

THE INSTITUTE FOR SYSTEMS RESEARCH

ISR TECHNICAL REPORT 2010-1

---

# Solving Continuous Replenishment Inventory Routing Problems with Route Duration Bounds

Jeffrey Herrmann, Samuel Fomundam

The  
Institute for  
**Systems**  
Research



**A. JAMES CLARK**  
SCHOOL OF ENGINEERING

ISR develops, applies and teaches advanced methodologies of design and analysis to solve complex, hierarchical, heterogeneous and dynamic problems of engineering technology and systems for industry and government.

ISR is a permanent institute of the University of Maryland, within the A. James Clark School of Engineering. It is a graduated National Science Foundation Engineering Research Center.

[www.isr.umd.edu](http://www.isr.umd.edu)

# Solving Continuous Replenishment Inventory Routing Problems with Route Duration Bounds

Jeffrey W. Herrmann, Samuel Fomundam

*Institute for Systems Research, University of Maryland, College Park, MD 20742*

In a public health emergency, resupplying points of dispensing (PODs) with the smallest number of vehicles is an important problem in mass dispensing operations. To solve this problem, this paper describes the Continuous Replenishment Inventory Routing Problem (CRIRP) and presents heuristics for finding feasible solutions when the duration of vehicle routes cannot exceed a given bound. This paper describes a special case of the CRIRP that is equivalent to the bin-packing problem. For the general problem, the paper presents an aggregation approach that combines low-demand sites that are close to one another. We discuss the results of computational tests used to assess the quality and computational effort of the heuristics and the aggregation approach.

**Keywords:** Emergency preparedness, homeland security, inventory routing, continuous replenishment.

---

## 1. Introduction

The Continuous Replenishment Inventory Routing Problem (CRIRP) is a special type of inventory routing problem (IRP) in which vehicle operations occur around the clock. Our study of the CRIRP is motivated by our work with public health officials who must plan the logistics for resupplying points of dispensing (PODs), which will dispense medications to the public in case of a public health emergency such as an anthrax attack. After receiving an initial but limited supply of medication, the PODs will operate continuously, around the clock, in order to give out thousands of doses of medication. Vehicles will resupply the PODs continuously from a central depot that has a stockpile of medication. Each vehicle repeatedly follows the same route, starting out as soon as it can after returning to the depot. At each site, the vehicle must deliver enough medication to satisfy demand until its next visit. Note that it is not necessary that all of PODs be resupplied at the same frequency. It may be more efficient for some PODs (especially those with high demand) to be resupplied more often than others.

A central concern is to determine how many vehicles are needed to resupply the sites, which sites each vehicle should resupply, and the route that each vehicle should take.

Minimizing the number of vehicles is an important objective due to the limited number of available drivers and vehicles. Also, continuous replenishment means that the operating costs are related to the number of vehicles, which are continuously running.

Minimizing the number of vehicles could yield solutions that have long route durations. Such solutions have possible practical problems. Each site will need to store a large amount of inventory in order to operate until the next delivery. In addition, drivers cannot perform extremely long routes due to safety and regulatory restrictions. The final problem is one of management perception: those who are overseeing the resupply operation may wish to resupply sites at relatively high frequencies.

Thus, this could be viewed as a bi-objective problem of minimizing the number of vehicles and minimizing the route durations. We will attack this problem by setting a constraint on the route duration and minimizing the number of vehicles subject to this constraint.

Previous work on the CRIRP considered the problem without route duration bounds. Fomundam (2008) developed a branch-and-bound scheme that can be used for small instances and tested a randomized heuristic that was shown to perform poorly. Fomundam and Herrmann (2009) tested heuristics and a genetic algorithm and demonstrated their use on a real-world instance. The solution approaches presented here include modifications to those heuristics and completely new algorithms.

Although motivated by public health emergency planning, the CRIRP can occur in any setting where operations occur continuously and the resupply frequency of sites can vary.

Problems with travel times of many days between deliveries can be viewed as continuous replenishment as well.

In a typical formulation of the IRP (see, for instance, Campbell *et al.*, 1998), there is a single product that each customer consumes at a constant daily rate. Each customer also has a predetermined inventory capacity. A customer's existing inventory must not run out before a vehicle resupply. The IRP is solved over a planning horizon (for example, one week). There is a fleet of homogenous vehicles of a given capacity, and the objective is to minimize the cost of supplying the customers by identifying which days each customer should be supplied, determining the quantity to be supplied to each customer, and routing the fleet of vehicles to supply the determined quantities to the customers assigned to a particular day. Other notable work on the IRP includes Golden *et al.* (1984), Bard *et al.* (1998), and Jaillet *et al.* (2002). Campbell and Savelsbergh (2004a) present a two-phase approach to solving the IRP, and Campbell and Savelsbergh (2004b) discuss Vendor Managed Inventory Replenishment. In this version of the IRP, a vendor monitors customers' inventories and conducts replenishment of their inventories by coordinating inventory levels and vehicle deliveries to minimize long term costs. Moin and Salhi (2007) provide a recent review of the IRP.

In the IRP, vehicle routing decisions are made for each day. The routes start and end in the same day; they don't go into the next day. All of the vehicles are available at the beginning of the next day. There is a "jump" from one day to the next where no vehicles are operating. This characteristic does not exist in the CRIRP, in which customers are supplied continuously, around the clock. When vehicles return to the depot, they immediately reload and resupply their customers. These continuous operations are essential in a public health emergency.

Additionally, unlike the IRP, the CRIRP considers only indirectly (through the route duration bound) limits on the maximum inventory that can be stored at customer sites.

The CRIRP problem can be viewed as a strategic IRP, in that we consider a fleet sizing problem that is similar to Webb and Larson (1995). However, unlike Webb and Larson, we consider the case where replenishments happen continuously. That is, a route begins as soon as the vehicle completes its route and returns to the depot.

This paper addresses the single-product, deterministic, steady-state problem in which the loading and unloading times at each site are modeled as a constant time. (This is reasonable if the marginal time needed to unload an item is small compared to the travel times.) Inventory is treated as a continuous variable. The storage capacity at each site is not given, for it will be set appropriately after the routing problem is solved. The depot always has enough inventory to load trucks.

The contribution of this paper is to introduce the CRIRP with route duration bounds, which has not been previously studied, and to present useful heuristics and an aggregation approach for solving the problem. The conclusions discuss directions for future work on the problem.

## **2. Problem Formulation**

In the CRIRP, there are  $n$  sites (customers). Each site ( $i = 1, \dots, n$ ) has a demand rate of  $L_i$  items per time unit. This is the rate at which the site consumes material. There is a depot ( $i = 0$ ) that has an unlimited amount of material. The time spent at site  $i$  (to refill a vehicle or deliver material) is  $p_i$  for  $i = 0, \dots, n$ . The time to travel from site  $i$  to site  $j$  is  $c_{ij}$ . The vehicles are identical, each with capacity of  $C$  items of material. Also given is a route duration bound  $Q$ .

The problem is to find a feasible solution with the smallest number of vehicles. A feasible solution specifies a route for each vehicle, and each site is assigned to one route. The delivery amount at a site is the route duration multiplied by the site's demand rate.

We note that, if vehicles must have regular breaks (for driver rest or vehicle maintenance), then we can set the effective vehicle capacity to a value that is smaller than the actual capacity. Then, any solution that is feasible with respect to the effective vehicle capacity allows the vehicle to carry and deliver more than that solution specifies, which will give the vehicle some extra time for breaks.

A vehicle may visit the depot and obtain more material multiple times during a route. A partial route that starts at the depot and ends at the depot is a "subroute." A vehicle may have multiple subroutes but visits each site just once on its route.

Given a solution, we evaluate its feasibility as follows. Let vehicle  $v$  have  $r$  subroutes. Let the sequence  $s_{vj} = \{0, [1], \dots, [k], 0\}$  be subroute  $j$  for vehicle  $v$ , where  $k$  is the number of sites on the subroute and  $[i]$  is the index of the  $i$ -th site visited. The total time to complete the subroute is  $T(s_{vj}) = p_0 + c_{0[1]} + p_{[1]} + c_{[1][2]} + \dots + p_{[k]} + c_{[k]0}$ . The total time for vehicle  $v$  to complete all of its subroutes is  $T_v = T(s_{v1}) + \dots + T(s_{vr})$ . A vehicle route is feasible only if  $T_v \leq Q$ .

When the vehicle visits site  $[i]$ , it will need to deliver  $L_{[i]}T_v$  units of material in order to keep the site supplied until the vehicle's next visit. Let the total demand rate for the subroute be  $D(s_{vj}) = L_{[1]} + \dots + L_{[k]}$ . When vehicle  $v$  starts subroute  $s_{vj}$ , it should take  $D(s_{vj})T_v$  items in order to satisfy the demand of all the sites on that subroute; this quantity is the *load* of that subroute. Let  $D_v^* = \max\{D(s_{v1}), \dots, D(s_{vr})\}$ . The maximum load for vehicle  $v$  is  $M_v = D_v^*T_v$ . The

solution is feasible only if each site is assigned to exactly one vehicle and each vehicle's maximum load is not greater than the vehicle capacity. That is,  $M_v \leq C$  for all vehicles  $v = 1, \dots, K$ .

In order to demonstrate the existence of feasible solutions, consider the trivial subroutes  $z_i = \{0, i, 0\}$ , for  $i = 1, \dots, n$ . Then,  $T(z_i) = p_0 + c_{0i} + p_i + c_{i0}$  and  $D(z_i) = L_i$ . Clearly there are feasible solutions to CRIRP if and only if  $D(z_i)T(z_i) \leq C$  and  $T(z_i) \leq Q$  for all  $i = 1, \dots, n$ .

The objective is to find a feasible solution with the minimal number of vehicles. It is easy to see that CRIRP is NP-hard, like virtually all vehicle routing problems (Lenstra and Rinnooy Kan, 1981).

### 3. Example

Consider the six-site problem instance (along with three subroutes) shown in Figure 1. At each site, the demand rate  $L_i$  (in items per time unit) is shown in parentheses. The service time  $p_i = 1$  time unit at the depot and all sites. The travel time equals one time unit between the depot and sites 1, 2, 4, and 6 as well as between sites 2 and 3, between sites 3 and 4, between sites 5 and 6. The travel time between the depot and site 5 equals 1.4 time units.

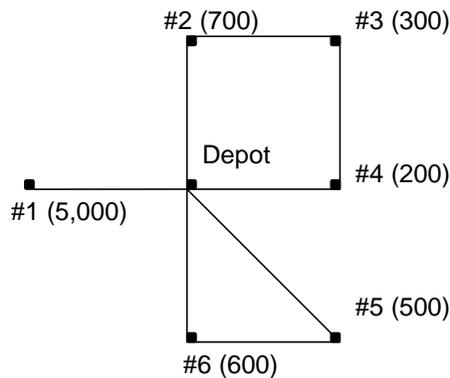


Figure 1. A six-site instance of the CRIRP, showing three subroutes.

If the vehicle capacity  $C = 20,000$  items and the route duration bound  $Q = 15$  time units, then one feasible solution to the instance in Figure 1 has two vehicles. The first vehicle follows only one subroute  $s_{11} = \{0, 1, 0\}$ . The demand  $D_1^* = D(s_{11}) = 5,000$  items per time unit, and the route duration  $T_1 = T(s_{11}) = 4$  time units, so the load  $M_1 = 20,000$  items. The second vehicle has two subroutes:  $s_{21} = \{0, 2, 3, 4, 0\}$  and  $s_{22} = \{0, 5, 6, 0\}$ . The first subroute demand  $D(s_{21}) = 1,200$  items per time unit, and the subroute duration  $T(s_{21}) = 8$  time units. The second subroute demand  $D(s_{22}) = 1,100$  items per time unit, and the subroute duration  $T(s_{22}) = 6.4$  time units. Therefore, the total route duration  $T_2 = T(s_{21}) + T(s_{22}) = 14.4$  time units.

$D_2^* = \max\{D(s_{21}), D(s_{22})\} = 1,200$  items, so  $M_2 = D_2^* T_2 = 17,280$  items. The load for the first subroute equals 17,280 items, and the load for the second subroute equals 15,840 items.

#### 4. The Special Case of Identical Demand

Consider the special case in which all  $L_i = L$  and  $Q \geq C/L$ . (This special case is a useful model for the POD resupply problem if the jurisdiction's mass dispensing plans call for a set of identical PODs.) In this case, we can show that the non-trivial subroutes of a feasible solution can be split into the trivial subroutes without increasing the maximum load of any vehicle. Thus, there is an optimal solution in which every vehicle's route is the concatenation of trivial subroutes.

**Theorem.** Given an instance of CRIRP in which all  $L_i = L$  and  $Q \geq C/L$ , then there exists an optimal solution in which every vehicle's route is the concatenation of trivial subroutes.

**Proof.** Consider a feasible solution in which a vehicle  $v$  visits  $n$  sites using  $r$  subroutes, where  $r < n$ . (The  $n$  here may be less than the number of sites in the entire problem.) Therefore,

at least one subroute visits more than one site. Let  $m_0 = 0$ . Renumber the sites and define  $m_k$  ( $k = 1, \dots, r$ ) so that the first subroute visits sites  $1, \dots, m_1$ , the second subroute visits sites  $m_1 + 1, \dots, m_2$ , and so forth, with  $m_r = n$ .

Let  $h = \max_{1 \leq k \leq r} \{m_k - m_{k-1}\}$ . Note that  $h \geq 2$  and  $hr \geq n$ . Let  $TT_k$  be the travel time of subroute  $k$ . Note that  $TT_k \geq c_{0i} + c_{i0}$  for all  $i \in \{m_{k-1} + 1, \dots, m_k\}$ .

Now consider the duration of each subroute  $k$ , and let  $T_0$  be the duration of the current route:

$$\begin{aligned} T(s_{vk}) &= p_0 + \sum_{i=m_{k-1}+1}^{m_k} p_i + TT_k \\ T_0 &= \sum_{k=1}^r T(s_{vk}) \\ &= rp_0 + \sum_{i=1}^n p_i + \sum_{k=1}^r TT_k \end{aligned}$$

For each subroute  $k$ , the total subroute demand rate  $D(s_{vk}) = (m_k - m_{k-1})L$ . The maximum subroute demand rate is therefore  $hL$ , and the maximum load is  $hLT_0$ . Because the solution is feasible,  $hLT_0 \leq C$ .

Now, modify this solution to construct a new solution in which this vehicle visits all of the same sites using trivial subroutes. Let  $t_i = p_0 + c_{0i} + p_i + c_{i0}$  for all  $i = 1, \dots, n$ . Let  $T_1$  be the duration of the new route:

$$\begin{aligned} T_1 &= \sum_{i=1}^n t_i = \sum_{i=1}^n (p_0 + c_{0i} + p_i + c_{i0}) \\ &= np_0 + \sum_{i=1}^n p_i + \sum_{i=1}^n (c_{0i} + c_{i0}) \end{aligned}$$

In this solution, the maximum subroute demand rate is  $L$ , and the maximum load is  $LT_1$ .

Now, we will show that  $LT_1 < hLT_0$  by showing that  $hT_0 - T_1$  is positive.

$$hT_0 - T_1 = p_0(hr - n) + (h-1) \sum_{i=1}^n p_i + \left( \sum_{k=1}^r hTT_k - \sum_{i=1}^n (c_{0i} + c_{i0}) \right)$$

Because  $hr \geq n$ , the first term is non-negative. Because  $h \geq 2$ , the second term is positive. To analyze the third term, we regroup the terms in the last summation by the subroutes to get the following:

$$\begin{aligned} \sum_{k=1}^r hTT_k - \sum_{i=1}^n (c_{0i} + c_{i0}) &= \sum_{k=1}^r \left( hTT_k - \sum_{i=m_{k-1}+1}^{m_k} (c_{0i} + c_{i0}) \right) \\ &\geq \sum_{k=1}^r \sum_{i=m_{k-1}+1}^{m_k} (TT_k - c_{0i} - c_{i0}) \end{aligned}$$

Each term of this double summation is non-negative. Therefore,  $hT_0 - T_1$  is positive, and  $LT_1 < hLT_0 \leq C$ . Moreover,  $T_1 \leq C/L \leq Q$ . This shows that using the trivial subroutes is also feasible because they reduce the load of the vehicle and do not excessively increase the route duration. Therefore, there exists an optimal solution with all trivial subroutes. QED.

Which vehicle should do which subroutes? Let  $t_i = p_0 + c_{0i} + p_i + c_{i0}$  for all  $i = 1, \dots, n$ .

Suppose vehicle  $v$  completes a set  $S_v$  of trivial subroutes. The route is feasible if and only if

$$M_v = L \sum_{i \in S_v} t_i \leq C, \text{ which is equivalent to } \sum_{i \in S_v} t_i \leq C/L.$$

This implies that the route duration is feasible as well. Thus, the problem becomes a bin packing problem in which the item size is  $t_i$  and the bin size is  $C/L$ . The packing of items into bins corresponds to the assignment of sites (and their trivial subroutes) to vehicles. Interestingly, the routing is trivial, because the load does not depend upon the sequence, so any sequence for a vehicle's route is sufficient in this special case. (Of course, the vehicle must follow the same sequence every time.)

## 5. Heuristics

Because CRIRP is NP-hard and we have no exact techniques that are useful for large instances, we developed and tested procedures for constructing solutions to the problem. We know of no other existing techniques for this problem.

The route-building heuristic presented below is a modification of the one presented by Fomundam and Herrmann (2009). The others are new algorithms.

### 5.1 Sequence-and-build heuristic

The first heuristic (which we denote *SB*) builds routes in two steps. First, it generates a sequence of all the sites. Then, it builds a feasible solution from this sequence. We considered eight versions of this heuristic by combining four sequencing algorithms with two algorithms that build solutions.

It is important to note that, when a solution is generated from this sequence, the sites may not remain in the same order. Thus, this is not a route-first, cluster-second heuristic (cf. Beasley, 1983). Instead, it operates in the spirit of the first-fit-decreasing heuristic used to solve the bin packing problem: the sequence is used only for determining which object to consider next.

#### Space-filling curve sequence

The first sequencing algorithm (which we denote *SFC*) generates a space-filling curve that begins at the depot and visits all of the sites.

To generate the space-filling curve, we use the procedure described in Bartholdi and Platzman (1988). The locations of the depot and sites are scaled and translated so that the depot is at the center of a unit square, and all of the sites fit into the unit square. (The translation procedure is described in Appendix A.) The space-filling curve assigns each site to a position between 0 and 1. Because sites with positions that are near 0 or 1 are in a corner of the unit

square (and far away from the depot), we generate a sequence of sites by starting with the sites in the interval  $[7/8, 1]$  and then visiting the sites in the interval  $[0, 7/8)$ .

### **Sweep sequence**

The second sequencing algorithm (which we denote *SWP*) sequences the sites using a simplified version of the sweep algorithm (Gillett and Miller, 1974).

The algorithm translates all of the sites so that the depot is at  $(0, 0)$ , determines each translated site's location in polar coordinates (with vectorial angles between  $-180$  and  $180$  degrees), and sorts the sites by their vectorial angles to generate a sequence.

The computational effort of sequencing the sites is  $O(n)$  to generate the vectorial angles and  $O(n \log n)$  to sort the sites based on position.

### **Nearest neighbor sequence**

The third sequencing algorithm (which we denote *NN*) sequences the sites by using the standard nearest neighbor algorithm to generating a tour that starts at the depot. The computational effort of sequencing the sites is  $O(n^2)$ .

### **Demand sequence**

The fourth sequencing algorithm (which we denote *DMD*) sequences the sites by their demand rates. Site  $i$  precedes site  $j$  if  $L_i > L_j$ . The computational effort of sequencing the sites is  $O(n \log n)$ .

### **Building a solution**

Given a sequence of the sites, we build a solution using an algorithm similar to that in the first stage of the build-and-bound heuristic. We considered two versions of the algorithm.

Given a choice of feasible locations for adding a site to an existing route, the first algorithm

(which we denote  $D$ ) selects the location that leads to the smallest route duration; the second algorithm (which we denote  $L$ ) selects the location that leads to the smallest maximum load.

First, the algorithm generates a route that visits only the first site in the sequence. Then, given a partial solution, the algorithm considers the next site in the sequence. It determines if there are any feasible ways to add this site to the first route in the partial solution by considering inserting the site into existing subroutes or by adding a new subroute with only this site. This algorithm (unlike the one in the build-and-bound heuristic) considers both the vehicle capacity and the route duration bound. If there are multiple ways to add the site, then it selects the one that results in the smallest route duration (or maximum load). If the site cannot be added to this route, it then considers the remaining routes until a feasible solution is found. If the site cannot be added to any route, then the algorithm creates a new route with only this site. After this site is considered, the algorithm goes on to the remaining sites in the sequence. The computational effort of building a solution from a given sequence is  $O(n^2)$ .

We denote the eight versions of the SB heuristic by the combinations SFC-D, SFC-L, SWP-D, SWP-L, NN-D, NN-L, DMD-D, and DMD-L.

## **5.2 Route-building Heuristic**

The second heuristic (which we denote  $RB$ ) builds routes by creating instances of the capacitated vehicle routing problem (CVRP), which has been studied extensively (see, for example, Toth and Vigo, 1998). First, the  $RB$  heuristic sequences the sites by demand rate from low to high. Then it begins building routes, starting with the low-demand sites. When a route has been built, the sites assigned to that route are removed from the instance, and the  $RB$  heuristic continues to build routes until all sites have been assigned to routes.

To build a route, the RB heuristic first finds a lower bound on how many of the unassigned sites can be placed feasibly into a route. Starting with the first unassigned site, it adds the trivial subroutes corresponding to those sites until adding the next one would cause the maximum vehicle load to exceed vehicle capacity. Let  $LL$  be the largest number of trivial subroutes that can fit into a feasible route. Then, starting with  $N = LL + 1$ , the RB heuristic tries to find a feasible route with the first  $N$  unassigned sites.

To find a feasible route, the RB heuristic constructs instances of CVRP that have the depot and the first  $N$  unassigned sites. The vehicle capacity remains equal to  $C$ , but the quantities to be delivered to the sites in the CVRP instance are determined by the duration bound  $B$ . The RB heuristic uses different values of  $B$  in the range from  $C/L_{\max}$  down to  $C/\sum L_i$  (where the max and the sum are taken over the set of  $N$  sites in the CVRP instance). For each value of  $B$ , the RB heuristic multiplies each site's demand rate by  $B$  to create that site's delivery quantity and then uses the Clarke-Wright savings algorithm to construct for a solution to the CVRP instance. If the total duration of the routes in the solution is not greater than  $B$  and not greater than  $Q$ , then the solution forms a feasible route for the CRIRP problem by using the solution's routes as the subroutes for a vehicle. (If a feasible solution is found for one value of  $B$ , no more values of  $B$  need to be checked.)

If no feasible solution can be found for  $N = LL + 1$ , then the RB heuristic sets  $N = LL$  and finds a feasible route (which must be possible because the  $LL$  trivial subroutes are a feasible solution.) Otherwise, it increases  $N$  by 1 and tries to find a feasible route for the expanded set of unassigned sites. It repeats this until, for some value of  $N$ , it can find no feasible route using the Clarke-Wright savings algorithm for any value of  $B$  (or all unassigned sites are in the feasible solution). At this point, the heuristic saves the last feasible solution found as a route for one

vehicle (the routes from the CVRP solution become the subroutes for this vehicle). As mentioned above, the sites on that route are removed from the instance, and the RB heuristic continues with the remaining sites until all of them have been assigned to routes. The number of vehicles in the solution equals the number of feasible routes that were saved.

It is easy to see that a CRIRP route built from a feasible CVRP solution is feasible.

Consider a subroute  $s_{vj} = \{0, [1], \dots, [k], 0\}$ . The total demand for the subroute is

$D(s_{vj}) = L_{[1]} + \dots + L_{[k]}$ , and its load is therefore  $D(s_{vj})T_v$ , where  $T_v$  is the route duration.

Because  $T_v \leq B$ , the load is not greater than  $B(L_{[1]} + \dots + L_{[k]}) = BL_{[1]} + \dots + BL_{[k]}$ , which is the sum of the delivery quantities for the sites on this subroute. Because these sites are a feasible route in the CVRP problem, this sum must be no greater than  $C$ . Therefore, the subroute load is not greater than  $C$ .

For example, if we consider the example from Section 3, then the RB heuristic will, at some point during its execution, consider the  $N = 5$  smallest demand sites, which are sites 4, 3, 5, 6, and 2. When  $B = C/L_{\max} = 20,000/700 = 28.57$ , it creates an instance of CVRP with delivery quantities for sites 2 to 6 of 20,000, 8,570, 5,710, 14,290, and 17,140. A feasible solution to the CVRP has four routes: 0-4-5-0, 0-3-0, 0-6-0, and 0-2-0. The total duration is 19.24 time units, which is less than  $B$ , so this is a feasible CRIRP route. To confirm this, note that the maximum subroute demand is 700 items per time unit, so the maximum load is 13,470 items, which is less than the vehicle capacity.

However, when  $N = 6$ , the largest value of  $B = 20,000/5,000 = 4$ , and there is no feasible solution because it is impossible to find a feasible route with a duration less than or equal to 4.

So the solution with five sites becomes the route for the first vehicle. A second vehicle is required for the last unassigned site (site 1).

In our implementation of the RB heuristic, the loop over  $B$  considered six equally-spaced values from  $C/L_{\max}$  down to  $C/\sum L_i$ .

The computational effort of the RB heuristic depends upon  $N_B$ , the number of values of the duration bound  $B$  that are considered. The Clarke-Wright savings algorithm requires  $O(n^2 \log n)$  effort (Golden *et al.*, 1980), and this is performed up to  $N_B$  times in order to find a feasible route for  $N$  unassigned sites. Altogether, the RB heuristic will try to find at most  $n$  feasible routes. Thus, the RB heuristic requires  $O(N_B n^3 \log n)$  effort.

### 5.3 Build-and-bound heuristic

The SB heuristic uses a simple algorithm to build one solution, while the RB heuristic repeatedly calls a more sophisticated algorithm in its search for good routes. The build-and-bound heuristic is a hybrid that first builds a solution using an algorithm similar to the SB heuristic and then solves a CVRP like the RB heuristic. Thus, it combines the relative simplicity of the SB heuristic with the power of the RB heuristic.

The heuristic (which we denote  $BB$ ) is a two-stage approach:

1. Build a solution without considering the route duration bound;
2. Remove any routes that are too long, construct and solve a CVRP, and use that solution to generate routes.

The first stage begins by ordering the sites by the demand rates  $L_i$  and generates a route that visits only the site with the largest  $L_i$ . Then, given a partial solution, this stage considers the site with the largest remaining  $L_i$ . It determines if there are any feasible ways to add this site to

the first route in the partial solution by considering inserting the site into existing subroutes or by adding a new subroute with only this site. At this point, the algorithm considers the vehicle capacity but not the route duration bound. If there are multiple ways to add the site, then it selects the one that results in the smallest maximum load. If the site cannot be added to this route, it then considers the remaining routes until a feasible solution is found. If the site cannot be added to any route, then the algorithm creates a new route with only this site. After this site is considered, the algorithm goes on to the remaining sites.

Given the solution that the first stage built, the second stage removes any routes that are longer than the route duration bound  $Q$  and creates a CVRP with these sites. In the CVRP, each site's delivery quantity equals  $QL_i$ . The algorithm then calls the Clarke-Wright savings algorithm to construct a solution to the CVRP, which yields a set of routes, which will become subroutes in the CRIRP solution. Then, the algorithm uses the first-fit-decreasing algorithm to find a solution to the bin-packing problem in which each unassigned subroute  $s_k$  is an item, the item size is the subroute duration  $T(s_k)$ , and the bin capacity is  $Q$ . This creates new feasible routes that are added to the feasible routes from the previous solution.

The computational effort of building a solution in the first stage is  $O(n^2)$ . The Clarke-Wright savings algorithm requires  $O(n^2 \log n)$  effort (Golden *et al.*, 1980), and the bin packing heuristic requires  $O(n \log n)$  time (because the number of subroutes to be packed is bounded by  $n$ ).

## 6. Aggregation

In an effort to improve the performance of these heuristics, we employed an aggregation approach that first aggregates an instance, constructs a solution for the aggregate instance, and

then disaggregates that solution. Aggregation is a well-known and valuable technique for solving optimization problems, especially large-scale mathematical programming problems. Model aggregation replaces a large optimization problem with a smaller, auxiliary problem that is easier to solve (Rogers *et al.*, 1991). The solution to the auxiliary model is then disaggregated to form a solution to the original problem. Model aggregation has been applied to a variety of production and distribution problems, including machine scheduling problems. For example, Rock and Schmidt (1983) and Nowicki and Smutnicki (1989) aggregated the machines in a flow shop scheduling problem to form a two-machine problem.

Francis and co-workers have developed methods to aggregate demand points in location problems (see, for example, Francis *et al.*, 2004, 2009). For the probabilistic traveling salesman problem, Campbell (2006) presented an aggregation approach that combined customers in the same region. For the probabilistic traveling salesman problem with deadlines, Campbell and Thomas (2009) used a temporal aggregation that modified an instance by using a larger time unit. Similarly, Newman *et al.* (2006, 2007) aggregated a multi-period production planning problem by combining multiple consecutive time periods.

Herrmann (2007) developed an aggregation scheme to generate better solutions for the response time variability (RTV) problem. Using aggregation with parameterized stride scheduling and an improvement heuristic generates solutions with lower RTV than those generated by parameterized stride scheduling and an improvement heuristic (Herrmann, 2009a, b). Using aggregation also reduces the computational effort.

The aggregation approach used here aggregates an instance by combining sites that have low demand rates and are close to each other. Aggregation removes these sites and adds a new aggregate site that has a demand rate equal to the sum of the sites' demand rates. Thus,

aggregation reduces the number of sites that need to be considered. It is important that a feasible solution to the aggregate instance can be disaggregated into a feasible solution for the original instance. Therefore, the unload time at the aggregate site is set to the sum of the unload times at the sites plus the travel times needed to move between the sites. The travel time between the aggregate site and any remaining site is set to the maximum travel time between the original sites and the other site.

The following pseudocode describes the aggregation algorithm. The value used for determining when sites are close enough to combine is based on the results of some pilot studies. The demand rate of any aggregate site will be no greater than the maximum demand rate of any original site. The sequences  $H_i$  are used to record which sites were combined to form the aggregate site.

**Aggregation.** Given: an instance with  $n$  sites.

1. Set  $c_{agg} = \frac{1}{10} \max \{c_{01}, \dots, c_{0n}\}$  and  $L_{max} = \max \{L_1, \dots, L_n\}$ .

Set  $H_i = (i)$  for  $i = 1, \dots, n$ .

2. Sort the sites by  $L_i$  (from small to large). Let  $[i]$  be the index of the  $i$ -th site in this sequence. Set  $i = 1$  and  $j = 2$ .

3. If  $L_{[i]} + L_{[j]} \leq L_{max}$  and  $c_{[i][j]} \leq c_{agg}$ ,

set  $T = p_0 + \max \{c_{0[i]}, c_{0[j]}\} + p_{[i]} + c_{[i][j]} + p_{[j]} + \max \{c_{[i]0}, c_{[j]0}\}$ .

If  $T \leq Q$  and  $T(L_{[i]} + L_{[j]}) \leq C$ , then go to step 4. Else, go to step 5.

4. Remove site  $[j]$  and update the information for site  $[i]$  as follows:

$$L_{[i]} \leftarrow L_{[i]} + L_{[j]}, \quad p_{[i]} \leftarrow p_{[i]} + c_{[i][j]} + p_{[j]}, \quad H_{[i]} \leftarrow H_{[i]}H_{[j]},$$

$c_{[i]k} \leftarrow \max \{c_{[i]k}, c_{[j]k}\}$  and  $c_{k[i]} \leftarrow \max \{c_{k[i]}, c_{k[j]}\}$  for all  $k$  except  $[j]$ ;

$n \leftarrow n - 1$ .

5. Increase  $j$  by 1. If  $j > n$  then increase  $i$  by 1 and set  $j = i + 1$ . If  $i = n$ , then stop.

6. Go to step 3.

The computational effort of aggregating an instance is  $O(n^3)$  because we must consider  $O(n^2)$  pairs of sites and each combination requires  $O(n)$  effort.

For the sequencing algorithms that require locations, the coordinates of the location of an aggregate site  $i$  are the average of the coordinates of the locations of the original sites in the sequence  $H_i$ .

To disaggregate a solution for the aggregate instance, each site  $i$  from the aggregate instance is replaced by the original sites in the sequence  $H_i$ . That is, all of these sites will be in the same subroute, and the vehicle will visit these sites consecutively in the order given by the sequence  $H_i$ . The number of vehicles (routes) and the number of subroutes remain the same.

Disaggregating a solution therefore requires  $O(n)$  effort.

## 7. Computational Tests

The purpose of the computational tests was to evaluate the relative performance of the heuristics with and without using aggregation. The heuristics were coded in Matlab. We used an implementation of the Clarke-Wright saving algorithm from Matlog, the Logistics Engineering Matlab Toolbox, created by Michael G. Kay at North Carolina State University.

To test the heuristics developed, we used four sets of location data obtained from the TSPLIB, a library of sample instances that provide either location data or the costs associated

with the paths of a graph. They serve as test data for TSP solvers. We selected the following four sets of data:

- Burma 14: 14 cities in Burma; and
- Ulyssess 22: 22 locations from the Odyssey of Ulysses.
- Berlin 52: 52 locations in Berlin, Germany;
- Bier 127: 127 beer gardens in Augsburg, Germany;

In each of these four data sets, the locations are sequentially indexed using positive integers. Each location also has Cartesian coordinates. Although the data are sufficient for testing TSP solvers, more data is needed for the CRIRP. We used the following scheme to ensure that all of the trivial subroutes are feasible routes and, therefore, each instance has at least one feasible solution.

We assigned the depot to the first location (which is a central location in these data sets). The other locations are then designated as sites and numbered from 1. We used the Euclidean distance between each pair of sites as the (symmetric) travel times between the sites. We then calculated the average travel time  $A$  of the instance. We then specified four values for the load time:  $A/50$ ,  $A/5$ ,  $A$ , and  $3A$ . (Every site in the instance had the same load time.)

We constructed three values for the vehicle capacity for each data set. To do this, we set the load times equal to  $3A$  and then found the largest trivial subroute duration. We multiplied the largest duration by 400 to get the maximum subroute load. (As discussed below, 400 is an upper bound on site demand rate.) We multiplied this by 1.5, 5, and 10 to get the values for the vehicle capacity.

We arbitrarily chose an average demand rate of 200 items per time unit. The depot demand rate was set to zero. We then generated a set of samples from a truncated standard

normal distribution. (Samples less than -2.5 and samples greater than 2.5 were discarded.) We constructed three sets of demand rates using three different values for the standard deviation: 80, 40, and 20. The demand rates at each site were determined by multiplying the standard deviation by the sample and adding 200 (so the average demand rate was approximately 200). Note that all of the demand rates are between 0 and 400.

In this manner, for each of the four TSP data sets, we created 4 load times, 3 sets of demand rates, and 3 vehicle capacities. Thus, we generated 36 unconstrained instances of CRIRP for each TSP data set, giving us a total of 144 unconstrained instances.

For each unconstrained instance, we created three route duration bounds. The smallest route duration bound was the largest trivial subroute duration:  $\max\{p_0 + c_{0i} + p_i + c_{i0}\}$ . The largest route duration bound was  $C / \max\{L_i\}$ . The third route duration bound was the average of the first two bounds. Note that all of the trivial subroutes are feasible with respect to these bounds. Combining the unconstrained instances with the route duration bounds gave us a total of 432 instances.

We ran the heuristics on each instance and tracked the time needed to run the heuristics. We also aggregated each instance, ran the heuristics on the aggregate instance, disaggregated these solutions, and tracked the time required.

The average time for the BB heuristic ranged from 0.0106 seconds on the 14-site instances to 0.9940 seconds on the 127-site instances. The average time for the RB heuristic ranged from 0.1035 seconds on the 14-site instances to 12.9014 seconds on the 127-site instances. The time required for the BB heuristic was not significantly affected by other changes in the instance, but the run time of the RB heuristic decreased when the vehicle capacity was small or when the load time was high. This occurs because, in these scenarios, long routes are

not feasible, so the RB heuristic needed less time to build a route. All eight versions of the SB heuristic required less time than the BB and RB heuristics, as shown in Table 1.

Using aggregation generally increased the time required for the heuristics. Except for the RB heuristic, the time required to aggregate and disaggregate overshadowed any savings due to fewer sites in the aggregate instances, as shown in Table 2. Two factors make using aggregation with the RB heuristic desirable. First, the computational effort of the RB heuristic (compared to the computational effort of the other heuristics) is more sensitive to  $n$ ; therefore, reducing the number of sites using aggregation has more impact on the run times. Second, the overhead of aggregation and disaggregation is relatively small for the RB heuristic, which has larger run times than the other heuristics.

Table 1. Run times for the heuristics on the CRIRP problem instances without aggregation  
(all run times in seconds)

$n$	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
14	0.0106	0.1035	0.0042	0.0044	0.0015	0.0016	0.0015	0.0016	0.0011	0.0012
22	0.0188	0.3151	0.0033	0.0035	0.0015	0.0017	0.0018	0.0020	0.0013	0.0015
52	0.1611	2.0818	0.0090	0.0108	0.0046	0.0064	0.0053	0.0074	0.0044	0.0062
127	0.9940	12.9014	0.0270	0.0389	0.0161	0.0285	0.0180	0.0318	0.0166	0.0301

Table 2. Run times for the heuristics on the CRIRP problem instances with aggregation  
(all run times in seconds)

$n$	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
14	0.0122	0.1022	0.0046	0.0047	0.0034	0.0035	0.0036	0.0037	0.0033	0.0034
22	0.0221	0.2862	0.0069	0.0072	0.0052	0.0054	0.0055	0.0058	0.0050	0.0053
52	0.1775	1.9321	0.0279	0.0300	0.0237	0.0257	0.0243	0.0266	0.0235	0.0254
127	1.0578	10.4549	0.1281	0.1397	0.1182	0.1294	0.1200	0.1330	0.1186	0.1298

As the vehicle capacity increases and the load time decreases and the route duration bound increases, the number of vehicles required decreases. Of course, at least one vehicle will be required. Therefore, for reporting the quality of the solutions generated, we will consider the

ratio of the number of vehicles in the solution generated by a heuristic for an instance to the number of vehicles in the best solution generated by any heuristic for that instance.

As shown in Tables 3, 4, 5, and 6, the relative solution quality of the heuristics depended greatly upon the route duration bound. The BB heuristic generally creates good solutions but does slightly worse with the medium route duration bounds. The RB heuristic generates poor solutions with small route duration bounds and very good solutions with large route duration bounds. For the SB heuristic, the SFC-D, SWP-D, NN-D, and DMD-D versions generate solutions that are worse as the route duration bound increases, but the SFC-L, SWP-L, NN-L, and DMD-L versions generate solutions that are better as the route duration bound increases. When the route duration bound was small or when the number of sites was large, using the DMD sequencing algorithm generally creates solutions that are not as good as those generated by using the SFC and SWP sequencing algorithms. The NN sequencing algorithm generates solutions that are sometime as good as the SFC and SWP solutions, sometimes as bad as the DMD solutions, and sometimes in between.

Table 3. Average relative solution quality for the heuristics for problem set Burma 14.

bound	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
small	1.0069	1.4722	1.0417	1.2292	1.0000	1.1875	1.0644	1.2102	1.2519	1.3144
med.	1.0880	1.0139	1.4630	1.1620	1.4630	1.1528	1.4815	1.1296	1.4769	1.1528
large	1.0000	1.0264	1.6245	1.0069	1.6190	1.0069	1.6468	1.0417	1.6468	1.0000

Table 4. Average relative solution quality for the heuristics for problem set Ulysses 22.

bound	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
small	1.0185	1.1858	1.0185	1.1944	1.0185	1.1968	1.0412	1.2843	1.1227	1.3565
med.	1.0972	1.0799	1.6412	1.0694	1.6782	1.0833	1.6991	1.0938	1.7315	1.0891
large	1.0278	1.0046	1.9213	1.0000	1.9716	1.0000	1.9901	1.0046	2.0225	1.0046

Table 5. Average relative solution quality for the heuristics for problem set Berlin 52.

bound	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
small	1.0014	1.7002	1.0577	1.2829	1.0610	1.2559	1.1547	1.4737	1.2498	1.4580
med.	1.0914	1.0838	1.9010	1.2899	1.9428	1.2851	1.9581	1.2714	2.1587	1.2925
large	1.0607	1.0242	2.1521	1.0114	2.2057	1.0057	2.2200	1.0143	2.4340	1.0306

Table 6. Average relative solution quality for the heuristics for problem set Bier 127.

bound	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
small	1.0062	1.9724	1.0655	1.3431	1.0746	1.4107	1.0886	1.4340	1.4355	1.6976
med.	1.0854	1.1215	1.9795	1.3144	2.0088	1.2737	1.9838	1.3225	2.3469	1.3690
large	1.0804	1.0548	2.3615	1.0739	2.3973	1.0435	2.3640	1.1040	2.7684	1.0578

Using aggregation led to better solutions more often when the instances had a high standard deviation of demand. In these instances, there are more small demand rate sites that can be combined. When the standard deviation of demand is medium, sites were combined only in instances from the Bier 127 data set. When the standard deviation of demand is low, no sites were combined. Table 7 shows the average number of sites in the aggregate instance for the instances with high and medium standard deviation of demand

Of course, sometimes aggregation can lead to worse solutions. Overall, as shown in Tables 8 and 9, it appears that using aggregation did not generally improve the quality solutions except when used with the DMD-L version of the SB heuristic on instances with high standard deviation of demand. Note that the results in these tables consider the ratio of the number of vehicles in the disaggregated solution generated by a heuristic on an aggregate instance to the number of vehicles in the solution generated by the same heuristic on the original instance.

Because the SFC, SWP, and NN sequencing algorithms do not consider site demand rate, aggregation does not significantly affect the order in which sites are considered when building the route. In contrast, the DMD sequencing algorithm does consider the site demand rate, and aggregation combines two or more small demand rate sites into a site with a larger demand rate,

so they are considered at a different point when building the route, which could lead to better solutions.

Table 7. Average number of sites (including depot) after aggregation.

demand std. dev.	Burma 14	Ulysses 22	Berlin 52	Bier 127
high	13.1	19.0	46.3	99.6
med.	14.0	22.0	52.0	124.1

Table 8. Number of instances when using aggregation generated a better solution.

demand std. dev.	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
high	2	24	7	20	13	18	6	22	22	46
med.	6	9	3	5	3	12	1	6	12	9

Table 9. Average relative solution quality when using aggregation.

demand std. dev.	BB	RB	SFC-D	SFC-L	SWP-D	SWP-L	NN-D	NN-L	DMD-D	DMD-L
high	1.0331	1.0037	1.0337	1.0111	1.0240	1.0009	1.0243	1.0003	0.9909	0.9692
med.	0.9962	0.9996	1.0041	0.9962	1.0007	0.9939	1.0014	0.9993	0.9979	0.9945

## 9. Summary and Conclusions

This paper presented a version of the CRIRP, a type of inventory routing problem that has an interesting link between the elements of time and demand. The continuous replenishment means that the operating costs are related to the number of vehicles. The demand at each site is a rate (items per time unit), not a fixed amount, so the delivery quantities depend upon the route duration. In the special case in which all sites have the same demand, the problem is equivalent to the bin packing problem. However, the more general case involves the traditional elements of routing as well as assignment. This paper considers the CRIRP with route duration bounds. Experimental results show that, compared to other heuristics, a fast sequencing and construction algorithm (the SB heuristic) finds better solutions quickly when the route duration bounds are large. When the route duration bounds are small, a hybrid approach (the BB heuristic) that

formulates and solves a CVRP for a limited number of sites generates solutions that are better than those that the other heuristics generate. A more computationally expensive procedure (the RB heuristic) that solves many CVRP subproblems does not generate better solutions.

This work has focused on formulating the problem and suggesting some approaches that can generate high-quality solutions quickly. These can be employed easily in decision support tools that help public health officials develop emergency preparedness plans. Additional work is needed to refine the heuristics, test search algorithms, and develop exact methods such as column generation and branch-and-cut procedures. Finally, other formulations of the bi-objective problem (minimizing the number of vehicles and minimizing route durations) remain to be studied.

### **Acknowledgements**

Kay Aaby, Rachel Abbey, Carol Jordan, and Kathy Wood at the Montgomery County, Maryland, Public Health Services provided helpful information about the medication distribution problem. This publication was supported by award number 5H75TP000309-02 from the Centers for Disease Control and Prevention (CDC) to National Association of County and City Health Officials (NACCHO). Its contents are solely the responsibility of Montgomery County, Maryland Advanced Practice Center for Public Health Emergency Preparedness and Response and the University of Maryland and do not necessarily represent the official views of the CDC or NACCHO. The authors also appreciate the comments of anonymous reviewers on an earlier version of this manuscript.

## References Cited

- Bard, Jonathan F., Liu Huang, Patrick Jaillet, and Moshe Dror, “A decomposition approach to the inventory routing problem with satellite facilities,” *Transportation Science*, Volume 32, Number 2, pages 189 – 203, 1998.
- Bartholdi, John J., III and Loren K. Platzman, “Heuristics Based on Spacefilling Curves for Combinatorial Problems in Euclidean Space,” *Management Science*, Volume 34, Number 3, pages 291-305, 1988.
- Beasley, JE, “Route First-Cluster Second Methods for Vehicle Routing,” *OMEGA*, Volume 11, Number 4, pages 403-408, 1983.
- Campbell, Ann Melissa, “Aggregation for the probabilistic traveling salesman problem,” *Computers & Operations Research*, Volume 33, pp. 2703–2724, 2006.
- Campbell, Ann Melissa, Clarke, L., Kleywegt, A., and Savelsbergh, M., “The Inventory Routing Problem,” *Fleet Management and Logistics* (eds. Crainic, G. L. and Laporte, G.), Kluwer Academic Publishers, Norwell, Massachusetts, pages 95-113, 1998.
- Campbell, Ann Melissa, and Martin W. P. Savelsbergh, “A Decomposition Approach for the Inventory-Routing Problem,” *Transportation Science*, Volume 38, Number 4, pages 488–502, November 2004a.
- Campbell, Ann Melissa, and Martin W. P. Savelsbergh, “Delivery Volume Optimization,” *Transportation Science*, Volume 38, Number 2, pages 210-223, May 2004b.
- Campbell, Ann Melissa, and Barrett W. Thomas, “Runtime reduction techniques for the probabilistic traveling salesman problem with deadlines,” *Computers & Operations Research*, Volume 36, Issue 4, pages 1231-1248, 2009.

- Clarke, G., and J.W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, Volume 12, Number 4, pages 568-581, 1964.
- Fomundam, Samuel, *Solving Continuous Replenishment Inventory Routing Problems*, M.S. Thesis, University of Maryland, College Park, 2008.
- Fomundam, Samuel, and Jeffrey W. Herrmann, "Solution Techniques for Continuous Replenishment Inventory Routing Problems," Technical Report 2009-11, Institute for Systems Research, University of Maryland, July, 2009.
- Available online at <http://hdl.handle.net/1903/9404>
- Francis, R. L., T. J. Lowe, M. B. Rayco, and A. Tamir, "Aggregation error for location models: survey and analysis," *Annals of Operations Research*, Volume 167, Number 1, pages 171-208, 2009.
- Francis, R. L., T. J. Lowe, A. Tamir, and H. Emir-Farinas, "A framework for demand point and solution space aggregation analysis for location models," *European Journal of Operational Research*, Volume 159, Issue 3, pages 574-585, 2004.
- Gillett, Billy E., and Leland R. Miller, "A Heuristic Algorithm for the Vehicle-Dispatch Problem," *Operations Research*, Volume 22, Number 2, pages 340-349, 1974.
- Golden, B., L. Bodin, T. Doyle, and W. Stewart Jr., "Approximate Traveling Salesman Algorithms," *Operations Research*, Vol. 28, No. 3, Part 2, pp. 694-711, 1980.
- Golden, Bruce, Arjang Assad, and Roy Dahl, "Analysis of a large scale vehicle routing problem with an inventory component," *Large Scale Systems*, Volume 7, pages 181-190, 1984.
- Herrmann, Jeffrey W., "Generating Cyclic Fair Sequences using Aggregation and Stride Scheduling," Technical Report 2007-12, Institute for Systems Research, University of Maryland, College Park. Available online at <http://hdl.handle.net/1903/7082>

- Herrmann, Jeffrey W., “Generating Cyclic Fair Sequences for Multiple Servers,” MISTA 2009, Dublin, Ireland, August 10-12, 2009a.
- Herrmann, Jeffrey W., “Using aggregation to reduce response time variability in cyclic fair sequences,” *Journal of Scheduling*, DOI 10.1007/s10951-009-0127-7, published online: 15 August 2009b.
- Jaillet, Patrick, Jonathan F. Bard, Liu Huang, and Moshe Dror, “Delivery cost approximations for inventory routing problems in a rolling horizon framework,” *Transportation Science*, Volume 36, Number 3, pages 292-300, 2002.
- Lenstra, J.K., and A. Rinnooy Kan, “Complexity of vehicle routing and scheduling problems,” *Networks*, Volume 11, pages 221-227, 1981.
- Moin, NH, and S Salhi, “Inventory routing problems: a logistical overview,” *Journal of the Operational Research Society*, Volume 58, pages 1185–1194, 2007.
- Newman, A.M., and M. Kuchta, “Using aggregation to optimize long-term production planning at an underground mine,” *European Journal of Operational Research*, Volume 176, pages 1205–1218, 2007.
- Newman, Alexandra, Michael Martinez, and Mark Kuchta, “A Review of Long- and Short-Term Production Scheduling at LKAB's Kiruna Mine,” in *Handbook of Production Scheduling*, J.W. Herrmann, editor, Springer, New York, 2006.
- Nowicki, E., and C. Smutnicki, “Worst-case analysis of an approximation algorithm for flow shop scheduling,” *Operations Research Letters*, 8:171-177, 1989.
- Prins, C, “A simple and effective evolutionary algorithm for the vehicle routing problem,” *Computers & Operations Research*, Volume 31, pages 1985-2002, 2004.

Rock, H., and G. Schmidt, "Machine Aggregation Heuristics in Shop Scheduling," *Methods of Operations Research*, 45:303-314, 1983.

Rogers, David F., Robert D. Plante, Richard T. Wong, and James R. Evans, "Aggregation and Disaggregation Techniques and Methodology in Optimization," *Operations Research*, 39(4):553-582, 1991.

Toth, P. and Vigo, D., "Exact Solution of the Vehicle Routing Problem," *Fleet Management and Logistics* (eds. Crainic, G. L. and Laporte, G.), Kluwer Academic Publishers, Norwell, Massachusetts, 1-31, 1998.

TSPLIB, <http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html>, accessed July 8, 2008.

Webb, Ian R., and Richard C. Larson, "Period and phase of customer replenishment: a new approach to the strategic inventory/routing problem," *European Journal of Operational Research*, Volume 85, pages 132-148, 1995.

#### **Appendix A. Site translation algorithm.**

Given the depot location  $(x_0, y_0)$  and the site locations  $(x_1, y_1), \dots, (x_n, y_n)$ , set

$$x_{\min} = \min \{x_1, \dots, x_n\}, \quad x_{\max} = \max \{x_1, \dots, x_n\}, \quad y_{\min} = \min \{y_1, \dots, y_n\}, \quad \text{and} \quad y_{\max} = \max \{y_1, \dots, y_n\}.$$

Then, set  $D_x = 2 \max \{x_0 - x_{\min}, x_{\max} - x_0\}$  and  $D_y = 2 \max \{y_0 - y_{\min}, y_{\max} - y_0\}$ .

The translated position of the depot is  $(\frac{1}{2}, \frac{1}{2})$ . The translated position of site  $i$ ,  $i = 1, \dots, n$ , is  $(\frac{1}{2} + (x_i - x_0)/D_x, \frac{1}{2} + (y_i - y_0)/D_y)$ . It is easy to show that all of the translated positions will be in the unit square with opposite corners at (0,0) and (1,1).