A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows

JEAN BERGER, MOHAMED BARKAOUI
Defence Research Establishment Valcartier, Decision Support Technology Section
2459 Pie-XI Blvd. North, Val-Bélair, PQ, Canada, G3J 1X5
email: jean.berger@drev.dnd.ca, barkaoui@oricom.ca

OLLI BRÄYSY
SINTEF Applied Mathematics
Department of Optimisation, P.O. Box 124 Blindern, N-0314 Oslo, Norway,
email: Olli.Braysy@math.sintef.no

A parallel hybrid genetic algorithm to address the Vehicle Routing Problem with Time Windows is presented. The proposed approach involves parallel co-evolution of two populations. The first population evolves individuals to minimize total traveled distance while the second focuses on minimizing temporal constraint violation to generate a feasible solution. New genetic operators have been designed to incorporate key concepts emerging from recent promising techniques such as insertion heuristics, large neighborhood search and ant colony systems to further diversify and intensify the search. Results from a computational experiment show that the proposed technique matches or outperforms the best-known heuristic routing procedures, providing six new best-known solutions. In comparison, the method proved to be fast, cost-effective and highly competitive.

1. INTRODUCTION

Vehicle routing problems are well known combinatorial optimization problems with considerable economic significance. The Vehicle Routing Problem with Time Windows (VRPTW) has received a lot of attention in the literature recently. This is mostly due to the wide applicability of time window constraints in real-world cases. In VRPTW, customers with known demands are serviced by a homogeneous fleet of vehicles of limited capacity. Routes are assumed to start and end at the central depot. Each customer provides a time interval during which a particular task must be completed such as loading/unloading the vehicle. It is worth noting that the time window requirement does not prevent any vehicle from arriving before the allowed start of service at a customer location. The objective is to minimize the number of tours or routes, and then for the same number of tours, to minimize the total traveled distance, such that each customer is serviced within its time window and the total load on any vehicle associated with a given route does not exceed the vehicle capacity.

A variety of algorithms including exact methods and efficient heuristics have already been proposed for VRPTW. For excellent surveys on exact, heuristic and metaheuristic methods, see Desrosiers et al. (1995), Cordeau et al. (2001) and Bräysy and Gendreau (2001a and 2001b) respectively. In particular, evolutionary and genetic algorithms have been among the most suitable approaches to tackle the VRPTW, and are of particular interest to us.

Genetic algorithms (Holland, 1975; De Jong, 1975 and Goldberg, 1989) 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.

Blanton and Wainwright (1993) were the first to apply a genetic algorithm to VRPTW. They hybridized a genetic algorithm with a greedy heuristic. Under this scheme, the genetic algorithm searches for a good ordering of customers, while the construction of the feasible solution is handled by the greedy heuristic. Thangiah (1995a and 1995b) uses a genetic algorithm to find good clusters of customers within a "cluster first, route second" problem-solving strategy. Thangiah et al. (1995) test the same approach to solve vehicle routing problems with time deadlines.

In the algorithm proposed by Potvin and Bengio, (1996) new offspring are created by connecting two route segments from two parent solutions or by replacing the route of the second parent-solution by the route of the first parent-solution. Mutation is then used to reduce the number of routes and to locally optimize the solution. Berger et al. (1998) present a hybrid genetic algorithm based on removing certain customers from their routes and then rescheduling them with well-known route-construction heuristics. The mutation operators are aimed at reducing the number of routes by rescheduling some customers and at locally reordering customers. Bräysy (1999a, 1999b) continues the study of Berger et al. (1998) by performing a comprehensive sensitivity analysis, and by creating new crossover and mutation operators. Also Berger et al. (1999) and Berger and Barkaoui (2000) have further continued the research direction started in Berger et al. (1998).

Homberger and Gehring (1999) propose two evolutionary metaheuristics based on the class of evolutionary algorithms called Evolution Strategies and three well-known route improvement procedures Or-opt (Or, 1976), λ-interchanges (Osman, 1993) and 2-opt* (Potvin and Rousseau, 1995). Gehring and Homberger (1999 and 2001) use a similar approach with parallel tabu search implementation. Bräysy et al. (2000) hybridize a genetic algorithm with an evolutionary algorithm consisting of several route construction and improvement heuristics. The recent genetic algorithm by Tan et al. (2001) is based on Solomon's (1987) insertion heuristic, λ-interchanges and the well-known PMX-crossover operator. Other recent studies on various metaheuristics for VRPTW can be found in Rochat and Taillard (1995), Taillard et al. (1997), Chiang and Russell (1997), Cordeau et al. (2001) (tabu searches), Gambardella et al. (1999) (ant colony optimization), and Liu and Shen (1999).

The previously proposed metaheuristics show significant variability in performance. They often require considerable computational effort and therefore fail to convincingly provide a single robust and successful technique. Consequently, there is a need to develop more robust, efficient and stable algorithms. The main contribution of this paper is to develop a new Parallel Hybrid Genetic Algorithm (PHGA) for the VRPTW. The proposed method is shown to be fast, cost-effective and highly competitive.

The novelty of the proposed approach is based on a new concept that combines constrained parallel co-evolution of two populations and partial temporal constraint relaxation to improve solution quality. The first population evolves individuals to minimize the total traveled distance while the second focuses on minimizing temporal constraint violation in trying to generate a feasible solution. Imposing a constant number of tours for each solution of a given population, temporal constraint relaxation allows escaping local minima while progressively moving toward a better solution. Populations interact with one another whenever a new feasible solution emerges, decreasing gradually the number of tours imposed on future solutions. New genetic operators have been designed to maximize the number of customers served within their time intervals first, and then temporal constraint relaxation is used to insert remaining unvisited customers. Key principles and variants emerging from recent promising techniques are also captured to further diversify and intensify the search..

The paper is outlined as follows. Section 2 introduces the basic concepts of the proposed parallel hybrid genetic algorithm. The basic principles and features of the algorithm are first described. Then, the selection scheme, recombination and mutation operators are presented. The combination of concepts borrowed or derived from well-known heuristics such as large-neighborhood search (Shaw, 1998), ant colony systems (Gambardella et al., 1999) and route neighborhood-based two-stage metaheuristic (Liu and Shen, 1999) 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. PARALLEL HYBRID GENETIC ALGORITHM

2.1 GENERAL DESCRIPTION

The proposed 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.

Our approach relies upon constrained parallel co-evolution and partial constraint relaxation. Two populations Pop_1 and Pop_2, primarily formed of non-feasible solution individuals, are evolving concurrently, each with their own objective functions. Pop_1 contains at least one feasible solution and is used to minimize total traveled distance while Pop_2 focuses on minimizing constraint violation. Constrained to a fixed number of tours over the same population, solution individuals differ by exactly one route across both populations. Parallel evolution is interrupted whenever a new best feasible solution is obtained. Populations are then reinitialized and co-evolution resumed, while decreasing the number of routes associated with solution individuals by one. The number of tours imposed on solution individuals in Pop_1 and Pop_2 are $R_{\min}$ and $R_{\min} - 1$, respectively. $R_{\min}$ refers to the number of routes found in the best feasible solution obtained so far. As a new feasible solution emerges from Pop_2, population Pop_1 is replaced by Pop_2, $R_{\min}$ is updated and, Pop_2 is reinitialized with the revised number of tours ($R_{\min}$ -1), using the RSS_M mutation operator. In addition, a post-processing procedure (RC_M) aimed at reordering customers, is applied to further improve the new best solution. Based on their current best solutions, populations share information through pheromone trail updates based on most promising connections linking consecutive customers. This information is then used by the ant colony system-based genetic operator. The evolutionary process is repeated until a predefined stopping condition is met.

The proposed approach uses a steady-state genetic algorithm that involves overlapping populations. At first, new individuals are generated and added to the current population Pop_p. 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

$$Eval_i = E_i + CV_i, \tag{1}$$

where

$$E_i = r_i - r_m + \frac{d_i}{\max\{d_m, d_i\}}, \tag{2}$$

$$CV_i = \sum_{j=1}^{n} \alpha_j \max\{0, b_j^i - l_j\} + \beta\, Viol_i \tag{3}$$

r_i = number of routes in individual *i*,

r_m = lower bound for number of routes (ratio of total demand over vehicle capacity),

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

d_m = average traveled distance over the individuals forming the initial population,

n = number of customers,

α_j = penalty associated with temporal constraint violation *j*,

b_j^i = scheduled time to visit customer *j* in individual *i*,

l_j = latest time to visit customer *j*,

β = penalty associated with number of violated temporal constraints,

$Viol_i$ = number of temporal constraints violated in individual *i*.

The proposed evaluation expression indicates that better individuals generally (but not necessarily) include fewer routes, and smaller total traveled distance, while satisfying temporal constraints. The general algorithm is 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 the n_p worst individuals from Pop_p using the evaluation function (1).
      p=p+1
   Until (all populations Pop_p have been considered)
   if (Pop_2 includes a new best feasible solution) ***then***
         {eliminate all Pop_1 individuals}
         Set Pop_1 = Pop_2
         Modify Pop_2 solutions by applying RSS_M {reduces number of routes by one}.
   endif
      Apply RC_M on the best solution {customer reordering}
      Update desirability matrix {pheromone trails update}
Until(convergence criteria **or** max number of generations)

Feasible solutions for initial populations are first generated using a sequential insertion heuristic in which customers are inserted in random order at randomly chosen insertion positions within routes. The initialization procedure then proceeds as follows:

For p = 1..2 ***do*** {revisit Pop_1 and Pop_2}
  For j = 1..n_p ***do***
    Generate a new solution S_j using the EE_M mutator (defined in Section 2.3.2)
    Add S_j in Pop_p
  end for
  Remove the n_p worst individuals from Pop_p using $Eval_i$
end for
Determine R_{min}, the minimum number of tours associated with a feasible solution in Pop_1 or Pop_2.
Replicate (if needed) best feasible solution (R_{min} routes) in Pop_1.
Replace Pop_1 individuals with R_{min}-route solutions using the procedure RI(R_{min}).
Replace Pop_2 members with R_{min}-1 route solutions using the procedure RI(R_{min}-1).

RI(r) is a re-initialization procedure creating an r-route solution. It first generates r one-customer routes formed from randomly selected customers. Then, it uses the insertion procedure proposed by Liu and Shen (1999) to insert as many customers as possible without violating time window constraints. Accordingly, customer route-neighborhoods are repeatedly examined for insertion. The next customer for insertion is selected by maximizing a so-called regret cost function that accounts for multiple route insertion opportunities:

$$\text{regret cost} = \sum_{r \in RN(i)} \{c_i(r) - c_i(r^*)\}, \tag{4}$$

where

$RN(i)$ = route-neighborhood of customer i,

$c_i(r)$ = minimum insertion cost of customer i within route r,

$c_i(r^*)$ = minimum insertion cost of customer i over its route-neighborhood.

Remaining unvisited customers (if any) are then inserted in the *r*-tour solution maximizing an extended insertion regret cost function, in which $c_i(r)$ includes an additional contribution reflecting temporal constraint violations:

$$\sum_{j=1}^{n_r} \alpha_j \max\{0, b_j - l_j\} + \beta\, Viol_r \qquad (5)$$

in which

n_r = current number of customers in route *r*,

α_j = penalty associated with temporal constraint violation *j*,

β = penalty associated with the number of violated temporal constraints,

b_j = scheduled time to visit customer *j* in route *r*,

l_j = latest time to visit customer *j*,

$Viol_r$ = current number of temporal constraints violated in route *r*.

2.2 SELECTION

The selection process consists of 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 (Goldberg, 1989). In this scheme, the probability of selecting an individual is proportional to its fitness value. An individual fitness value is computed as follows

Population Pop_1:

$$fitness_i = d_i + \sum_{j=1}^{n} \alpha_j \max\{0, b_j^i - l_j\} + \beta\, Viol_i \qquad (6)$$

Population Pop_2:

$$fitness_i = \sum_{j=1}^{n} \alpha_j \max\{0, b_j^i - l_j\} + \beta\, Viol_i \qquad (7)$$

The notations are the same as in equations 1–3. Better individuals generally (but not necessarily) tend to include short total traveled distance in Pop_1 and satisfy as many temporal constraints as possible in Pop_2.

2.3 GENETIC OPERATORS

The proposed genetic operators mostly rely on two basic principles. First, for a given number of tours, an attempt is made to construct feasible solutions with as many customer visits as possible. Second, the remaining customers are inserted into existing routes through temporal constraint relaxation. Constraint violation is used to restrict the total number of routes to a constant value. The proposed genetic operators incorporate some key features of the best heuristic routing techniques such as Solomon's (1987) insertions heuristic I1, large neighborhood search (Shaw, 1998), ant colony systems (Gambardella et al., 1999) and the route neighborhood-based two-stage metaheuristic (RNETS) (Liu and Shen, 1999). Details on the recombination and mutation operators used are given in the next sections.

2.3.1 RECOMBINATION

Two recombination operators are considered, namely IB_X(k) and IRN_X(k). The insertion-based IB_X crossover operator creates an offspring by combining, one at a time, k routes of parent solution P_1 with a subset of customers, formed by nearest-neighbor routes $\{r_2\}$ in parent solution P_2. The k routes ($\{r_1\}$) are selected either randomly, with a probability proportional to the relative number of customers or based on the average distance separating consecutive customers on the routes. A removal procedure is first carried out to remove from r_1 some key customers believed to be most suitably relocated within some alternate routes. More precisely, the stochastic customer removal procedure removes either randomly some customers, customers rather distant from their successors, or customers with waiting times. Then, a modified insertion heuristic of Solomon (1987) is applied to build a feasible route, considering the modified partial route r_1 as the initial solution and unrouted customers in routes r_2 for insertion. The I1 standard insertion heuristic of Solomon (1987) is coupled to a random customer selection procedure, to choose the next candidate customer to be routed. 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 selected from P_1. Finally, if necessary, the child inherits the remaining "diminished" routes of P_1. If unrouted customers still remain, additional routes are built using a nearest-neighbor procedure of Solomon (1987). 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 (2000). In order to keep the number of routes of a child solution identical to its parents, a post-processing procedure is applied. If the solution has a larger number of routes than expected, the RSS_M (Section 2.3.2) procedure is used repeatedly to reduce the number of routes. Conversely, for solutions having a smaller number of routes, new feasible routes are constructed repeatedly by breaking the most populated route in two until the targeted number of routes is obtained.

IRN_X (insert in route neighborhood) limits the total number of routes to a constant value for a solution. IRN_X first removes illegally routed customers that violate the time window constraints from their routes. Additional customers are then removed using the same strategy presented in IB_X. In the following reinsertion phase, feasible solutions are constructed by routing as many customers as possible relying on the same reinsertion technique as used in IB_X. Then, similarly to the RI re-initialization procedure described in Section 2.1, the remaining customers are inserted into existing routes using the insertion procedure proposed by Liu and Shen (1999) in which the regret cost function (equation (4)) has been extended to include a constraint violation contribution (equation (5)).

2.3.2 MUTATION

A suite of six mutation operators is proposed, namely AC_M, LNSB_M, EE_M, RS_M, RSS_M and RC_M. The AC_M (ant colony) mutation operator generates a new solution offspring using the ant colony system approach developed by Gambardella et al. (1999). While traveling from a food source to the nest and vice versa, ants deposit on the ground a substance called pheromone, thus creating a pheromone trail. If pheromone trails (or desirability) are present, ants tend to follow with a higher probability the pheromone trail having the largest concentration. As ants deposit additional pheromone when traveling, preferential paths are reinforced for future visits. Further details about ant colony optimization can be found in Dorigo et al. (1999). For a fixed number of routes, the AC_M operator first relies on the insertion procedure proposed in Gambardella et al. (1999) to create feasible routes. More precisely, two measures are associated with each arc: the attractiveness N_{ij} and the pheromone trail T_{ij}. The attractiveness N_{ij} is computed by taking into account the distance between customers, the time window of the considered customer and the number of times the considered customer has not been inserted in the solution. The tours are constructed using the nearest-neighbor heuristic with probabilistic rules, i.e., the next customer node to be inserted at the end of the current tour is not always the best according to N_{ij} and T_{ij}. Unrouted customers are then inserted into existing routes as specified in Liu and Shen (1999) but using an extended regret cost function (equation (4)) that accounts for temporal constraint violation (equation (5)).

The LNSB_M (Large Neighborhood Search -based) mutation operator relies on the concepts of the Large Neighborhood Search (LNS) proposed by Shaw (1998). The LNS consists of exploring the search space by repeatedly removing related customers and reinserting them using constraint-based tree search (constraint programming). As in Shaw (1998), a set of related customers is first removed. In addition, LNSB_M removes customers violating temporal constraints from their routes. The proposed customer re-insertion method differs from the procedure proposed by Shaw (1998) in two respects, namely, the insertion cost function used, and the order in which customers are considered for insertion (variable ordering scheme) during the branch-and-bound search process. Unvisited customers (if any) are then reinserted using the same customer re-insertion method while relaxing temporal constraints. Insertion cost is defined by the sum of key contributions referring respectively to increased traveled distance, and delayed service time, as specified in Solomon’s (1987) procedure I1 (c_{11}+c_{12}), as well as to constraint violation (equation (5)). Concerning customer visit ordering, customers ($\{c\}$) are sorted

(*CustOrd*) according to a composite ranking. 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 ($Rank_{|Pos|}$(c)) on the other hand:

$$CustOrd \leftarrow Sort\{c\} \ (Rank_{Cost}(c) + Rank_{|Pos|}(c) \ ) \tag{8}$$

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

$$customer \leftarrow CustOrd[INTEGER(L \ rand^{D})] \tag{9}$$

in which

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).

Once a customer is selected, tree search is carried out over its different insertion positions as specified in Shaw (1998). However, the search tree expansion is initiated using a non-constant discrepancy factor, selected randomly over the set {1,2,3}.

The EE_M (edge exchange) and RS_M (repair solution) mutators focus on inter-route improvement. EE_M uses the λ-interchange mechanism of Osman (1993), performing reinsertions of customer sets over two neighboring routes. Here, route neighborhood is determined by route centroid proximity. Customer exchanges take place as soon as the solution improves, i.e., we use the first-accept strategy. Assuming the notation (x,y) to describe the different sizes of customer sets (λ) issued from the neighboring routes, the current operator explores values running over the range (x=1, y=0,1,2). The RS_M mutation operator focuses on exchanges involving one illegal customer. Each illegal customer in a route is exchanged with an alternate legal one or two-customer sequence in order to generate a new set of customers with either violated or non-violated temporal constraints. The objective is to further explore the solution space (diversity) while possibly improving quality.

The RSS_M (reinsert shortest Solomon) mutation operator tries to eliminate the shortest route (smallest number of customers) of the solution, decreasing by one the total number of routes. Customers from the shortest route are first removed. Then, following an iterative process, unvisited customers are re-inserted into existing routes using the insertion procedure proposed by Liu and Shen (1999) in which the regret cost function (equation (4)) has been extended to include a constraint violation contribution (equation (5)). The entire iterative process is repeated over I different sets (e.g. I=20) of randomly generated parameter values.

The RC_M (reorder customers) mutation operator is an intensification procedure that tries to reduce the total distance of feasible solutions by reordering customers within a route. The procedure consists of repeatedly reconstructing a new tour using the sequential insertion procedure I1 of Solomon (1987) over I different sets (e.g. I=2) of randomly generated parameter values.

3. COMPUTATIONAL EXPERIMENT

A computational experiment has been conducted to compare the performance of the proposed algorithm with some of the best techniques designed recently for VRPTW. The algorithm has been tested with 56 VRPTW benchmark problems of Solomon (1987). Each problem involves 100 customers, randomly distributed over a geographical area. The travel time separating two customers corresponds to their relative Euclidean distance. Customer locations for a problem instance are either generated randomly using a uniform distribution (problem data sets R1 and R2), clustered (problem data sets C1 and C2) or mixed, combining randomly distributed and clustered customers (problem data sets RC1 and RC2). The proposed algorithm has been implemented in C++, using the GAlib genetic algorithm library of Wall (1995) and the experimental tests were carried out on a Pentium 400 MHz processor with 128M of RAM. The maximum run time was limited to 1800 seconds and the experiment consisted of performing three simulation runs for each problem instance in a given data set. The parameter values for the investigated algorithm are described below.

Within the LNSB_M(d) mutation operator the number of customers considered for elimination varies within the range [12, 17]. The discrepancy factor *d* is randomly chosen over {1,2,3}. In fitness, evaluation and insertion cost functions

$$\alpha_j = 100, \ \forall j$$

$$\alpha_0 = 1000$$

$$\beta = 100$$

The probabilities and parameter values for the proposed genetic operators are defined as follows. For all data sets except C1 and C2:

- Population size: 10
- Pop_1:
  - Population overlap per generation: n_1=1
  - LNSB_M(*d*) (100%)
  - RS_M + EE_M (50%), EE_M (50%)
- Pop_2:
  - Population overlap per generation n_2=2.
  - IRN_X(*k*=2) (50%)
  - LNSB_M(*d*) (50%)
  - RS_M + EE_M (100%)

For data sets C1 and C2:

- Population size: 50
- Pop_1:
  - Population overlap per generation: n_1=25
  - IB_X(*k*=2) (100%) (for C2: *k*=1)

RC_M(I=2) (100%)

Pop_2:

Population overlap per generation n_2=2.

IRN_X(k=2) (100%) (for C2: k=1)

Because of limited computational resources, the parameter values were determined by trying just a few intuitively selected combinations, and selecting the one that yielded the best average output. This is justified by the fact that the sensitivity of the results with respect to changes in the parameter values such as recombination and mutation rates was found to be generally quite small. For a matter of run-time convenience, different parameter settings are proposed for C1 and C2, as opposed to other data sets. The chosen parameters were inspired from a previous genetic algorithm by Berger and Barkaoui (2000). In fact, this class of problem instances does not present a real challenge for most VRPTW metaheuristics as convergence generally occurs very quickly.

The results for the six problem data sets are summarized in Tables 1-3 for some of the best reported methods for VRPTW, namely GTA (Gambardella et al., 1999), RT (Rochat and Taillard, 1995), SW (Shaw, 1998), KPS (Kilby et al., 1999), TB (Taillard et al., 1997), CR (Chiang and Russell, 1997), LS (Liu and Shen, 1999), HG (Homberger and Gehring, 1999), GH (Gehring and Homberger, 1999), CLM (Cordeau et al., 2001) and BBB for our parallel hybrid genetic algorithm. The results are usually ranked according to a hierarchical objective function, where the number of vehicles is considered as the primary objective and, for the same number of vehicles, the secondary objective is often either total traveled distance or total duration of routes. An exception is found in KPS, where the only objective is to minimize total distance.

Table 1 presents the average results of various well-known procedures. The average is computed over both the independent run sequences and all problem instances in the corresponding data set. Each entry refers to the best average performance obtained with a specific technique over a particular data set. The first column describes the various data sets and corresponding measures of performance defined by average number of routes (or vehicles), total traveled distance and run-time in seconds. The following columns refer to particular problem-solving methods. The performance of our PHGA is depicted in the last column (BBB). Related computer platforms include Sun UltraSparc 1 167 MHz (70Mflops/s) for GTA, Silicon Graphics 100 MHz (15 Mflop/s) for RT, Sun Ultra Sparc 143 MHz (63 Mflop/s) for SW, DEC Alpha (25 Mflops/s) for KPS, Sun Sparc 10 50 MHz (10Mflops/s) for TB, Pentium 200 MHz (24 Mflop/s) for GH and Pentium 400 MHz (54 Mflops/s) for BBB respectively.

Table 1: Average performance comparison among VRPTW algorithms.

Problem		**GTA**	**RT**	**SW**	**KPS**	**TB**	**GH**	**BBB**
R1	**Vehicles**	12.38	12.58	12.33	12.67	12.33	12.41	12.17
	Distance	1210.83	1197.42	1201.79	1200.33	1220.35	1201	1251.40
	Time	1800	2700	3600	2900	13774	300	1800
R2	**Vehicles**	3.00	3.09		3.00	3.00	2.91	2.73
	Distance	960.31	954.36		966.56	1013.35	945	1056.59
	Time	1800	9800		2900	20232	300	1800
C1	**Vehicles**	10.00	10.00		10.00	10.00	10.00	10.00
	Distance	828.38	828.45		830.75	828.45	829	828.50
	Time	1800	3200		2900	14630	300	1800
C2	**Vehicles**	3.00	3.00		3.00	3.00	3.00	3.00
	Distance	591.85	590.32		592.29	590.91	590	590.06
	Time	1800	7200		2900	16375	300	1800
RC1	**Vehicles**	11.92	12.33	11.95	12.12	11.9	12.00	11.88
	Distance	1388.13	1269.48	1364.17	1388.15	1381.31	1356	1414.86
	Time	1800	2600	3600	2900	11264	300	1800
RC2	**Vehicles**	3.33	3.62		3.38	3.38	3.25	3.25
	Distance	1149.28	1139.79		1133.42	1198.63	1140	1258.15
	Time	1800	7800		2900	11596	300	1800

The results of the experiment do not show any conclusive evidence to support a dominating heuristic over the others. But, on average, PHGA proves to be competitive as it mostly matches the performance of best-known heuristic routing procedures. The robustness shown over the quality of the computed solutions suggests a small simulation sample is acceptable. Solution quality/run-time ratio reported for the procedure is also very good in comparison to alternate techniques.

Table 2: Best performance comparison among VRPTW algorithms.

Problem		RT	LS	CR	TB	GTA	HG (ES1)	HG (ES2)	BBB
R1	**Vehicles**	12.25	12.17	12.17	12.17	12.00	11.92	12.00	11.92
	Distance	1208.50	1249.57	1204.19	1209.35	1217.73	1228.06	1226.38	1221.1
R2	**Vehicles**	2.91	2.82	2.73	2.82	2.73	2.73	2.73	2.73
	Distance	961.72	1016.58	986.32	980.27	967	969.95	1033.58	975.43
C1	**Vehicles**	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00
	Distance	828.38	830.06	828.38	828.38	828.38	828.38	828.38	828.48
C2	**Vehicles**	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
	Distance	589.86	591.03	591.42	589.86	589.86	589.86	589.86	589.93
RC1	**Vehicles**	11.88	11.88	11.88	11.50	11.63	11.63	11.50	11.50
	Distance	1377.39	1412.87	1397.44	1389.22	1382.42	1392.57	1406.58	1389.89
RC2	**Vehicles**	3.38	3.25	3.25	3.38	3.25	3.25	3.25	3.25
	Distance	1119.59	1204.87	1229.54	1117.44	1129.19	1144.43	1175.98	1159.37
ALL	**Vehicles**	**415**	**412**	**411**	**410**	**407**	**406**	**406**	**405**
	Distance	57231	59317	58502	57522	57516	57876	58921	57952

The best computed results are shown in Table 2. Results indicate that PHGA matches or outperforms the best-known heuristic routing procedures. The last row refers to the cumulative number of routes and traveled distance over all problem instances. The total number of tours computed over all problem data sets outperform by one the best-computed result so far, reported by Homberger and Gehring (1999). In addition, PHGA is the only method that found the minimum number of tours consistently for all problem data sets. PHGA also succeeded in improving six of the best-known solutions. Accordingly, Table 3 provides six new best-known solutions and compares them with the previous best-known solutions. Details of the new solutions are presented in Appendix I.

Table 3: New best computed solutions for some Solomon problem instances

Problem	Best-Known Solutions		New Best Solutions		
	Vehicles	Distance	Reference	Vehicles	Distance
R108	9	963.99	SW	9	960.88
R110	10	1125.04	CLM	10	1119
RC105	13	1637.15	HG	13	1629.44
RC106	11	1427.13	CLM	11	1424.73
R210	3	955.39	HG	3	954.12
R211	2	910.09	HG	2	906.19

4. CONCLUSION

A parallel hybrid genetic algorithm (PHGA) to address the Vehicle Routing Problem with Time Windows (VRPTW) was presented. PHGA synergistically combines constrained parallel co-evolution and partial temporal constraint relaxation to achieve a more thorough exploration of the search space. The approach exploits two populations made of solution individuals, focusing on the minimization of traveled distance and temporal constraint violation, respectively. New genetic operators were designed to incorporate new variants and key concepts emerging from recent promising techniques (such as, insertion heuristics, large neighborhood search and ant colony systems) in order to ensure diversification and intensification of the search. Results from a computational experiment showed that PHGA is cost-effective and very competitive, matching and even outperforming the best-known VRPTW metaheuristics. The developed heuristic improved the overall total number of tours while generating six new best-known solutions for Solomon data sets.

Future work will be conducted to further improve the proposed algorithm. The computational cost of key genetic operators will be reduced, and alternate metaheuristics features and insertion procedures examined. For instance, new mixed variable/value ordering strategies for the large neighborhood search -based mutator will be explored to take advantage of route contention during route construction and to carry out better customer insertion. Parallel implementation of the proposed technique will be explored as a natural step to gain significant speed-up as well. Accordingly, the improved procedure will be tested and compared over larger problem instances.

ACKNOWLEDGEMENTS

This work was partially supported by the Department of National Defence, Canada, and by the TOP project funded by the Research Council of Norway. This support is gratefully acknowledged. We would also like to thank Dr. Geir Hasle (SINTEF Applied Mathematics, Norway) for his valuable comments.

REFERENCES

Berger, J., M. Salois and R. Begin (1998), “A Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows”, *Lecture Notes in Artificial Intelligence* 1418, AI’98, Advances in Artificial Intelligence, Vancouver, Canada, 114–127.

Berger J., M. Sassi and M. Salois (1999), “A Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows and Itinerary Constraints”, In *Proceedings of the Genetic and Evolutionary Computation Conference*, Orlando, USA, 44–51.

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.

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.

Bräysy, O., J. Berger and M. Barkaoui (2000), “A New Hybrid Evolutionary Algorithm for the Vehicle Routing Problem with Time Windows”, Presented in Route 2000 Workshop, Skodsborg, Denmark.

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

Chiang, W.-C. and R.A. Russell (1997), “A Reactive Tabu Search Metaheuristic for the Vehicle Routing Problem with Time Windows”, *INFORMS Journal on Computing* 9, 417–430.

Cordeau, J.F., G. Desaulniers, J. Desrosiers, M.M. Solomon, and F. Soumis (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., G. Laporte and A. Mercier (2001), “A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows”, *Journal of the Operational Research Society* 52, 928–936.

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.

Desrosiers, J., Y. Dumas, M.M. Solomon and F. Soumis (1995), “Time Constrained Routing and Scheduling”, In *Handbooks in Operations Research and Management Science*, *Vol. 8. Network Routing*, M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser (eds), North-Holland, Amsterdam, 35–139.

Dorigo, M, G. Di Caro and L.M. Gambardella (1999), “Ant Algorithms for Discrete Optimization”, *Artificial Life* 50, 1–36.

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

Gehring, H. and J. Homberger (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, K. Miettinen, M. Mäkelä and J. Toivanen (eds), 57–64.

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.

Goldberg, D.E (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, New York.

Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

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

Kilby, P., P. Prosser, and P. Shaw (1999), "Guided Local Search for the Vehicle Routing Problems With Time Windows", In *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*, S. Voss, S. Martello, I.H. Osman and C. Roucairol (eds.), 473–486, Kluwer Academic Publishers, Boston.

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.

Or, I. (1976). Traveling Salesman-Type Combinatorial Problems and their Relation to the Logistics of Regional Blood Banking. Ph.D. Thesis, Northwestern University, Evanston, U.S.A.

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

Potvin, J-Y. and J.-M. Rousseau (1995), "An Exchange Heuristic for Routeing Problems with Time Windows", *Journal of the Operational Research Society* 46, 1433–1446.

Potvin, J-Y. and S. Bengio (1996), "The Vehicle Routing Problem with Time Windows Part II: Genetic Search", *INFORMS Journal on Computing* 8, 165–172.

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

Rochat, Y. and E. Taillard (1995), "Probabilistic Diversification and Intensification in Local Search for Vehicle Routing", *Journal of Heuristics* 1, 147–167.

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

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.

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

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.

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.

Thangiah, S. (1995a), "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.

Thangiah, S.R. (1995b), "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.

Wall, M. (1995), *GAlib - A C++ Genetic Algorithms Library, version 2.4*. (http://lancet.mit.edu/galib-2.4/), MIT, Boston.

APPENDIX I

The new best-known solutions for Solomon's (1987) data sets are specified as follows:

R108:
Routes: 9
Total Traveled Distance: 960.876
1. 73 72 75 56 23 67 39 55 25 54 26
2. 92 98 91 44 14 38 86 16 61 85 100 37
3. 27 69 1 53 40 21 4 74 22 41
4. 28 12 80 76 3 79 78 34 29 24 68 77
5. 50 33 81 51 9 35 71 65 66 20
6. 2 57 15 43 42 87 97 95 94 13 58
7. 6 96 59 93 99 5 84 17 45 83 60 18 89
8. 52 7 48 82 8 46 47 36 49 19
9. 31 88 10 62 11 64 63 90 32 30 70

R110:
Routes: 10
Total Traveled Distance: 1119
1. 2 41 22 74 73 40 53 26 54 24
2. 31 11 63 90 10 20 66 65
3. 52 82 8 18 7 48 46 45 60 89
4. 21 72 75 56 23 67 39 25 55 4
5. 59 98 44 16 86 38 14 43 42 13 58

6. 95 15 57 87 94 97 92 37 100 91 93
7. 27 69 30 51 9 71 35 34 78 33 1
8. 88 62 19 47 36 49 64 32 70
9. 28 76 12 29 81 79 3 50 77 68 80
10. 83 5 17 84 61 85 99 96 6

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

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

R210:
Routes: 3
Total Traveled Distance: 954.121
1. 95 92 42 15 23 67 39 75 22 41 57 87 99 6 94 53 40 21 73 72 74 56 4 55 25 54 26 58
2. 28 69 1 30 65 71 33 50 76 12 29 3 79 78 81 9 51 20 32 90 63 10 31 70 66 35 34 24 80 68 77
3. 27 52 7 47 36 64 11 62 88 18 45 16 44 14 38 86 61 84 5 60 83 8 82 48 19 49 46 17 85 98 37 97 13 2 43 100 91 93 59 96 89

R211:
Routes: 2
Total Traveled Distance: 906.192
1. 95 92 98 42 15 2 21 72 73 39 67 23 75 22 41 57 87 40 53 12 76 3 29 79 33 81 51 30 71 65 35 34 78 9 66 20 32 10 70 1 50 77 68 80 24 25 55 54
2. 28 27 69 31 52 83 5 61 16 44 14 38 86 85 99 18 82 7 88 62 19 11 90 63 64 49 36 47 48 46 8 45 84 6 94 96 59 93 37 97 13 58 26 4 56 74 43 100 91 17 60 89