EXPERIMENTS TO DETERMINE THE SIMULATED ANNEALING PARAMETERS CASE STUDY IN VRP

Babak Sohrabi

Department of Industrial Management, University of Tehran
Tehran, Iran, bsohrabi@ut.ac.ir

M.H. Bassiri

Amirkabir University of Technology
Tehran, Iran

(Received: August 6, 2003 - Accepted in Revised Form: February 26, 2004)

Abstract In this paper we describe the distribution problems faced by one of the largest retailers in the distribution and sale of fast moving consumer goods in the U.K. The paper describes an initial solution method, which is then improved by a novel form of simulated annealing. A computational experiment for the improvement algorithm has been also carried out in order to illustrate the flexibility of the computer programming and to demonstrate how it can be used to address broader management issues.

Key Words Vehicle Routing Problem, Simulated Annealing, Computational Experiment

چکیده در این مقاله، مساله توزیع و فروش یکی از بزرگترین شرکتهای عرضه کننده مواد غذایی در انگلستان مورد مطالعه قرار گرفته است. روشی برای بدست آوردن یک جواب اولیه شدنی قابل قبول و سپس بهبود آن توسط یک شکل جدید از یکی از الگوریتم های هوش مصنوعی (SA) توصیف شده است. همچنین آزمایش محاسباتی برای الگوریتم بهبود به منظور نشان دادن انعطاف پذیری و اثبات اینکه چگونه برنامه فوق می‌تواند مورد استفاده مدیران قرار گیرد، انجام شده است.

1. INTRODUCTION

The transport of goods or people from one place to another is a very important economic activity that is usually known as distribution. Vehicle routing problems arise in many companies, where goods must be distributed from suppliers to customers. These problems have been extensively studied, especially in the last decades. The reason for their receiving so much attention is that it is simultaneously very important for the economy of the firms and very interesting for operational research (OR) scientist. The basic vehicle routing problem can be defined as follows: Given a depot and a number of customers with known geographical locations and demands, the VRP (vehicle routing problem) consists of determining set routes that minimize the total distance traveled, whilst satisfying all the customers' requirements. A route consists of a sequence of visits to customers, starting and ending at the depot. The total demand of all the customers on a route must not exceed the vehicle capacity. Each route, which may also have limited total distance or duration, is assigned to one vehicle, all of which are assumed to be identical, and each customer is visited exactly once.

Real world problems are generally quite different from the basic vehicle routing problems dealt with in the literature [1], as the context of the problem generates a wide variety of constraints, such as the numbers of vehicles and drivers, different commodities with different time windows for customers, which may vary in duration. The number of constraints that must be considered is often large. Moreover, the real characteristics of the distribution problems vary from one company to another.

The following characteristics of a practical problem are examples that may make it different from the basic VRP:

- Dealing with both pickup and delivery,
- Delivery of different types of commodity,
- Loading constraints that may depend on the type of commodity,
- Pickup and delivery from a central hub depot,
- Time window constraints, which may depend on the type of commodity, and
- Multiple trips in one vehicle route.

2. AN OVERVIEW TO THE REAL DISTRIBUTION PROBLEM

Developments in vehicle routing have been driven by a need to obtain solutions to ever more complex real-world problems and then to improve upon those solutions. Thus additional complications, like having multiple depots and vehicles of different sizes, have been considered. However, none is quite as multifaceted as the logistics problem faced by the UK's major superstore operators; Asda, Safeway, Sainsbury and Tesco. Although now increasingly selling durables all have evolved from grocery retailers with operations as described below.

Suppliers do not normally deliver directly to any of the hundreds of stores. Instead, each must deliver to a regional distribution center (RDC), of which there are likely to be between 10 and 20, at times predetermined by the retailer. However, not all RDCs will stock every product; for example, medium-moving grocery products will only be kept at a limited number of 'central' locations. Then the basic problem is to schedule deliveries from a RDC to the retail stores in its area, if necessary picking up goods en route from a central warehouse, generally referred to as a hub depot. Whilst a superstore retailer will stock thousands of different products, in bar code terms, there are basically five product categories for distribution.

Grocery items only stocked at a hub depot, from which deliveries must be made directly to some retail outlets, Grocery items available at a RDC for delivery to the stores, Produce, such as is sold by the traditional greengrocer, Perishable goods, like dairy products, Frozen products, which are predominantly foods.

The individual items will have been picked in the warehouse to fulfill a store's order and stacked on pallets A pallet may contain different products from the same category but products from different categories will not be on the same pallet. The delivery scheduler is only interested in the number of pallets of each category required by a store and is unconcerned about the product mix on the pallets.

At the RDC, grocery items are stocked at the ambient temperature, produce and perishable goods are kept in a chilled environment and frozen products are maintained below zero. The delivery vehicles, which are loaded from the rear, can be divided into three compartments, kept at different temperatures during a trip. The front compartment, loaded first and unloaded last, is for pallets of frozen products. The middle section is for both produce and perishable goods. The back compartment, unloaded first, is for grocery items, irrespective of whether loaded on to the vehicle at the RDC or the hub. Because the sizes of the compartments are totally flexible, a full vehicle could carry one, two or all three of these product types. However, rear end loading and unloading precludes total flexibility in terms of the mix of product types carried by a vehicle, if it is to deliver to more than one store. For example, a vehicle could not make one drop of grocery and frozen products at one store and produce or perishable at a second store. Instead, it would have to deliver grocery to the first store, produce or perishables to the second store and then return to the first store with the frozen products Similarly, a vehicle partly laden with pallets from the RDC could not travel to the hub and fill up with pallets from there, if the first delivery was for only RDC loaded goods. Thus the product type mix of pallets for the various stores is a significant determinant of the vehicle delivery schedule, not covered in the VRP literature.

Because of their importance, time windows was the first complexity introduced into the VRP In practice, there are both hard and delivering to soft time windows, with hard time windows not to be violated at any price However, soft time windows allow a vehicle to arrive early or late but a penalty, depending on the degree of violation, is incurred

for doing so. Suppliers the RDCs complain that time windows are hard and narrow. However, since the vehicle scheduler and the person responsible for inward goods at the stores work for the same company, time windows for non-urgent items can be negotiated on the telephone to improve delivery schedules As is immediately apparent, this favors existing schedules, when making comparative tests of any new scheduling method.

The VRP literature distinguishes between backhauls, which is characterized by picking up product for delivery to the depot after all the outgoing deliveries have been made, and the so-called pickup and delivery. In the latter case, goods are collected from pick-up points on a route and dropped at delivery points on the same route; transportation of the infirm has this structure. Superstore vehicle schedulers face both these problems to different extents. The most frequent type of collection is the return from stores to any depot of salvage, which is packaging materials, pallets etc This can only be undertaken after all the product deliveries have been made. Products are also collected from suppliers, most usually for delivery to the depot where the vehicle is based, but more complex backhauls are possible. For example, a collection may be made from a supplier, after delivering to a nearby store, and trucked to a distant depot, before returning to its base, making an inter-depot transfer in the process As well as the limit to a vehicle's pallet capacity, the driver's time is also regulated, in terms of both the length of the driving day and the inclusion of rest periods. The vehicle schedules must include allowances for loading and unloading that depend on the number of pallets involved and preparation times, which are independent of the number of product categories.

The collaborating retailer uses a commercially available routing and scheduling package to produce a base schedule for morning (AM) and afternoon (PM) deliveries for each day of the week [2] The base schedule is updated every six months, because the software is not designed to handle complex routing and is insufficiently fast to route initiate for each shift The scheduler at a depot arrives at 6.00 am to begin scheduling the PM deliveries Up-to-date grocery demands will be known, whilst those for produce, perishable and frozen products will become known between 7.00 am and 8.00 am, as will backhauls for both the PM and the following AM shifts The scheduler uses this information to update the demands in the base PM schedule and modify routes. The commercial software is used at this stage to calculate the times of modified routes and to prevent the scheduler violating either the vehicle capacity or hard time window constraints Obviously if some demands have been increased, then the scheduler may be unable to make complex adjustments, resulting in split-loads. This process is completed by 9.30 am and the printed schedule given to the warehouse for loading. At 10 am, the scheduler starts the same process for the next day's AM schedule, except that now up-to-date grocery information is missing Nevertheless, a schedule is printed at 1 pm and retained until 6 pm, when the required grocery data arrives. The AM schedule is then updated manually and passed to the warehouse for loading the vehicles, which will leave the RDC at regular intervals from 3.30 am onwards.

3. THE STRUCTURE OF THE ALGORITHM

Quite simply, an algorithm is required to solve the real-world scheduling problem at the lowest possible cost and sufficiently quickly to be used daily for both the AM and PM schedules, without the need to make adjustments to a predetermined skeletal schedule The approach described in this paper involves using a meta-heuristic, which starts with an initial solution that is improved iteratively. Whilst Gendreau, et al. [3] consider tabu search to be the most effective approach, their conclusion was drawn from CVRPs, where the complexity is less, as far fewer constraints must be included and is therefore very different from the problem of the grocery superstore retailers Simulated annealing was chosen as the basis for the heuristic as it has performed well on complex problems and for its ease of coding.

A trip is defined to be a journey starting and finishing at the RDC, whilst a route consists of all the trips made by one vehicle within the schedule. An initial feasible solution is obtained with 3 simple steps that construct vehicle routes sequentially.

1. **Starting a New Trip or Route** If there are still demands to be met, whether a new trip within the present route or a new route is started depends on the remaining driving time. Note that all trips start and finish at the RDC and all demands are either to be picked up from the RDC or the hub or they are backhaul Add the closest customer not already scheduled to the RDC to a new trip for an existing vehicle route or start a new route, as is appropriate, and go to Step 2.
2. **Adding a Demand to the Current Trip** Unless all demands have been scheduled, find the one nearest to the vehicle's current location and go to the appropriate subroutine for adding a customer. Repeat until either a customer has been added or none of the unscheduled demands can be included in the trip. Then go to Step 3.
3. **Continuing to Schedule Demands** Go to Step 1 if either the unscheduled demands are unsuitable for the present trip or there is insufficient time left on the route for another demand to be scheduled Otherwise, go to Step 2.

The subroutines for adding a customer distinguish between whether the demands are supplied from the RDC or the hub or are backhaul, as follows.

3-1. Demands Supplied From The RDC

If the first demand is for more than 19 pallets of the same product type, one trip is assigned, because the vehicle will be almost full. If the first demand is for fewer pallets, a new trip is started unless the category is frozen, when the next demand is considered instead. This is because frozen products are placed at the front of the vehicle but the first scheduled demand is at the rear. When the first customer requires more than one category, all those with overlapping time windows are considered. The capacity and driving time constraints are checked for feasibility. The trip terminates if the first customer (or any other) has salvage to be returned to the RDC or hub. Otherwise, additional customers are considered for inclusion with a customer having demands for *n* product categories being taken to be *n* customers with demands in time window order. Category compatibility and time windows are checked and acceptable demands included in the trip.

3-2. Demands Supplied From The Hub

As the hub only stocks grocery, loading feasibility is excluded from the constraints. A vehicle will be routed via the hub when customers are to be supplied from it.

3-3. Backhauls

The time window of a backhaul is checked before its insertion in a trip When the backhaul is delivered to the RDC, a new trip may be started. If it is delivered elsewhere, such as to another supplier or RDC, and if sufficient driving time is available, then the trip may be continued by returning to Step 2., In obtaining an initial solution, the number of vehicles is not limited but hard time windows are used.

The improvement algorithm is essentially standard simulated annealing [4], except that a given number (M) of best-improve moves are made at each of the (N) iterations with the same temperature, before consideration is given to the probabilistic acceptance of a worse solution. Following Osman [5], the λ-interchange generation mechanism is used to explore the neighborhood structure. For deliveries from the RDC, $\lambda = 1$ is used, so that two customers in different trips may be swapped or a customer may simply be shifted from one trip to another. All the product categories to be delivered at a drop are moved. However for deliveries from the hub, the pick-up from the hub and the delivery of those goods are not allowed to be moved independently (i.e. $\lambda = 2$), because the hub and the customer requiring the hub groceries must be in the same trip. Similarly for a backhaul, both its origin and destination must be moved together. The simulated annealing stops when a prescribed number of cycles of N iterations have been executed without changing the value of the best solution. The costs of distribution, which change with the schedule, are the tractor, trailer and driver fixed costs and the variable cost per mile Thus when a customer is removed from a route, there is a cost reduction equal to that for one vehicle, if the route contained one customer, and zero otherwise In addition, there will be a decrease in the mileage cost. The converse applies when a customer is added to a route.

In obtaining an initial solution, it is assumed that the vehicle capacity is fixed and that time windows are hard. In the real world, the scheduler can always load a few extra pallets on to a nominally full vehicle and the time windows are

soft. Hence the improvement algorithm includes penalty costs for overloading, expressed as pounds per additional pallet, and time window penalties in pounds per hour waiting or late at a customer Any changes in these penalty costs must be included with the vehicle and mileage cost changes, whenever a customer is removed from or added to a route Obviously the change in the capacity penalty, which is usually zero, depends on the vehicle capacity and the number of pallets on the vehicle before and after the customer removal or addition. However, the removal or addition of a customer may alter the time window penalties for late customers in the route. Thus the changes summed over those customers must be calculated. When a customer is added to a trip, loading feasibility has to be checked against the product categories of the previous and following customer. This is unnecessary for a removal.

A trip with only one customer is not destroyed if that customer is removed during the simulated annealing, because it provides an opportunity to insert another customer. In practice, it is expected that the improvement algorithm will reduce the number of vehicles, unless the initial solution is very good.

The λ-interchange mechanism could be extended but it becomes complex for sets of customers Instead, it can very easily be applied to complete trips in routes, instead of customers, with $\lambda = 1$. No costs will change, except the time window penalties, and the only constraint is the length of the driver's day.

4. SETTING VALUES FOR THE SIMULATED ANNEALING PARAMETERS

In any application to a particular combinatorial optimization problem, we must take a number of decisions on choices. These choices fall into two classes according to Johnson et al. [6]: problem specific choices and generic choices for cooling schedules. In order to experiment with the parameters, we need to work on the generic choices, which are as follows:

- Generic choices define the components of the cooling schedule. A cooling schedule must give specific answers to the following questions on how to determine:

1. The initial starting value of the temperature T.
2. The number of iterations to be performed at each temperature.
3. The cooling rate and the temperature update rule. The termination of the algorithm (stopping criterion).

As was stated before, there are four important parameters that have to be considered in this algorithm in order to escape from a local optimal solution. These parameters are as follows:

- **T: The Initial Temperature** In the process, the temperature (T) descends slowly through a series of levels. The value of the initial temperature determines the initial probability of accepting an uphill move. If it is set too low, the algorithm may not be able to move to other regions of the solution space and may become permanently trapped in a local optimum. If it is set too high, uphill steps may be accepted over many iterations during early iterations and computing time may be wasted before the probability of accepting an uphill step is significantly reduced. There are various more sophisticated methods for setting the initial temperature value, but we are simply using experimentation to find a suitable value.

- **N: The Number of Iterations at Each Temperature** When the parameters receive a value, the algorithm starts to work. As stated above, T gets a suitable value in order to escape from the local optimal solution. So with first value of T, the first iteration starts. The modification made to the standard simulated annealing algorithm ensures that the entire neighborhood has been searched within each iteration. The algorithm in this paper performs M trials, taking the best improvement at each one, so for this implementation the value of N does not need to relate to the size of the neighborhood. In all the runs, M is set to 100. The number of iterations at each temperature affects the rate of cooling. The higher the value of N, the slower the cooling will be. Generally, higher values of N may lead to better quality solutions, but at the expense of a longer computation time.

TABLE 1. Experiment with Different Values of the Different Parameters (T = 400).

T	N	r	Sc	Cost	No. its	CPU
400	200	0.99	100	3266.86	21057	821
400	200	0.99	50	3266.86	19687	767
400	200	0.90	100	3401.20	9939	387
400	200	0.90	50	3401.20	8569	334
400	100	0.99	100	3266.86	21057	821
400	100	0.99	50	3266.86	19687	767
400	100	0.90	100	3482.70	6339	247
400	100	0.90	50	3482.70	4969	193

• **r: The Temperature Update Rule** As stated previously, the rapid quenching process can be viewed as analogous to local optimization via steepest descent. In this implementation, simple geometric cooling is used where at each temperature change, the current temperature is multiplied by a constant, r where $0 < r < 1$. The closer the value of r to 1, the slower will be the cooling and so may produce better quality solutions, but at the expense of longer computation time.

• **Sc: The Stopping Criterion** The stopping criterion, which has been used in this algorithm, represents the number of iterations, which have taken place with no change in the value of the best solution.

After some initial experiments (results not recorded) testing the effect of different values, a series of experimental runs was carried out where each parameter was given two possible values in the region of what was expected to be a suitable value. The Tables 1 and 2 show the different values for the parameters. All the experiments have been carried out using a PC Pentium 3. In both tables:

T: initial temperature
N: number of iterations
r: the temperature update rule
Sc: stopping criteria
Cost: total cost
No. its: total of number of the iterations
CPU: total time in second

The above tables show the best cost, which is £3266.86, is obtained with different parameter values. This result appears eight times in two tables. There are five following points: After the initial experiments with parameter values, the results are not too sensitive to the values used in these experiments.

The same cost resulted whether stopping criteria (Sc) was set to 50 or 100. Therefore there is no need to wait for 100 iterations without improvement before stopping; 50 iterations without improvement will be sufficient.

When t is 0.99, the best result of £3266.88 is

TABLE 2. Experiment with Different Values of the Different Parameters (T = 2000).

T	N	r	Sc	Cost	No. its	CPU
2000	200	0.99	100	3266.86	21057	821
2000	200	0.99	50	3266.86	19687	767
2000	200	0.90	100	3319.70	12939	504
2000	200	0.90	50	3319.70	11569	451
2000	100	0.99	100	3266.86	21057	821
2000	100	0.99	50	3266.86	19687	767
2000	100	0.90	100	3401.20	7839	305
2000	100	0.90	50	3401.20	6469	252

always achieved, but when t is set to 0.9 a worse result is always achieved, so t should be set to 0.99

When r is 0.99 the best cost and total number of iterations (No. its) are not affected by the value of T and N used. For the later experiments, we decided, to use T = 400 and N = 100.

The similarity in the result for different values proposed for the simulated annealing parameters arises because all runs are started from the same initial solution, the same neighborhood moves are tried, any reduction in cost is always accepted and when there is an increase in cost, the probabilities of acceptance implied by these parameter values are similar.

4.1 Changing The Time Windows One of the other experiments is changing the time windows for the customers. The reason for this is because after discussing the previous results with the company, they suggested that the grocery time window should be tighter. In order to do that, we kept the same time windows for perishable and produce, which was one hour, and for frozen, which was two hours. But we decreased by one hour the latest time for deliveries of grocery. Runs were again carried out with different values of the simulated annealing parameters to check whether the conclusions from the previous section were still valid. The results appear in the Tables 3 and 4.

The tables show that the results have a similar sensitivity to the values of the simulated annealing parameters as in the previous section. As the results show, again the best result was found when T = 400, N = 100, r = 0.99 and Sc = 50. This confirms the choice of parameter values, which will now remain fixed for other experiments. As expected, the cost increased because the time window was decreased for grocery. This made some deliveries in the previous solution infeasible. So this set of experiments shows two important points: The best cost was found again using the same values assigned to the parameters as for the previous experiments with the original time windows for grocery. The best solution cost increases when a time window is decreased.

4.2. Changing the Capacity of The Vehicles An important issue for the management of any

TABLE 3. Experiment with Different Time Windows (T = 2000).

T	N	r	Sc	Cost	No. its	CPU
2000	200	0.99	100	3353.45	21451	836
2000	200	0.99	50	3353.45	19705	768
2000	200	0.90	100	3377.63	13077	510
2000	200	0.90	50	3377.63	11595	452
2000	100	0.99	100	3353.45	21181	826
2000	100	0.99	50	3353.45	20029	781
2000	100	0.90	100	3459.13	7933	309
2000	100	0.90	50	3459.13	6735	262

distribution operation is deciding on the size and type of the vehicles used. One use of the computer programming is to examine the sensitivity of the distribution cost to the capacity of the vehicles. In the following experiment, the capacity of the vehicles was increased from 21 to 24 pallets.

After finding the best values for the parameters which are T = 400, N = 100, r = 0.99 and Sc = 50, we set up the parameters according to those values. The time windows used were the same as for Table 5.

After increasing the capacity, the number of vehicles is still 12 and the distance was the same. There was a change just in the penalty function for capacity. These results show that a simple assumption that distribution costs are inversely proportional to the capacity of the vehicles used can be very misleading. This program can be used to investigate the relationship between vehicle capacity and distribution costs more precisely.

4.3 Different Demands In all the above experiments the demand of the customers was the same. But in this experiment, we changed the demand of the customers to represent the fluctuations, which may occur on different days, and the result is shown in table 6. In this case, we tried different values for the parameters and again the best result was found using T = 400, N = 100, r = 0.99 and Sc = 50.

4.4 Relaxing Time Window Constraints As was stated in Section 2, company has divided its customers into two parts: AM and PM. Delivery for AM is done between 3 a.m. and 3 p.m. and for PM between 3 p.m. and 3a.m. In this experiment, we used the algorithm to examine the effect of removing the time windows. In other words we assigned to all the customers the same time window, which was from 3 a.m. until 3 p.m., and as it is shown in Table 7 the number of vehicle required is 10.

The results show if the time windows become wider, there is a big difference in terms of the cost and number of the vehicles. It gives an indication

TABLE 4. Experiment with Different Time Windows (T = 400).

T	N	r	Sc	Cost	No. its	CPU
400	200	0.99	100	3353.45	21443	836
400	200	0.99	50	3353.45	19700	767
400	200	0.90	100	3459.13	10061	392
400	200	0.90	50	3459.13	8685	338
400	100	0.99	100	3353.45	21327	831
400	100	0.99	50	3353.45	19683	765
400	100	0.90	100	3540.63	6375	248
400	100	0.90	50	3450.63	5189	202

TABLE 5. Changing Capacity from 21 to 24.

• T	• N	• r	• Sc	• cost	• No. its	• CPU	• Cap
• 400	• 100	• 0.99	• 50	• 3214.86	• 19687	• 767	• 24

TABLE 6. Changing Demand.

• T	• N	• r	• Sc	• cost	• No. its	• CPU	• Cap
• 400	• 100	• 0.99	• 50	• 3316.51	• 19532	• 732	• 21

TABLE 7. Experiment with Different Time Windows (T = 400).

• T	• N	• r	• Sc	• Cost	• No. its	• No.vehicle
• 400	• 100	• 0.99	• 50	• 2772.01	• 19687	• 10

to the management of the scale of cost reduction, which is possible if a way can be found to relax time window constraints.

5. CONCLUSION

A particular distribution problem has been studied

in this paper which is typical of that faced by companies operating supermarkets and other types of shop. The program runs quickly enough, so that it can be used on a daily basis using the most recent demand figures, instead of planners having to manually update a base schedule. The program is designed to be user friendly so can be used by a scheduler and is very flexible in terms of being able to change time windows, demands and other inputs.

Further computational experiments illustrate the flexibility of the computer programming and demonstrate how it can be used to address broader management issues (such as the capacity of the vehicles to be used), as well as the daily scheduling.

6. ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewer for their helpful comments on the earlier version of this paper.

7. REFERENCES

1. Golden, B. L. and Assad, A. A. "Vehicle Routing: Methods and Studies", Edited by B. L. Golden and A. A. Assad, Elsevier Science Publishers B.V. (1988).
2. Website www.paragon.com.
3. Gendreau, M., Laporte, G. and Potvin, J. Y., "Metaheuristics for the Capacitated Vehicle Routing Problem", In: Toth and Vigo D (Eds), The Vehicle Routing Problem, SIAM Monographs, Philadelphia, PA, (2002), 129-154.
4. Kirkpatrick, S, Gelatt, D. D. and Vecchi, M. P., "Optimisation by Simulated Annealing", ***Science***, Vol.220, (1983), 671-680.
5. Osman, I. H, "Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem", ***Annals of Operations Research***, Vol. 41, (1993), 421-451.
6. Johnson, S, Aragon, C, Mccgeoch, L. and Schevon, C., "Optimisation by Simulated Annealing: an Experiment Evaluation, Part I, Graph Partitioning", ***Operations Research***, Vol. 37, (1989), 865-892.