zintegrowane Zarządzanie, Oficyna Wydawnicza Polskiego Towarzystwa Zarządzania Produkcją, T. 2, s. 552–557, Opole.

Woch, M., Łebkowski, P. (2008). *Manufacturing Cost Management in Microsoft Dynamics NAV 5.0 – Case Study.*

Woch, M., Łebkowski, P. (2008). *Rozwiązanie problemu dostaw z oknami czasowymi w systemie Microsoft Dynamics NAV 5.0.* XI Międzynarodowa Konferencja Naukowa Zarządzanie Przedsiębiorstwem – Teoria i Praktyka, Kraków 27-28.11.2008.

Woch, M., Łebkowski, P. (2009). *Rozwiązanie problemu dostaw z oknami czasowymi w systemie Microsoft Dynamics NAV 5.0.* [w:] Łebkowski P. (red.), Innowacyjno-efektywnościowe problemy teorii i praktyki zarządzania, AGH Uczelniane Wydawnictwa Naukowo-Dydaktyczne, s. 73–80, Kraków.