2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

An Improved Particle Swarm Optimization Algorithm for Vehicle Routing Problem with Time Windows

Qing Zhu, Limin Qian, Yingchun Li and Shanjun Zhu
Department of Automation, Tsinghua University
Beijing, 100084, China
zhuqing04@mails.tsinghua.edu.cn

***Abstract*—Vehicle Routing Problem with Time Windows (VRPTW) is of crucial importance in today's industries, accounting for a significant portion of many distribution and transportation systems. In this paper, we present a computational-efficient VRPTW algorithm, which is based on the principles of particle swarm optimization (PSO). PSO follows a collaborative population-based search, which models over the social behavior of bird flocking and fish schooling. PSO system combines local search methods (through self experience) with global search methods (through neighboring experience), attempting to balance exploration and exploitation. We discuss the adaptation and implementation of the PSO search strategy to VRPTW and provide a numerical experiment to show the effectiveness of the heuristic. Experimental results indicate that the new PSO algorithm can effectively and quickly get optimal resolution of VRPTW.**

Keywords— Vehicle Routing Problem with Time Windows; Particle Swarm Optimization

I. INTRODUCTION

A. VRP and VRPTW

Vehicle routing problems (VRP) are omnipresent in today's industries, accounting for a significant portion of many distribution and transportation systems, including bank deliveries, postal deliveries, school bus routing, and security patrol services [1]. Its purpose is to design the least costly (distance, time) routes for a fleet of capacitated vehicles to serve geographically scattered customers. There may be some restrictions such as the capacity for each vehicle, total traveling distance allowed for each vehicle, time window to visit the specific customers, and so forth.

The vehicle routing problem with time windows (VRPTW) is a hard combinatorial optimization problem that has received considerable attention in the last decades. The time window variant of the problem imposes the additional constraint that each customer must be visited within a specific period of time. One can wait in case of early arrival, but late arrival is not permitted.

The objective function for this class of problem varies from one instance to the other. The primary objective might be to reduce the total travel distance, the number of vehicles, or even the time spent on the road. But most of the times these objectives, are considered simultaneously, in a hierarchical fashion.

B. Mathematical Formulation for the VRPTW

The VRPTW based on service classes will be stated as follows [2]. Let $G=(C,A)$ be a graph with nodes set $C=N\cup\{0\}$ and arc set A. The Set $V=\{1,2,...K\}$ denotes the overall vehicles, $N=\{1,2,...N\}$ represents the customer set and node 0 refers to the central depot. Each node $i\in N$ is associated with a customer demand $d_i(d_0=0)$, a service time $s_i(s_0=0)$, and a service-time window $[e_i,l_i]$. For every arc $(i,j)\in A$, a non-negative cost c_{ijk} for vehicle k and a non-negative travel time t_{ij} are known. Moreover, vehicles housed at the central depot are identical. Without loss of generality, each customer demand is assumed to be less than the vehicle capacity *q*. In addition, the demand of each customer cannot be split. The VRPTW is to find an optimal set of routes in such a way that

a) all routes start and end at the depot0 and end at node N+1;

b) each customer in N is visited exactly once within its service time;

c) the total of customer demands for each route cannot exceed the vehicle capacity q.

The service-time windows considered in this paper constitute "hard" constraints. That is, a vehicle cannot visit a customer beyond a specified latest arrival time. However, a vehicle can wait if it arrives too early. Our primary objective is to minimize the number of vehicles used, followed by minimizing the total distance traveled.

Our formulation is based upon the model defined by Solomon [3].

Define variable

$$x_{ijk}=\begin{cases}1, & \text{if vehicle } k \text{ travels directly from } i \text{ to } j\\ 0, & \text{otherwise}\end{cases}$$

s_{ik} : Vehicle *k* serves customer *i* at the time s_{ik}.

If vehicle *k* doesn't visit customer *i*, then s_{ik} has no meaning.

$$\min Z = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K} c_{ijk} x_{ijk} \quad (1)$$

Subject to:

$$\sum_{i=1}^{N}(d_i \sum_{j=1}^{N} x_{ijk}) \le q \quad \forall k \in V \quad (2)$$

$$a_i \le s_{ik} \le b_i \quad \forall i \in N, \forall k \in V \quad (3)$$

$$\sum_{k=1}^{K}\sum_{j=1}^{N} x_{ijk} = 1 \quad \forall i \in N \quad (4)$$

$$\sum_{j=1}^{N} x_{0jk} = 1 \quad \forall k \in V \quad (5)$$

$$\sum_{i=1}^{N} x_{ihk} - \sum_{j=1}^{N} x_{hjk} = 0 \quad \forall h \in N \quad \forall k \in V \quad (6)$$

$$\sum_{i=1}^{N} x_{i,n+1,k} = 1 \quad \forall k \in V \quad (7)$$

$$s_{ijk} + t_{ij} - K(1 - x_{ijk}) \le s_{jk} \quad \forall i, j \in N, \forall k \in V \quad (8)$$

$$x_{ijk} \in \{0,1\} \quad \forall i, j \in N, \forall k \in V \quad (9)$$

The objective is to minimize the total vehicle travel cost (1) subject to vehicle capacity, travel time and arrival time feasibility constraints. A feasible solution for the VRPTW services all the customers without the vehicle exceeding the maximum capacity of the vehicle (2) or the travel time of the vehicle (3). In addition, each customer can be served by one and only one vehicle (4). Equation (5)-(7) makes sure that each starts from node0 and ends at node *N*+1; (8)means that having served customer *i*, vehicle k still has enough time to catch up with customer *j*.

It is obvious that the VRPTW is NP-hard [4] due to the NP-hardness of VRP. The exact algorithms recently developed for solving the VRPTW can be found. However, on the 56 100-customer benchmark problems by Solomon, only a total of 11 problems were solved to optimality [5][6]. Because of the high computational complexity inherent in solving the VRPTW, solution methods in general resort to the heuristic design. Parallel route building algorithm [7] and GRASP heuristic [8] is some typical algorithms. In contrast to the classical route construction/improvement heuristics mentioned above, metaheuristics based on the methods such as tabu search [9], simulated annealing [10] and genetic search have shown better performance on average. It's undeniable that they do show effectiveness to some degree.

However, one of the disadvantages that they have in common is that the process is generally much too complicated. Compare the performance of PSO algorithm with well-known GA algorithm [11] for a number of randomly generated mapping problem instances and the results showed that the PSO algorithm solution quality is better than that of GA in most of the test cases. Moreover, the PSO algorithm runs faster as compared with GA.

II. PARTICLE SWARM OPTIMIZATION (PSO)

Particle Swarm Optimization (POS), a new heuristic global optimization technique motivated from the simulation of social behavior was originally designed and developed by Eberhart and Kennedy in 1995. It is based on swarm intelligence and each particle of the swarm represents one candidate solution of the problem being optimized. The algorithm finds optimal regions of complex problem spaces through the interaction of particles. In PSO, instead of using genetic operators, each particle (individual) adjusts its "flying" according to its own flying experience and its companions' flying experience. In [11], the maximum velocity on the performance of the particle swarm optimizer was first analyzed and the guidelines for selecting parameters are provided. The impact of the inertia weight was firstly introduced in [12] and the efficiency of PSO has improved greatly ever since. Fuzzy rules were used to determine the value of weight in [13].

Each particle is treated as a point in a D-dimensional space and the number of all particles is i. The i th particle is represented as $X_i = (X_{i1}, X_{i2}, ..., X_{iD})$. The best previous position (the position giving the best fitness value) of the i th particle is recorded and represented as $P_i = (p_{i1}, p_{i2}, ..., p_{iD})$. The index of the best particle among all the particles in the population is represented by the symbol g. The rate of the position change (velocity) for particle i is represented as $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$. The particles are manipulated according to the following equation:

$$v_{id}(t+1) = w \times v_{id}(t) + c_1 \times rand() \times [p_{id}(t) - x_{id(t)}(t)] + c_2 \times Rand() \times [p_{gd}(t) - x_{id(t)}(t)] \quad (10)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad 1 \le i \le n; 1 \le d \le D \quad (11)$$

Where c_1 and c_2 are two positive constants, rand() and Rand() are two uniform random number generation functions in the range [0,1], and w is the inertia weight. Equation (10) is used to calculate the particle's new velocity according to its previous velocity and the distances of its current position from its own best experience (position) and the group's best experience. Then the particle flies toward a new position according to equation (11). The performance of each particle is measured according to a

pre-defined fitness function, which is related to the problem to be solved. A larger inertia weight w facilitates global exploration (searching new areas) while a smaller inertia weight tends to facilitate local exploration to fine-tune the current search area. Suitable selection of the inertia weight w can provide a balance between global and local exploration abilities and thus require less iteration on average to find the optimum. The distance and velocity dynamic range of the element d in each particle is $[-XMAX_d, XMAX_d]$ and $[-VMAX_d, VMAX_d]$ respectively. If the sum of accelerations would cause the velocity on that dimension to exceed the range, which are specified by the user, then the velocity on that dimension is limited to $\pm XMAX_d$ or $\pm VMAX_d$.

III. Particle Swarm Optimization for VRPTW

A. Particle Definition and Solution Expression

In this section, we describe the formulations of the PSO algorithm for VRPTW. In this paper, we considered homogeneous systems where the processor computing powers are the same. In addition, we assume that processors directly connected with each other. We apply the PSO as if they were searching a continuous space, and then round the positions (routes) to integer values.

One of the key issues in designing a successful PSO algorithm is the representation step, i.e. finding a suitable mapping between problem solution and PSO particle. In our work, we set up a new particle coding for the VRPTW problem, which helps convert the discrete combinational problem into continuous problem so that the PSO algorithm can be directly applied.

We assume a VRPTW problem with n customers and k available vehicles for delivery, and there are no differentiations among vehicles. For every possible solution, it can be represented as a series of the n customers with k+1 center nodes. For example, in a 3-vehicles and 6-customers problem, a possible route can be shown as:

Vehicle 1: 0 – 5 – 0
Vehicle 2: 0 – 4 – 3 – 6 – 0
Vehicle 3: 0 – 2 – 1 – 0

By eliminating the overlapped node 0, we can combine the series in to one number sequence, which is called the route sequence:

Route sequence: 0 – 5 – 0 – 4 – 3 – 6 – 0 – 2 – 1 – 0

It can be seen that each route sequence must start with node 0 and end with 0 as well. Therefore, the two 0 nodes can be reduced without any confusion between the routes. Thus, the basic route can be presented as 5 – 0 – 4 – 3 – 6 – 0 – 2 – 1. There is one situation that should be pointed out: If two 0 nodes are neighbors, it means that there is a vehicle leaving and returning without reaching any customers, which means that the vehicle is not assigned any delivery job.

In general, a route for an n customers and k vehicles VRPTW problem can be presented as a n+k-1 dimension vector with each dimension defining the sequence of the corresponding customers or the center node. In designing the PSO algorithm, we also define a real-number n+k-1 dimension particle, which can be normalized into integer index as follows:

Node : 1 2 3 4 ····· n 0 0 ··· 0
Index : 5 i j 9 ····· 3 1 7 ··· 8

For example, also with the 3-vehicles and 6-customers problem, we can get

Node : 1 2 3 4 5 6 0 0
Index : 8 7 4 3 1 5 2 6

With this definition, each particle can represent a route for the VRPTW problem.

B. Description of the PSO for VRPTW

In fact, the VRPTW problem is a combinatorial optimization problem with constraints. These constraints (2) ~ (9) must be met by the solutions. In this paper, some of the constraints are automatically satisfied because of our particle definition and the solution expression. However, the constraint (2) that the total request of each vehicle on its route cannot exceed its own capacity and the constraint of the time window for the vehicle to arrive have to be included in the consideration. Therefore, we will use large positive numbers R, S, and T as the punishment coefficients to deal with these constraints. With these coefficients, these constraints can be incorporated into the object function as follows:

$$\min Z = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K} c_{ijk}x_{ijk} + R\cdot\sum_{i=1}^{N}\max\left((d_i\sum_{j=1}^{N}x_{ijk} - q), 0\right) + S\cdot\sum_{i=1}^{N}\max\left(a_i - s_i, 0\right) + T\cdot\sum_{i=1}^{N}\max\left(s_i - b_i, 0\right) \quad (12)$$

If R, S, and T are large enough, those infeasible solutions would attain very large fitness value. During the PSO algorithm iterations, the particles will move to the feasible solution sets.

The details of the PSO algorithm for the VRPTW problem are explained below:

Step 1 Particle initialization: Initially the particles are assigned a random number between 1 and n+k-1 for each dimension of the particle position vector X. Assign a random number between –(n+k-2) ~ (n+k-2) for each dimension of the velocity vector V.

Step 2 Normalization: Normalize the particles into index sequence so that each particle can be interpreted as a solution of the route.

Step 3 Evaluation: Evaluate every particle according to (12). After evaluation, each particle saves its historical

optimal solution as *pbest* and then the global optimal one *gbest* can be retrieved from all the particles. For the initial situation, choose the initial value for the *pbest* and *gbest*.

Step 4 Update: Each particle is updated according to (11) with the new velocity vector from (10). Do the normalization as well for the newly generated particles.

Step 5 Judgement: if the termination condition is not met, then continue with step 3.

IV. COMPUTATIONAL ANALYSIS

A. Case One

In order to test the effectiveness of the algorithm, we apply the PSO to the following VRPTW problem and compare its performance with that of the improved GA program in [14].

Assume that there is an 8-customer service system with each customer demand d_i (i = 1,2, 3… 8). The service time and the service time window $[e_i\ l_i]$ of each customer are shown in Table Ⅰ. Three vehicles with a capacity of 8 will do all of the jobs. The distances between the center and each customer and the distances among the customers are presented in Table Ⅱ. The vehicle average speed and its unit cost are supposed as 50 and 1 respectively.

In our computational effort, we find the optimum parameters for the PSO algorithm by trial and error, which can be found in TableⅢ.

We run the PSO and GA algorithm 100 times separately. The result shows that the rate of reaching the best solution (TableⅣ) is 82% for the PSO algorithm, much higher than the 36% of the GA algorithm. Moreover, since it involves only the basic computational operations, the PSO algorithm gets the best solution much faster than the GA algorithm. A good process of the best particle in the swarm particle fitness value evolution is illustrated in Fig. Ⅰ.

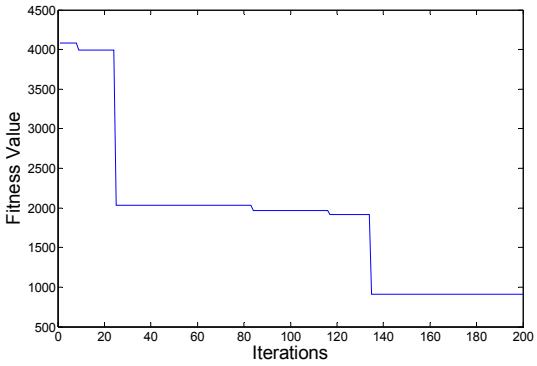

Figure 1. Particle Fitness Evolution

B. Case Two

In this case, we randomly generate several VRPTW problems to test the effectiveness of the PSO algorithm. The customers are distributed on the range of [-100, 100] coordinates with their demands between [1, 5]. The results show that PSO algorithm can achieve most of the best solutions with comparative performance of the other algorithm.

Several remarks on designing the parameters of the PSO for the VRPTW problems can be attained from these tests.

- **Inertia weight *w*:** according to [15], adaptively modifying the inertial weight w in (10) which balances

TABLE I. CUSTOMERS DEMAND AND SERVICE TIME

Index	1	2	3	4	5	6	7	8
Demand	2	1.5	4.5	3	1.5	4	2.5	3
S	1	2	1	3	2	2.5	3	0.8
$[e_i\ l_i]$	[1, 4]	[4, 6]	[1, 2]	[4, 7]	[3, 5.5]	[2, 5]	[5, 8]	[1.5, 4]

TABLE II. DISTANCE BETWEEN THE NODES

Di	0	1	2	3	4	5	6	7	8
0	0	40	60	75	90	200	100	160	80
1	40	0	65	40	100	50	75	110	100
2	60	65	0	75	100	100	75	75	75
3	75	40	75	0	100	50	90	90	150
4	90	100	100	100	0	100	75	75	100
5	200	50	100	50	100	0	70	90	75
6	100	75	75	90	75	70	0	70	100
7	160	110	75	90	75	90	70	0	100
8	80	100	75	150	100	75	100	100	0

TABLE III. PSO ALGORITHM PARAMETERS

C_1	C_2	R	S	T	ω	Iterations	Num. of Particles
3	2	1000	1000	1000	1	200	200

TABLE IV. BEST SOLUTION

Vehicle	Route
Vehicle 1	0 – 3 – 1 – 2 – 0
Vehicle 2	0 – 6 – 4 – 0
Vehicle 3	0 – 8 – 5 – 7 – 0

the global exploration and local exploitation can overcome the problem of local trapping in the PSO convergence. However, the modifying of the inertia weight w can seldom affect the searching result in our tests, because the VRPTW is basically a combinational optimization problem.

- **Number of the particles:** in order to avoid local optimum convergence, the number of the particles must be increased according to the size of the service. Basically, for a median size of the customers, the number of the particles should be no less than 10 times of the customers.
- **Number of iterations:** when the number of the particles reaches a certain level, the increase of the iterations can have little contribution to the improvement of the solution.

V. CONCLUSIONS

In the industrial business today, the optimal selection of the vehicle route is a key issue to improve the service quality, reduce the operation cost and increase the profits. In this paper a computational-efficient PSO algorithm is introduced to solve the VRPTW problem. The particle expression and the approach of the whole algorithm are detailed. The effectiveness of the algorithm is demonstrated by the experimental results.

REFERENCES

[1] Blanton (Jr), J. L. and Wainwright, R. L. "Multiple vehicle routing with time and capacity constraints using genetic algorithms." Proceedings of the Fourth International Conference on Genetic Algorithm, 1991, pages 452-459.

[2] Fuh-Hwa Franklin Liu, Sheng-Yuan Shen, "A route-neighborhood-based metaheuristic for vehicle routing problem with time windows", European Journal of Operational Research 118, 1999, pp.485-504.

[3] Solomon, Marius M., "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints". Operations Research 35 (2), 1987, pp. 254-265.

[4] Lenstra, J., A. H. G. Rinnooy Kan. "Complexity of vehicle routing and scheduling problems". Networks 11, 1981. pp.221–227.

[5] M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh, F. Soumis, "Vehicle routing with time windows: Optimization and approximation", B.L. Golden, A.A. Assad (Eds.), Vehicle Routing: Methods and Studies, North-Holland, Amsterdam, 1988, pp.65~84.

[6] J. Desrosiers, Y. Dumas, M. Solomon, F. Soumis, "Time constrained routing and scheduling", M. Ball, T. Magnanti, M. Monma, G. Nemhauser (Eds.), Handbooks in Operations Research and Management Science, vol. 8: Network Routing, North-Holland, Amsterdam, 1995, pp. 35~139.

[7] J. Potvin, J. Rousseau, "A parallel route building algorithms for the vehicle routing and scheduling problem with time windows", European Journal of Operational Research 66, 1993, pp.331~340.

[8] G. Kontoravdis, J.F. Bard, "A GRASP for the vehicle routing problem with time windows", ORSA Journal of Computing 7, 1995, pp.10~23.

[9] Andrew Lim and Fan Wang, "A Smoothed Dynamic Tabu Search Embedded GRASP for m-VRPTW", Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, 2004, pp.1082-3409/04.

[10] Zbigniew J. Czech, Piotr Czarnas, "Parallel simulated annealing for the vehicle routing problem with time windows", Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing (EUROMICRO-PDP.02), 2002, pp.1066-6192/02

[11] Kennedy J, Eberhart R C, Shi Y. Swarm Intelligence. San Francisco: Morgan Kaufman Publishers, 2001.

[12] Shi Y, Eberhart R C. "A modified particle swarm optimizer". Proceedings of the IEEE International Conference on Evolutionary Computation, 1998, pp.69-73.

[13] Shi Y, Eberhart R C. "Fuzzy adaptive particle swarm optimization". Proceedings of the IEEE Congress on Evolutionary Computation, 2001, pp.101-106.

[14] ZHANG Liping, CHAI Yue-ting, CAO Rui, "Improved Genetic Algorithm for Vehicle Routing Problem with Time Windows", Computer Integrated Manufacturing Systems, Vol. 8, No. 6 Jun., 2002

[15] Eberhart R.C., Shi Y. "Particle swarm optimization: developments, applications and resources", Proc. Congress on Evolutionary Computation 2001. Piscataway, NJ: IEEE Press, 2001, pp81- 86.