Parallel simulated annealing for the vehicle routing problem with time windows

Zbigniew J. Czech *
Silesia University of Technology, Gliwice,
and Silesia University, Sosnowiec, Poland
e-mail: zjc@polsl.gliwice.pl

Piotr Czarnas
University of Wroclaw, Wroclaw, Poland
e-mail: piotr.czarnas@finexaq.com

Abstract

A parallel simulated annealing algorithm to solve the vehicle routing problem with time windows is presented. The objective is to find the best possible solutions to some well-known instances of the problem by using parallelism. The empirical evidence indicate that parallel simulated annealing can be applied with success to bicriterion optimization problems.

Key words. *Parallel simulated annealing, message passing model of parallel computation, vehicle routing problem with time windows, bicriterion optimization*

1 Introduction

The vehicle routing problem with time windows (VRPTW) considered in this work consists in finding a set of routes originating and terminating at a depot which serves a set of customers. For the purpose of delivery (or pick up) there is a fleet of vehicles, each vehicle of some capacity. The customers have the given delivery demands and a vehicle on its route cannot serve more customers than its capacity permits. For each customer a time interval, called the time window, and a time of service are defined. The aim is to establish a set of routes which covers each customer exactly once, ensures that the service at any customer starts within the time window and preserves the vehicle capacity constraints. Furthermore the set of routes should minimize, firstly, the number of vehicle used, and secondly, the total distance traveled by vehicles. Thus the VRPTW is a bicriterion optimization problem.

The practical applications of the VRPTW include deliveries of goods to department stores, school bus routing, newspaper, laundry and mail distribution, security patrol or maintenance services, etc.

*This research was supported in part by the State Committee for Scientific Research grant BK-280-RAu2-2001.

The previous works on the VRPTW can be divided into two classes, exact optimization and heuristic (or approximate) algorithms. In the first class the works by Desrochers, Desrosiers and Solomon (1992), Halse (1992), Fisher, Jörsten and Madsen (1997), Kohl and Madsen (1997), Kohl et al. (1999) can be cited. The methods developed in these papers have been able to solve to optimality some of the Solomon (1987) benchmark problems of size up to 100 customers. In the second class a variety of metaheuristics to solve the VRPTW were applied. Among them were tabu search (Taillard et al. 1997, Cordeau, Laporte and Mericier 2000), simulated annealing (Chiang and Russel 1996), genetic algorithms (Berger, Barkaoui and Bräysy 2000), evolution strategies (Homberger and Gehring 1999), local search (Shaw 1997, Li, Lim and Huang 2001, Ibaraki et al. 2001), constraint programming (Shaw 1998), ant colony systems (Gambardella, Taillard and Agazzi 1999).

In this work a parallel simulated annealing algorithm to solve the VRPTW is presented. The objective is to find the best possible solutions to some Solomon (1987) instances of this problem by using parallelism. The empirical evidence indicate that parallel simulated annealing can be applied with success to bicriterion optimization problems. To our knowledge the application of parallel simulated annealing to the VRPTW was not reported in the literature. The methods of parallelization of simulated annealing are discussed in Aarts and van Laarhoven (1987), Aarts and Korst (1989), Greening (1990), Abramson (1991), Azencott (1992), Boissin and Lutton (1993), and Verhoeven and Aarts (1996).

In section 2 the problem under investigation is formulated. Section 3 describes a sequential annealing algorithm. In section 4 the parallel simulated annealing algorithm is presented. Section 5 describes the empirical results. Section 6 concludes the work.

2 Problem formulation

The vehicle routing problem with time windows which is a modification of the well-known vehicle routing prob-

lem (VRP) can be formulated as follows. There is a central depot of cargo and n customers (nodes) located at the specified distances from the depot. The locations of the depot ($i = 1$) and the customers ($i = 2, 3, \ldots, n+1$), and the shortest distances d_{ij} and the corresponding travel times t_{ij} between any two locations i and j are given. The cargo have to be delivered to (or picked up from) each customer i according to the delivery demand q_i by a fleet of vehicles. Each vehicle serves a subset of customers on the route which begins and ends at the depot. The vehicles have the same capacity Q. The total sum of demands of customers served by a vehicle on a route cannot exceed Q. For each customer a service time window $[e_i,\ f_i]$ and a service time s_i are defined. e_i and f_i determine, respectively, the earliest and the latest time for start servicing. The customer i is served by a single vehicle exactly once, within the time window $[e_i,\ f_i]$. The vehicle can arrive at the customer before the time window but in such a case it has to wait until time e_i when the service can begin. The latest time for arrival of the vehicle at customer i is f_i. Let b_i be the current time when service can begin at customer i, and let b_{ij} be the current time when service can begin at customer j, given the customer j is inserted immediately after customer i in the route. Then we have $b_{ij} = \max\{e_j,\ b_i + s_i + t_{ij}\}$. A wait time $w_j = e_j - (b_i + s_i + t_{ij})$ is required if a vehicle arrives at customer j before e_j.

The objective is to find the set of routes which guarantees the delivery of cargo to all customers and satisfies the time window and vehicle capacity constraints. Furthermore, the size of the set equal to the number of vehicles needed (primary goal) and the total travel distance (secondary goal) should be minimized.

Lenstra and Rinnooy Kan (1981) proved that the VRP and the VRPTW are NP-hard combinatorial optimization problems.

3 Sequential simulated annealing

The algorithm of simulated annealing which can be regarded as a variant of local search was first introduced by Metropolis et al. (1953), and then used to optimization problems by Kirkpatrick, Gellat and Vecchi (1983), and Cěrny (1985). A comprehensive introduction to the subject can be found in Reeves (1995). The application of simulated annealing to solve the VRPTW is as follows. Initially a solution to the problem is taken as the best solution to the problem known so far, or the solution found using the sequential simulated annealing algorithm (Czarnas 2001). On every step a neighbor solution is determined by either moving the best customer from one route to the best place (in terms of the solution cost) of another route (perhaps empty) or by selecting the best customer and moving it to the best place within its route. All the routes mentioned above are chosen randomly. The neighbor solutions of lower costs obtained in this way are always accepted. The solutions of higher costs are accepted with the probability

$$T_i/(T_i + \delta), \tag{1}$$

where T_i, $i = 0, 1, \ldots, i_{\max}$, is a parameter called a temperature of annealing, which falls from the initial value $T_0 = \gamma \cdot cost(s_0)$ according to the formula $T_{i+1} = \beta T_i$, where γ and $\beta < 1$ are constants and s_0 is the initial solution to the problem, and δ denotes the increase in the solution cost. Equation (1) implies that large increases in solution cost, so called uphill moves, are more likely to be accepted when T_i is high. As T_i approaches zero most uphill moves are rejected. The cost of a solution s is computed as follows:

$$cost(s) = d + \sigma(cn + e_{\min}) \tag{2}$$

where d is the total travel distance of the routes, σ is a constant, c is the number of routes in solution s (equals to the number of vehicles needed), n is the number of customers, and $e_{\min}$ is the number of customers in the shortest route. Since the basic criterion of optimization is the number of routes, the constant σ should be large enough, so that $\sigma(cn + e_{\min}) \gg d$.

The sequential algorithm of annealing stops if equilibrium is encountered. We assume that equilibrium is reached if τ consecutive stages of temperature reduction fail to improve the best solution found so far. Contrary to the classical approach in which a solution to the problem is taken as the last solution obtained in the annealing process, we memorize the best solution found during the whole annealing process.

Summing up, the simulated annealing algorithm performs the local search by sampling the neighborhood 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.

The theoretical behavior of simulated annealing can be modeled 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).

The worst case time complexity of the sequential simulated annealing algorithm for the VRPTW is $\mathcal{T}(n) \leq an^3 = O(n^3)$, where a is the number of cooling stages.

4 Parallel simulated annealing

Let us assume that p processors are available and each of them is capable of generating its own annealing process. The processors can be used either to speed up the sequential annealing algorithm or to achieve a higher accuracy of solutions to a problem. In this work we consider the latter goal. The accuracy of a solution is meant as its proximity to the global optimum solution.

In the parallel simulated annealing algorithm the processes P_1, P_2, ..., P_p co-operate with each other every ω step passing their best solutions found so far. Suppose for a moment that the temperature of annealing, T, is fixed. Let $V_r^{(j)}(T)$, $j = 1, 2, \ldots, p$, $r = 1, 2, \ldots, r_{\max}$ be the Markov chain for each of the processes, let $\mathcal{P}_T(V)$ be a realization of one step of the chain at temperature T and with starting point V, and let $\bar{V}_r^{(j)}$ be the best solutions found by processes $j = 1, 2, \ldots, p$, so far, i.e. between step 1 and r. We assume the following scheme of co-operation:

$$V_{r+1}^{(1)} = \mathcal{P}_T(V_r^{(1)}), \quad (3)$$

$$V_{r+1}^{(j)} = \mathcal{P}_T(V_r^{(j)}) \text{ for } j \neq 1, \text{ and if } r+1 \neq u\omega, \quad (4)$$

$$V_{u\omega}^{(j)} = \mathcal{P}_T(V_{u\omega-1}^{(j)}) \text{ if } cost(\mathcal{P}_T(V_{u\omega-1}^{(j)})) \leq cost(\bar{V}_{u\omega}^{(j-1)}), \quad (5)$$

$$V_{u\omega}^{(j)} = \bar{V}_{u\omega}^{(j-1)} \text{ otherwise.} \quad (6)$$

In this scheme the processes co-operate at steps $u\omega$, $u = 1, 2, \ldots, u_{\max}$, where each step consists of a single realization in the Markov chain, i.e. of an annealing step. The chain for the first process ($j = 1$) is completely independent. The chain for the second process is updated at steps $u\omega$ to the better solution between the best solution found by the first process so far, $\bar{V}_{u\omega}^{(1)}$, and the realization of the last step of the second process, $\mathcal{P}_T(V_{u\omega-1}^{(2)})$. Similarly, the third process chooses as the next point in its chain the better solution between $\bar{V}_{u\omega}^{(2)}$ and $\mathcal{P}_T(V_{u\omega-1}^{(3)})$. Clearly, the best solution found by the l-th process is propagated for further exploration to processes m, $m > l$. As in the sequential annealing, the Markov chains generated by the processes are non-homogeneous since the probability of moving from one solution to another depends not only on the costs of these solutions and the current temperature but also on the cost of a solution computed by the left neighbor in a line of co-operating processes.

The above scheme of co-operation is a modification of the scheme given by Aarts and Laarhoven (1987), Graffigne (1992), and Azencott and Graffigne (1992). Their scheme uses in Equations (5) and (6) the value of $V_{u\omega}^{(j-1)}$ instead of $\bar{V}_{u\omega}^{(j-1)}$. That is, process j updates its chain to the better solution found by its left neighbor in step $u\omega-1$, $\mathcal{P}_T(V_{u\omega-1}^{(j-1)})$, and its own realization of this step, $\mathcal{P}_T(V_{u\omega-1}^{(j)})$.

Now note that the temperature of annealing decreases according to the formula $T_{i+1} = \beta T_i$ for $i = 0, 1, 2, \ldots, i_{\max}$, where $i_{\max}$ is the number of cooling stages. There are two possibilities in establishing the points in which the temperature drops and the processes interact. Namely, we may assume that the processes interact frequently during each of temperature plateaus, or that the temperature drops several times before an interaction takes place. In this work the former approach is adopted.

The parallel simulated annealing algorithm to solve the VRPTW written for the message passing model of parallel computation is shown below. The processes P_j, $j = 1, 2, \ldots, p$, carry out the annealing searches using the same initial solution and cooling schedule as in the sequential algorithm (see section 3). At each temperature process P_j executes n^2 annealing steps (line 24). The processes co-operate every $\omega = n$ step passing their best solutions (lines 25–34). On completion n^2 annealing steps the processes send their best local solutions to process P_0 (line 36). The process P_0 chooses the best global solution among the local solutions, i.e. in the set of best local solutions (line 7). Then it tests whether to update the final solution with the best global solution (lines 8–13). If such an update is made then the equilibrium counter is set to 0. The searches stop when equilibrium is reached, i.e. variable *go* sent to processes P_j takes on value *false* (line 14). Note that equilibrium is encountered if $\tau n^2 p$ annealing steps executed by the processes during τ consecutive temperature falls do not change the final solution.

Parallel simulated annealing for the VRPTW

```
    PROCESS P_0:
1   Take the initial_solution as the best solution
      to the problem known so far;
2   Send initial_solution to processes P_j, j = 1, 2, ..., p;
3   final_solution := initial_solution;
4   equilibrium_counter := 0;
5   while equilibrium_counter ≤ τ do
6     Receive best_local_solution_j from processes P_j,
        j = 1, 2, ..., p;
7     Choose best_global_solution among
        best_local_solution_j, j = 1, 2, ..., p;
8     if cost(best_global_solution) <
        cost(final_solution) then
9       final_solution := best_global_solution;
10      equilibrium_counter := 0; {final_solution
          was updated}
11    else
12      equilibrium_counter := equilibrium_counter + 1;
13    end if;
14    go := (equilibrium_counter ≤ τ);
15    Send go to processes P_j, j = 1, 2, ..., p;
```

```
16  end while;
17  Produce final_solution as the solution to the VRPTW;

    PROCESSES P_j, j = 1, 2, ..., p:
18  Receive initial_solution from P_0;
19  old_solution_j := initial_solution;
20  best_local_solution_j := initial_solution;
21  T := γ * cost(initial_solution); {initial
      temperature of annealing}
22  loop
23    for iteration_counter_j := 1 to n^2 do
24      annealing_step(old_solution_j,
          best_local_solution_j); {as in the sequential
          algorithm}
25      if number of annealing steps executed so far
          is a multiplicity of n then {co-operate}
26        if j = 1 then Send best_local_solution_1
            to process P_2;
27        else {j > 1}
28          receive best_local_solution_{j-1}
              from process P_{j-1};
29          if cost(best_local_solution_{j-1}) <
              cost(best_local_solution_j) then
30            best_local_solution_j :=
                best_local_solution_{j-1};
31          end if;
32          if j < p then Send best_local_solution_j
              to process P_{j+1}; end if;
33        end if;
34      end if;
35    end for;
36    Send best_local_solution_j to P_0;
37    Receive go from P_0;
38    if go = false then stop; end if;
39    T := β * T; {temperature reduction}
40  end loop;
```

The worst case time complexity of the parallel simulated annealing algorithm for the VRPTW is $\mathcal{T}_p(n) \leq pn+a(n^3+(p-1)n^2+pn) = O(n^3+pn^2)$. As compared to the sequential simulated annealing algorithm the complexity is higher by a cost of communication among processes P_1, P_2, ..., P_p (lines 25–34) and between process P_0 and P_1, P_2, ..., P_p (line 18 and lines 36–37). Consider process P_p. There are n steps of communication (lines 25–34) within the **for** loop. The cost of a single communication step is proportionate to $(p-1)n$, as $p-1$ messages of size $O(n)$ are sent and received. Thus the cost of execution of the **for** loop in process P_p is $n^3 + (p-1)n^2$. The cost of communication in line 18 and lines 36–37 is proportionate to pn. Thus the overall cost of execution of the VRPTW does not exceed $pn + a(n^3 + (p-1)n^2 + pn)$, where a is the number of iterations of the **while** loop (lines 5–16).

5 Experimental results

The parallel simulated annealing algorithm described in section 4 was implemented using C language and the message passing interface (MPI) library. The tests of the algorithm were carried out on the IBM RS/6000 SP, SGI Origin 2000 and Sun Enterprise 6500 multiprocessor computers with the problem instances published by Solomon (1987). The Solomon test set consists of 56 problem instances. Each of these instances comprises 100 customers. The location of the depot and the customers are given as integer values from the range 0 .. 100 in a Cartesian coordinate system. It is assumed that the travel times t_{ij} are equal to the corresponding Euclidean distances d_{ij} between the customer locations. The test problems are grouped into six problem types. In problem sets R1 and R2 the customer locations are generated randomly in a given area according to a uniform distribution. The geographical distribution of customers in sets C1 and C2 is clustered, whereas in sets RC1 and RC2 it is semi-clustered with a mix of randomly distributed and clustered customers. Sets R1, C1 and RC1 have narrow time windows and allow fewer customers per route. Problem sets R2, C2 and RC2 have wider windows and allow a larger number of customers per route. As already mentioned the objective of our work is to find as good solutions as possible to some Solomon (1987) benchmark problems by using parallelism. For this purpose the problem instances RC101 .. RC108 and RC201 .. RC208 were selected. Table 1 contains the test results obtained for $p = 5$ processes and many executions of the parallel algorithm. The best known solutions to the instances are marked in bold (the details regarding the best solutions are given in the Appendix). One can see (the last two columns of Table 1) that using parallel simulated annealing we were able to find the new best solution to the problem instances RC202, RC203, RC205, RC206 and RC207. For nine instances we found the best solutions known so far. Only for two instances from the RC group we foud the inferior solutions as compared to the best known. Although the results are quite good we believe that the parallel simulated annealing algorithm needs further refinements (see remarks in section 6). In particular, the suitable values of algorithm parameters are to be selected. So far we have used the following values and ranges of parameters: $\beta = 0.92$, $\gamma = 0.001 .. 1.0$, $\tau = 20 .. 40$, $\sigma = 0.5 .. 5$, $\omega = n = 100$.

6 Conclusions

The parallel simulated annealing algorithm, based on the scheme of co-operation of processes as described in section 4, proved effective to solve the delivery problem (Czech

Probl. inst.	Best published solution			Best comp. solution	
	N	D	Reference	N	D
RC101	**14**	**1696.94**	TBGGP-97	**14**	**1696.94**
RC102	**12**	**1554.75**	TBGGP-97	**12**	**1554.75**
RC103	**11**	**1261.67**	S-98	**11**	**1261.67**
RC104	**10**	**1135.48**	CLM-00	**10**	**1135.48**
RC105	**13**	**1629.44**	BBB-01	**13**	**1629.44**
RC106	**11**	**1424.73**	LLH-01	**11**	**1424.73**
RC107	**11**	**1230.48**	S-97	11	1230.54
RC108	**10**	**1139.82**	TBGGP-97	**10**	**1139.82**
RC201	**4**	**1406.94**	CLM-00	**4**	**1406.94**
RC202	3	1374.27	LLH-01	**3**	**1367.09**
RC203	3	1060.45	HG-99	**3**	**1049.62**
RC204	**3**	**798.46**	GTA-99	**3**	**798.46**
RC205	4	1300.25	IKMUY-01	**4**	**1297.65**
RC206	3	1152.03	IKMUY-01	**3**	**1146.32**
RC207	3	1062.05	CLM-00	**3**	**1061.14**
RC208	**3**	**828.14**	IKMUY-01	3	828.71

Table 1. The test results for the parallel simulated annealing algorithm (N — number of vehicles needed; D — total distance traveled). The best known solutions are marked in bold.

2001). The results of the present work show that the algorithm is also useful for the vehicle routing problem with time windows. By making use of it we found excellent solutions to some instances of Solomon (1987) benchmark set. This proves that parallel simulated annealing can be applied with success to bicriterion optimization problems.

In order to attain good optimization results one has to adjust the meta-heuristic of simulated annealing to the problem being solved. With respect to the VRPTW one needs to decide the following choices:

(i) *Initial temperature of annealing*. We set $T_0 = \gamma \cdot cost(s_0)$, where γ is a parameter and s_0 is the initial solution to the problem. If T_0 is too high then almost all moves are accepted and the search produces a series of random solutions. Each of these solutions can be an initial solution itself. Conversely, when T_0 is too low, little movement is allowed what narrows the scope of the search. Thus T_0 is to be selected as the result of some compromise.

(ii) *Cooling schedule*. We adopt a geometric reduction function of the temperature of annealing, $T_{i+1} = \beta T_i$, where $\beta = 0.92$. This value of β was found to be the best for the delivery problem (Czech 2001) and, as the present work shows, it is also quite a good choice for the VRPTW.

(iii) *Number of annealing steps executed in each temperature*. It is usually related to the size of a solution neighborhood. In our computational experiments the number between n^2 and $10n^2$ steps were tried out.

(iv) *Termination condition*. It derives from the notion of equilibrium. We define that the search reaches a state of equilibrium if τ consecutive stages of temperature reduction fail to improve the best solution found to date.

(v) *Relative importance of the number of routes against the travel distance*. The "shape" of routes generated in the annealing process is assessed by the measure $cn + e_{\min}$, where c is the number of routes in a solution and $e_{\min}$ is the the number of customers in the shortest route. This measure takes on a small value if a solution consists of a small number of routes. This value also decreases when the shortest route gets shorter, what as a consequence may result in reducing the number of routes in a solution. The relative importance of this measure with regard to the travel distance is controlled by parameter σ (see Equation (2)). A question whether Equation (2) is the best way to define a single optimization criterion needed for a simulated annealing algorithm solving a bicriterion optimization problem remains open.

In addition to these choices the following issues have to be addressed in parallel implementations of simulated annealing:

(vi) *Scheme of co-operation of processes*. The scheme investigated is described in section 4.

(vii) *Frequency of interaction*. It is assumed that the processes interact every $\omega = n$ step during a temperature plateau which comprises n^2 annealing steps. As hinted in section 4 the interactions which span over several temperature falls can also be considered.

(viii) *Number of co-operating processes*. In our tests up to $p = 5$ co-operating processes were used.

To conclude, we believe that further research on issues (v)–(viii) may enhance the quality of solutions to the VRPTW achieved in the present work by exploiting the co-operation of processes in the parallel simulated annealing algorithm.

Acknowledgements

We thank the Wroclaw Centre of Networking nad Supercomputing for the computing grant No 04/97, and the Computer Center of the Silesia University of Technology for the similar computing grant which enabled us to obtain the empirical results described in this work.

References

[1] Aarts, E.H.L., and Korst, J.H.M., Simulated annealing and Boltzmann machines, Wiley, Chichester, 1989.

[2] Aarts, E.H.L., and van Laarhoven, P.J.M., Simulated annealing: Theory and applications, Wiley, New York, 1987.

[3] Abramson, D., Constructing school timetables using simulated annealing: sequential and parallel algorithms, Man. Sci. 37, (1991), 98–113.

[4] Azencott, R., Parallel simulated annealing: An overview of basic techniques, in Azencott, R. (Ed.), Simulated annealing. Parallelization techniques, J. Wiley, NY, (1992), 37–46.

[5] Azencott, R., and Graffigne, C., Parallel annealing by periodically interacting multiple searches: Acceleration rates, In: Azencott, R. (Ed.), Simulated annealing. Parallelization techniques, J. Wiley, NY, (1992), 81–90.

[6] Berger, J., Barkaoui, M., and Bräysy, O., A parallel hybrid genetic algorithm for the vehicle routing problem with time windows, Working paper, Defense Research Establishment Valcartier, Canada, 2001.

[7] Boissin, N., and Lutton, J.-L., A parallel simulated annealing algorithm, Parallel Computing 19, (1993), 859–872.

[8] Cěrny, V., A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm, J. of Optimization Theory and Applic. 45, (1985), 41-55.

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

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

[11] Czarnas, P., A simulated annealing algorithm (in Polish), MSc thesis, (2001), in preparation.

[12] Czech, Z.J., Parallel simulated annealing for the delivery problem, Proc. of the 9th Euromicro Workshop on Parallel and Distributed Processing, Mantova, Italy, (February 7–9, 2001), 219–226.

[13] Desrochers, M., Desrosiers, J., and Solomon, M., A new optimization algorithm for the vehicle routing problem with time windows, Oper. Res. 40, (1992), 342–354.

[14] Fisher, M.L., Jörnsten, K.O., and Madsen, O.B.G., Vehicle routing with time windows – two optimization algorithms, Opns. Res. 45, (1997), 488–498.

[15] Gambardella, L.M., Taillard, E., and Agazzi, G., 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.), McGraw-Hill, London, (1999), 63–76.

[16] Graffigne, C., Parallel annealing by periodically interacting multiple searches: An experimental study, in Azencott, R. (Ed.), Simulated annealing. Parallelization techniques, J. Wiley, NY, (1992), 47–79.

[17] Greening, D.R., Parallel simulated annealing techniques, Physica D 42, (1990), 293–306.

[18] Hajek, B., Cooling schedules for optimal annealing, MOR 13, (1988), 311–329.

[19] Halse, K., 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, (1992).

[20] Homberger, J., and Gehring, H., Two evolutionary metaheuristics for the vehicle routing problem with time windows, INFOR 37, 3 (Aug. 1999), 297–318.

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

[22] Kirkpatrick, S., Gellat, C.D., and Vecchi, M.P., Optimization by simulated annealing, Science 220, (1983), 671-680.

[23] Kohl, N., Desrosiers, J., Madsen, O.G.B., Solomon, M.M., Soumis, F., 2-path cuts for the vehicle routing problem with time windows, Transportation Science 33, 1 (Feb. 1999), 101–116.

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

[25] Lenstra, J., and Rinnooy Kan, A., Complexity of vehicle routing and scheduling problems, Networks 11, (1981), 221–227.

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

[27] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., and Teller, E., Equation of state calculation by fast computing machines, Journ. of Chem. Phys. 21, (1953), 1087-1091.

[28] Reeves, C.R., (Ed.) Modern Heuristic Techniques for Combinatorial Problems, McGraw-Hill, London, 1995.

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

[30] Shaw, P., 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, (1998), 417–431.

[31] Solomon, M.M., Algorithms for the vehicle routing and scheduling problems with time window constraints, Operations Research 35, (1987), 254–265, see also `http://w.cba.neu.edu/~msolomon/problems.htm`).

[32] Taillard, E., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y., A tabu search heuristic for the vehicle routing problem with soft time windows, Transportation Science 31, 2, (May 1997), 170–186.

[33] Verhoeven, M.G.A., and Aarts, E.H.L., Parallel local search techniques, Journal of Heuristics 1, (1996), 43–65.

Appendix: Best solutions

The customers are denoted as 2–101. Each route starts and ends at the depot (node 1).

Problem instance: RC101. Distance: 1696.94. Vehicles: 14.
route 1: 64 77 52 23 50 21 25; **route 2**: 70 99 89 54 79 56 69; **route 3**: 28 30 32 31 35 27 33 94; **route 4**: 93 96 63 68 72 95 51 81; **route 5**: 40 43 45 62 82 55 97; **route 6**: 65 91 85 57 67; **route 7**: 60 76 88 98 59 78; **route 8**: 84 24 22 20 19 49 26; **route 9**: 83 12 16 17 10 11 14 18; **route 10**: 73 37 39 42 41 44 38 36; **route 11**: 66 53 100 58 87 75; **route 12**: 15 48 13 74 80 47 5 61; **route 13**: 29 34 86 90 92; **route 14**: 6 46 3 8 7 9 4 2 71 101;

Problem instance: RC102. Distance: 1554.75. Vehicles: 12.
route 1: 34 29 28 27 32 35 51 94 95 81; **route 2**: 43 62 82 91 69 55 97; **route 3**: 93 96 63 30 31 33 90; **route 4**: 66 84 20 24 49 19 22 26 78; **route 5**: 70 89 54 10 11 14 18 13; **route 6**: 15 48 74 79 80 8 56 101 71; **route 7**: 86 64 77 52 23 50 21 25; **route 8**: 83 12 16 17 88 98 76 59; **route 9**: 92 65 100 87 58 75 60 53; **route 10**: 37 45 41 39 42 44 36 38 73; **route 11**: 40 72 68 85 57 67; **route 12**: 3 46 2 4 6 9 7 47 5 61 99;

Problem instance: RC103. Distance: 1261.67. Vehicles: 11.
route 1: 66 100 98 76 59 78; **route 2**: 70 89 54 11 14 17 18 48; **route 3**: 65 21 23 50 20 19 49 22 26 58; **route 4**: 93 96 63 51 68 85 57 67; **route 5**: 13 15 16 12 10 88 60 75 87 53 83 91; **route 6**: 84 25 24 77 90 64 86 52 92; **route 7**: 62 43 44 45 41 39 42 73 72 94 95 81; **route 8**: 34 28 31 33 29 27 30 32 35; **route 9**: 3 46 2 4 6 9 7 56 69; **route 10**: 99 61 74 79 80 8 47 5 101 71; **route 11**: 82 40 37 36 38 55 97;

Problem instance: RC104. Distance: 1135.48. Vehicles: 10.
route 1: 86 64 90 77 85 52 65 84 67; **route 2**: 23 21 50 20 24 22 49 19 26 25; **route 3**: 81 92 57 96 63 68 95 94 72 73 55 97 82; **route 4**: 62 71 2 4 6 46 9 47 5 101 69; **route 5**: 91 66 100 98 76 59 78; **route 6**: 43 45 44 39 38 36 37 41 40 42; **route 7**: 89 61 79 74 80 8 7 3 56; **route 8**: 70 99 54 13 18 17 14 11 83; **route 9**: 93 51 34 33 31 29 27 28 30 32 35; **route 10**: 15 48 16 12 10 88 60 75 87 58 53;

Problem instance: RC105. Distance: 1629.44. Vehicles: 13.
route 1: 73 72 82 42 55 97 95 94; **route 2**: 70 89 79 74 56 69; **route 3**: 43 62 9 7 47 5 4 2 101; **route 4**: 93 96 65 100 53 87 58 75; **route 5**: 91 54 67 57; **route 6**: 66 83 13 12 88 60 98 76 59; **route 7**: 64 63 68 85 52 86 92; **route 8**: 84 20 24 19 23 50 21 78; **route 9**: 3 46 6 8 80 61; **route 10**: 32 30 28 31 29 27 33 35 51 81; **route 11**: 40 37 45 39 41 38 36 44 71; **route 12**: 34 77 90 49 22 26 25; **route 13**: 99 15 48 16 17 10 11 14 18;

Problem instance: RC106. Distance: 1424.73. Vehicles: 11.
route 1: 70 99 89 54 13 11 10 14 18; **route 2**: 15 12 88 60 76 98 59 75; **route 3**: 96 63 64 86 77 52 85 57 67; **route 4**: 43 45 40 41 37 39 42 44 38 36; **route 5**: 73 72 68 31 33 35 51 94 81; **route 6**: 3 46 6 9 8 7 47 5 4 2 101; **route 7**: 93 62 82 91 95 97 55 69; **route 8**: 83 53 100 87 58 23 50 21 25 92; **route 9**: 66 84 65 20 24 22 19 49 26 78; **route 10**: 34 32 30 28 29 27 90; **route 11**: 16 17 48 79 74 80 61 56 71;

Problem instance: RC108. Distance: 1139.82. Vehicles: 10.
route 1: 91 62 82 97 95 94 73 55; **route 2**: 72 68 63 51 64 86 85 93 57 92 81; **route 3**: 70 99 89 54 79 74 80 61 56 71 69; **route 4**: 96 34 31 29 27 28 30 32 35 33; **route 5**: 83 100 53 87 88 60 98 76 59 75; **route 6**: 3 7 8 9 47 5 46 6 4 2 101; **route 7**: 66 84 58 25 23 50 22 24 26 78; **route 8**: 65 52 77 90 19 49 20 21 67; **route 9**: 42 43 45 44 41 39 38 36 37 40; **route 10**: 13 15 48 18 17 16 14 10 12 11;

Problem instance: RC201. Distance: 1406.94. Vehicles: 4.
route 1: 66 15 48 60 76 17 16 12 13 74 79 80 8 7 9 47 4 5 2 56 69; **route 2**: 93 96 64 34 29 28 30 32 31 63 68 72 91 100 58 87 88 10 54 11 98 75 14 18 61 101 71; **route 3**: 43 37 40 73 46 6 3 99 89 62 45 41 39 42 82 95 51 35 33 27 90 49 25 26 78 59; **route 4**: 70 83 53 84 65 20 24 22 19 77 86 85 52 50 23 21 67 57 97 55 38 44 36 94 92 81;

Problem instance: RC202. Distance: 1367.09. Vehicles: 3.
route 1: 92 93 96 86 64 34 29 27 28 30 32 31 63 68 72 62 42 39 41 82 91 85 50 21 67 57 51 35 33 90 49 22 25 26 78 76 59 53; **route 2**: 66 83 13 15 48 16 12 84 65 24 20 52 77 19 23 58 87 88 10 11 98 60 75 14 18 8 5 61 101 71; **route 3**: 99 46 6 4 2 43 37 40 45 70 89 74 17 100 54 79 80 9 7 47 3 56 69 55 44 36 38 73 97 94 95 81;

Problem instance: RC203. Distance: 1049.62. Vehicles: 3.
route 1: 2 4 6 46 61 74 17 16 12 87 58 65 24 22 49 19 77 90 64 86 85 52 23 20 50 21 57 67 53 11 13 15 48 18 14 75 60 98 76 59 78 26 25 84; **route 2**: 82 62 43 40 37 44 45 70 89 99 83 100 88 10 54 79 80 9 7 8 47 5 3 56 69 71 101; **route 3**: 91 66 92 93 96 51 34 33 35 32 30 28 27 29 31 63 68 95 97 55 42 39 41 36 38 73 72 94 81;

Problem instance: RC204. Distance: 798.46. Vehicles: 3.
route 1: 81 92 93 95 63 52 90 77 19 24 22 49 20 50 21 58 100 53 88 10 14 87 75 60 98 76 59 78 26 25 23 84 66 91; **route 2**: 70 99 83 11 12 16 17 18 48 15 13 54 61 79 74 80 8 9 47 46 6 4 2 5 7 3 89 56 101 71 62 69; **route 3**: 82 97 55 42 40 43 45 44 41 37 36 38 39 73 72 94 68 85 86 64 34 33 31 29 27 28 30 32 35 51 96 57 65 67;

Problem instance: RC205. Distance: 1297.65. Vehicles: 4.
route 1: 70 83 12 16 17 48 15 13 89 79 74 80 8 7 9 47 5 4 2 71 101; **route 2**: 3 46 6 43 40 37 73 72 63 95 62 45 41 39 42 82 91 54 99 56 69 44 36 38 55 97 94 92 81; **route 3**: 66 84 65 20 24 22 19 58 87 53 100 10 88 60 76 98 11 67 57 51 35 33 27 90 49 26 78 59; **route 4**: 93 96 34 29 28 30 32 31 64 77 86 68 85 52 50 23 21 25 75 14 18 61;

Problem instance: RC206. Distance: 1146.32. Vehicles: 3.
route 1: 66 84 53 83 13 15 48 17 16 12 60 76 24 22 19 20 50 23 58 100 87 88 98 10 11 14 18 61 56 101 71 69; **route 2**: 73 93 96 63 32 30 28 29 31 34 64 86 77 52 65 85 68 72 95 82 91 67 57 51 35 33 27 90 21 25 49 26 78 59 75; **route 3**: 70 99 3 46 6 45 43 40 39 37 41 42 62 89 54 79 74 80 8 7 9 47 5 4 2 44 36 38 55 97 94 92 81;

Problem instance: RC207. Distance: 1061.14. Vehicles: 3.
route 1: 66 84 65 96 68 32 30 29 31 34 64 77 52 20 22 19 24 76 60 88 75 87 58 23 21 50 26 78 59 98 14 11 18 61 56 101 71 69; **route 2**: 70 99 89 3 6 43 45 41 39 40 42 73 72 94 82 62 91 57 85 86 95 97 93 63 51 35 28 27 33 90 49 25 67 92 81; **route 3**: 83 100 53 10 12 16 17 48 15 13 54 79 74 80 8 7 9 47 5 46 4 2 44 37 36 38 55;