

A Genetic Algorithm for the Home-Delivered Meals Location-Routing Problem

Hakan Yildiz¹, Michael P. Johnson², and Stephen Roehrig²

¹ Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213
hakanyil@andrew.cmu.edu

² H. John Heinz III School of Public Policy and Management, Carnegie Mellon University, Pittsburgh, PA, 15213
{ johnson2, roehrig } @andrew.cmu.edu

Abstract. Home-delivered meals (HDM) provision is a volunteer-staffed activity for which little strategic planning is currently performed. We present and evaluate a Genetic Algorithm to solve the HDM location-routing problem (LRP). This planning model addresses facility location, allocation of demand to facilities, and design of delivery routes, while balancing efficiency and effectiveness considerations. We provide computational results on benchmark LRP instances.

1 Introduction

Home-delivered meals (HDM) provision refers to the issue of delivering hot meals to the homebound infirm and elderly to ensure adequate nutrition and independent living. HDM, often referred to as “meals-on-wheels,” is a service usually provided by volunteers who cook and deliver meals. This service is inexpensive when measured in dollar costs. However coordination is often difficult due to changes in meal demand and variations in the supply of volunteers, and financial support for purchase of meal ingredients. HDM planning is a topic of interest to public-sector planning agencies and funding bodies since many regions are experiencing a shift in demand for HDM service as a result of outmigration from central cities and aging of suburban populations. This population shift results in increased demand for HDM service in areas that are currently underserved by HDM kitchens.

This paper is organized as follows. We define HDM-LRP and summarize the related literature in Section 2. Section 3 summarizes the proposed Genetic Algorithm (GA). Next, computational results on benchmark problems are given in Section 4. Section 5 concludes and identifies a number of research extensions.

2 Problem Definition and Related Literature

The general planning problem that this paper addresses is the location-routing problem for home-delivered meals, referred to as HDM-LRP. In this problem,

the goal is to simultaneously choose “facility” (kitchen) locations that provide “products” (meals) to spatially dispersed customers via routes driven by multiple vehicles, each of which leave a facility, visit multiple customer locations, and then return to the same facility when customer deliveries are completed. Each customer, if served, must be served on a single route by a single kitchen. In addition to this, the duration of each route should not exceed the maximum allowable duration. Furthermore, the number of routes assigned to each kitchen should not exceed the maximum allowable number of routes; finally, a kitchen, if located, cannot serve more than its maximum capacity of meals.

Traditional LRPs (Perl and Daskin 1985, Laporte, Norbert and Taillefer 1988) have minimized the sum of fixed facility location costs, vehicle operating costs, and vehicle routing costs, the latter usually approximated by total distance travelled. If current demand cannot be met by a realistic set of kitchens, an effectiveness measure must be also optimized. Thus, we treat HDM-LRP as, fundamentally, a multi-objective planning problem: namely, minimizing the total cost, and maximizing the total demand served.

For the purposes of model simplicity, a number of assumptions are made. First, despite the known elderly population changes in the study areas over time, it is assumed that the planning horizon is a single period (but see results presented by Laporte and Dejax (1989) for a multi-period LRP). Second, while it is impractical to identify exact locations of all customers currently receiving HDM services in typical study areas, it is assumed that exact customer locations are known in the current planning period (but see results presented by Laporte, Louveaux and Mercure (1989) for a stochastic LRP).

LRP is a generalization of two already difficult problems: Facility Location Problem (FLP) and Vehicle Routing Problem (VRP). Locational decisions are usually made on a strategic level, whereas routing decisions are solved at an operational level. A mathematical formulation of HDM-LRP is given in Appendix A. This model, in the traditional three-index row-based style, is based on those of Perl and Daskin (1985) and Laporte (1988).

Both FLP and VRP have been shown to be NP-hard (Cornuejols et al. (1977) and Karp (1972), Lenstra and Rinnooy Kan (1981)), so the location routing problem is NP-hard as well. Due to its complexity, exact-solution approaches to LRP have been very limited. One of the earliest studies was an exact algorithm for the single facility LRP due to Laporte and Norbert (1981). They formulated the problem as an integer-linear program and used a constraint relaxation technique to solve it. Following this study, Laporte et al. (1983, 1986) used a similar approach to solve both the uncapacitated and capacitated multi-facility LRP. Bookbinder and Reece (1988) formulated a three-layer multi-commodity, capacitated distribution system as a nonlinear mixed integer program, and applied Benders’ decomposition method.

Due to the exponential growth in the problem size, exact methods for LRP have been limited to small and medium size instances. Indeed, Laporte, Norbert, and Taillefer (1988) have solved LRPs to optimality for at most 80 nodes. Berger (1997) reports optimal solutions to LRP for 25 facilities and 150 customers, under

the assumption that vehicles do not have to return to the facility after making deliveries.

The difficulty of LRP in general precludes exact techniques for realistically sized problems. However, researchers have developed a number of heuristic methods for solving LRP. These include location-allocation first, route second (Madsen 1983), improvement/exchange (Or and Pierskalla 1979), and a combination of solutions to multiple related combinatorial optimization problems (Perl and Daskin 1985), (Hansen et al. 1994). In recent years metaheuristics are increasingly used to solve LRPs. These include tabu search (Renaud, Laporte and Boctor 1996), nested heuristic methods with tabu search (Nagy and Salhi 1996), two-phase tabu search (Tuzun and Burke 1999), simulated annealing (Wu, Low and Bai 2002), and threshold accepting and simulated annealing (Lin, Chow and Chen 2002). The literature is inconclusive as to the relative efficacy of these alternative approaches. In this paper, HDM-LRP is solved using a Genetic Algorithm and this appears to be the first study that uses GAs to solve LRP.

Typically, an approach to solving LRPs is to first decompose the larger problem into subproblems, and then solving these subproblems either sequentially or iteratively. We are not aware of any explanation about this preference other than the computational inefficiency of dealing with LRP as a whole. However we think that another reason might be that all of the heuristics proposed for solving LRP are local search variants. Thus, it is necessary to define a proper representation of a solution and a neighborhood structure. This is straightforward for the subproblems FLP and VRP, but finding a neat way of representing solutions and neighborhood structures for the LRP is less clear. This may be due to the fact that the decisions for the two subproblems are, by definition, unrelated; the idea of decomposition also seems quite intuitive. Moreover, a significant amount of literature already exists for the subproblems. In this paper, we use a holistic approach as opposed to the decomposition approach.

A review of the relevant literature finds scant research on the location-routing model being applied to the home-delivered meals planning problem. In their study, Johnson, Gorr, and Roehrig (2002) present an interactive GIS-based heuristic. However, there has been limited research in the more general area of HDM planning. Bartholdi, Platzman, Collins, and Warden (1983) have presented low-technology approaches to designing HDM delivery routes for individual kitchens, but did not address the problem of designing an entire network of HDM kitchens and delivery drivers. Wong and Meyer (1993) developed a spatial decision support system for HDM-LRP, but in using an location-allocation-first, route-second approach, their research does not address the multi-kitchen planning problem from an LRP perspective.

3 Solution Strategy

As discussed in the previous sections, exact methods for LRP are computationally impractical for the realistic size problems. There remains a need for effective computational approaches to solve large LRPs. Metaheuristics, such as

tabu search, genetic algorithms, simulated annealing, neural networks, and ant colony optimization, are widely used to solve important practical combinatorial optimization problems. All of these metaheuristics aim to search the solution space more effectively than conventional approaches. They show great promise in solving difficult combinatorial problems such as LRP.

Among the metaheuristics mentioned, GA is an adaptive heuristic search method based on population genetics, and it borrows its vocabulary from that domain. The basic concepts of a GA were primarily developed by Holland (1975) and described later by Goldberg (1989). The GA consists of a population of chromosomes; in essence, a set of character strings that are analogous to the base-4 chromosomes that we see in our own DNA that evolve over a number of generations and are subject to genetic operators at each generation. Each chromosome represents a potential solution to the problem being solved. This solution is obtained by means of an encoding/decoding mechanism. Most of the developmental work of GA theory was performed using a binary-coded GA, and, historically, is the most widely used representation. In a binary coding each chromosome is a vector comprised of zeroes and ones, where each bit represents a gene. However different encodings are also possible.

Initially, a feasible set of chromosomes are needed, which can be generated randomly or by using a heuristic. Each chromosome has an associated *fitness value*; a set of “best fit” chromosomes from each generation survive into the next generation. The genetic operations that the chromosomes are subjected to are *crossover* and *mutation*. Typically, crossover is defined so that two individuals (the parents) combine to produce two more individuals (the children). But asexual crossover or single-child crossover are defined as well, which we refer to as “reproduction” in this paper. The primary purpose of the crossover operator is to transmit genetic material from the previous generation to the subsequent generation. Mutation is a genetic operator that alters one or more gene values in a chromosome from its initial state. The mutation operator introduces a certain amount of randomness to the search. It can help the search find solutions that crossover alone might not encounter.

In this study, GAs are chosen to solve HDM-LRP for two reasons. First, as opposed to the other metaheuristics, GA uses a population of solutions in each iteration, instead of a single solution. Thus, the outcome of a GA is also a population of solutions, making GAs suitable for solving multi-objective optimization problems. Since the ideal strategy for multi-objective optimization requires multiple trade-off solutions to be found, a GA’s population-approach can be suitably utilized so that it finds multiple solutions in a single simulation run. Second, a review of the relevant research does not find any models where GAs have been applied to LRP, but they have been successfully applied to its two subproblems, namely Facility Location (Correa, et al. 2001), and Vehicle Routing (Tavares, et al. 2003).

A GA for a particular problem must have the following five components, as stated by Michalewicz (1996):

- a genetic representation for potential solutions to the problem,

- genetic operators that alter the composition of children,
- a way to create an initial population of potential solutions,
- an evaluation function that plays the role of the environment, rating solutions in terms of their “fitness”, and
- values for various parameters that the GA uses (population size, probabilities of applying genetic operators, etc.)

3.1 Representation and Genetic Operators

This paper proposes two different representation schemes with corresponding genetic operators for LRP, namely Facility-Route(F-R) Representation and Binary Representation.

Facility-Route (F-R) Representation

A set of possible facility locations $\{1, \dots, N\}$ and a set of customers $\{N+1, \dots, M+N\}$ are given. If facility i is open, the corresponding entry is 1, otherwise the entry is 0. In addition, there is a segment for each open facility where the first entry is the facility number, and the following entries are the customers on a route, in order, followed by the same facility number at the end. If there is more than one route at the same facility, each is separated by the facility number. The following example demonstrates the representation of two solutions:

Example 1: Given four facility locations: $\{1, 2, 3, 4\}$, and ten customers: $\{5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$

Solution 1:

Facility 1: Route 1: 5-6-7; Route 2: 9-11

Facility 2: Closed

Facility 3: Route 1: 8-13-12-14-10

Facility 4: Closed

The F-R representation is as follows:

```
1 0 1 0
1 5 6 7 1 9 11 1
3 8 13 12 14 10 3
```

Solution 2:

Facility 1: Closed

Facility 2: Route 1: 12-7-10 Route 2: 13-14

Facility 3: Route 1: 8-9-6

Facility 4: Route 1: 5-11

F-R representation is as follows:

```
0 1 1 1
2 12 7 10 2 13 14 2
```

3 8 9 6 3
 4 5 11 4

The actual network for Solution 1 and Solution 2 are given in Fig. 1 and Fig. 2, respectively.

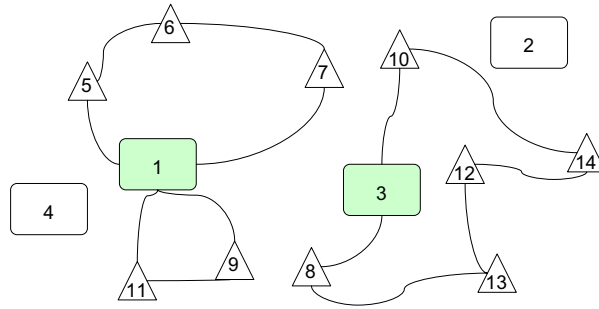


Fig. 1. Actual network for Solution 1. Rectangles represent the facilities and the triangles represent the customers. Shaded facilities(1 and 3) are open, the rest is closed. There are three routes: 1-5-6-7-1, 1-9-11-1, and 3-8-13-12-14-10

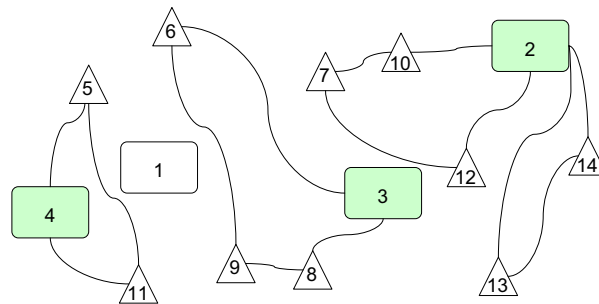


Fig. 2. Actual network for Solution 2. Rectangles represent the facilities and the triangles represent the customers. Shaded facilities(2,3 and 4) are open, 1 is closed. There are four routes: 2-12-7-10-2, 2-13-14-2, 3-8-9-6-3, and 4-5-11-4

F-R Genetic Operators

Below are two types of crossovers, two types of reproductions, and two types of mutations.

Facility Crossover: Two solutions are taken, and a one-point crossover is applied, where a crossover point is randomly selected and all the data beyond that point is swapped between the two parent chromosomes, using the binary genes corresponding to the facilities. Duplicate customers are eliminated based on a predetermined dominance rule among the two solutions for a particular offspring. This operator does not however guarantee to generate offspring that serve all the customers, even if the parents serve all the customers. This operator is demonstrated in the following example:

Example 2: Given the two solutions in Example 1, and assume that the crossover point is selected after the second facility and also assume that the parent that defines the first facility for an offspring is the dominant one, the two resulting offspring are:

Offspring 1:

1 0 1 1
 1 5 6 7 1 9 11 1
 3 8 3
 4

Offspring 2:

0 1 1 0
 2 12 7 10 2 13 14 2
 3 8 3

Note that customers 10, 12, 13, and 14 are not served in the first offspring, and customers 5, 6, and 9 are not served in the second offspring.

It is not hard to see that if the parents are feasible, then the offspring generated with this operator are also feasible. The reason is as follows. Each facility that is open in an offspring is also open in the parent that gave the genetic material corresponding to that facility. Moreover, no extra customers are allocated to the facilities. Thus, the capacity constraints for the facilities in the offspring are all satisfied. In addition, the length of all the routes of an offspring are no longer than the corresponding routes in the parents, since the customers either being kept in their original routes or being removed as duplicates. Finally, the number of routes for each facility never increases in an offspring since there is no route exchange between the facilities and also no new route is generated. However, some existing routes from the non-dominant parent may be removed if all the customers on that route are already inherited from the dominant parent.

Facility Reproduction: This operator does not promote a mutual exchange of genetic material between two parents, but it operates in the following way: When a solution is submitted to this kind of operation, it receives a fragment of genetic material from another parent, and inserts it as an initial set of open

facilities and routes. The fragment must consist of facilities that are not already open in the receiving solution. After insertion, a repair process checks the original facilities and routes of the receiving individual and removes all customers that also appear in the inherited material. The donor is not modified. This operator can be considered as a special case of facility crossover, except the fact that it maintains coverage of all customers served by either of the parents. This operator is demonstrated in the following example:

Example 3: Given Solutions 1 and 2 of Example 1, and assume that Solution 2 is the donor that gives its second and fourth facilities, then the offspring created is as follows:

Offspring 3:
 1 1 1 1
 1 6 1 9 1
 2 12 7 10 2 13 14 2
 3 8 3
 4 5 11 4

This operator also generates a feasible offspring from feasible parents. Every facility (and the routes of it) that is received from the donor are absolutely untouched, so they remain feasible. The facilities from the receiver might have customers being removed from them if the same customers are already inherited from the donor. But this can only decrease the amount of demand served from those facilities, shorten the routes and decrease the number of routes. Thus, feasibility is preserved.

Route Crossover: Take two solutions, and swap the routes. Keep the exchange happen among the same set of facilities. Similar to the Facility Crossover, this operator does not guarantee to generate offspring that serve all the customers even if the parents serve all the customers. This operator is demonstrated in the following example.

Example 4: Swap the route of facility 3 (of Solution 2) with the first route of facility 1 (of Solution 1), and swap the route of facility 4 (of Solution 2) with the second route of facility 1 (of Solution 1). The two resulting offspring are as follows:

Offspring 5:
 1 0 1 0
 1 8 9 6 1 5 11 1
 3 13 12 14 10 3

Offspring 6:
 0 1 1 1

```
2 12 10 2 13 14 2
3 5 6 7 3
4 9 11 4
```

If the parents are feasible, then the offspring generated with this operator might only violate the capacity constraints of some facilities. However this can be avoided by checking if the capacity of a facility is violated before allowing the exchange. The other constraints are not violated. The reason is as follows. The length of all the routes of an offspring are no longer than the corresponding routes in the parents, since the customers either being kept in their original routes or being removed as duplicates. The number of routes for each facility never increases in an offspring since route exchanges are only allowed between the same set of facilities.

Route Reproduction: Similar to the Facility Reproduction, this operator also does not promote a mutual exchange of genetic material between two parents. A route is taken from the donor and attached to the closest feasible facility as a new route. After that, a repair process checks the original routes of the receiving individual and removes all customers that also appear in the inherited material. The donor is not modified. This operator can be considered as a special case of route crossover, except the fact that it maintains coverage of all customers served by either of the parents. This operator is demonstrated in the following example.

Example 5: Take Solutions 1 and 2, and assume that Solution 1 is the donor that gives its two routes of its first facility; assume that the first route is attached to facility 3 and second route is attached to facility 4 of Solution 2. The resulting offspring is as follows:

```
Offspring 7:
0 1 1 1
2 12 10 2 13 14 2
3 8 3 5 6 7 3
4 9 11 4
```

If the parents are feasible, then the offspring generated with this operator might violate the capacity constraints of some facilities. However this can be avoided by checking if the capacity of a facility is violated before allowing the exchange. Similarly the routes are not allowed to be attached to facilities that already has the maximum number of routes allowed. The route length constraints are not violated since the length of all the routes of an offspring are no longer than the corresponding routes in the parents.

Mutations

Customer Swap: Select two customers and swap them. Selected customers can

belong to the same route or to different routes.

Displacement: Select a sub-route and insert it into another place.

Clearly these operators might create an infeasible solution when applied to a feasible solution, so we need to use them with caution.

Binary Representation

In this representation, each customer is represented by a binary string of length $P + Q + R$. From right to left, the first P bits are reserved for the facility the customer is allocated. The next Q bits represent the route the customer is on. Finally the last R bits are reserved for the position of the customer on the route. Thus, the total length of a solution is $N(P + Q + R)$. The binary representation of Solution 1 is given in Fig. 3.

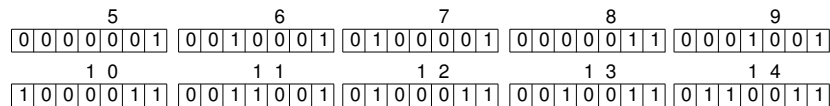


Fig. 3. Solution 1 in binary representation. From right to left, the first $P=3$ bits are reserved for the facility where the customer is allocated. The next $Q=1$ bit is reserved for the route that the customer is on. If it is 0, the customer is on route 1; if it is 1, the customer is on route 2. Finally the last $R=3$ bits are reserved for the position of the customer on the route.

Binary Genetic Operators

Below are two types of crossovers and a mutation for the Binary Representation.

One Point Crossover: A crossover point is randomly selected. All data beyond that point is swapped between the two parent chromosomes. The resulting chromosomes are the children. This operator is demonstrated in the following example. Notice that the examples provided for the binary genetic operators are separate from the LRP examples examined earlier.

Example 6:

Parent 1: 11001|010
 Parent 2: 00100|111

After interchanging the parent chromosomes at the crossover point, the following

offspring are produced:

Offspring 8: 11001|111

Offspring 9: 00100|010

The offsprings generated by this operator is likely to generate infeasible solutions. The reason is as follows. If a crossover point is selected such that the customers to the left of the point from parent 1 and to the right of the point from parent 2 are concentrated on some subset of facilities and/or routes, then the capacity constraint on those facilities and route length constraint on the routes might be violated. The remaining constraint, number of routes at each facility, might be violated only if the crossover point happens to be in the segment reserved for the route number of a customer. In that case, the number of routes of the facility that serves that customer might exceed the allowed limit.

Two Points Crossover: Two points are randomly selected. Everything between the two points is swapped between the parent chromosomes, rendering two child chromosomes. This operator is demonstrated in the following example.

Example 7:

Parent 3: 11|001|010

Parent 4: 00|100|111

After interchanging the parent chromosomes at the crossover point, the following offspring are produced:

Offspring 10: 11|100|010

Offspring 11: 00|001|111

This operator also is likely to generate infeasible offsprings. The reason is very similar to the reason explained for the one point crossover.

Mutation: This simply inverts the value of the chosen gene (0 goes to 1 and 1 goes to 0).

Comparison of F-R and Binary Representations

One of the most desirable properties of genetic operators is to produce feasible offspring from feasible parents. If the genetic operators do not have this property, then the infeasible solutions must be addressed by some mechanism, such as penalizing infeasibilities, removing infeasible solutions from the population, etc.

As discussed in the corresponding sections, the Facility Crossover and Reproduction operators of F-R Representation has this desirable property. Although Route Crossover and Reproduction operators may create infeasible solutions

when applied without any caution, it is not hard to deal with this before creating infeasible offspring. However, the crossover operators of Binary Representation do not have this desirable property. This makes F-R Representation more attractive than the Binary Representation.

In terms of difficulty in implementing the operators, F-R operators are harder than the Binary operators. However, since the feasibility issue is more important when compared to the easiness of implementation, the F-R representation is chosen for implementation.

3.2 Initial Population of Solutions

The initial set of solutions are formed by using two heuristics. The first one is a greedy heuristic. Starting from a possible facility location, the closest customer is added to the route until the time limit for that route is exceeded. Then, another route starts from the same facility. This is repeated until the maximum number of routes allowed for the facility is reached. After that, the same procedure is repeated for another possible facility location until all customers are visited. This heuristic is run as many times as needed with a different sequence of possible facility considerations.

The second heuristic works as follows: A random set of facility locations is selected to be open. Then, customers are assigned to the nearest open facility. When all the customers are assigned, routes are formed at each open facility using the nearest neighbor approach.

3.3 Evaluation Function

We consider two objective functions for the HDM-LRP. First one is minimizing the total cost. The other objective is maximizing the number of customers served, or equivalently minimizing the number of customers not served. For the single objective version of the problem we only consider the first objective. For the bi-objective version of the problem we take both objectives and combine them to create the following evaluation function. The fitness of a solution is equal to the total cost divided by the number of customers served. Thus, the fitness is the cost per customer served. A solution with a low fitness value is considered a better solution.

One might argue that such a fitness value might result in a very low demand satisfaction in the optimum solution. In fact the best solution is not serving any customers at all, which results in a cost of zero. To prevent this, we require at least one of the facilities to be open. Since there is a fixed facility cost associated with each facility, once a facility is opened, increasing the demand served from that facility would result in a decrease in the cost per demand since the fixed cost will be shared by more customers. The optimum solution, therefore, balances the two objectives.

3.4 Additional Features

In order to increase the quality of the solutions we have added two more features to the GA, which we explain in this section.

Route Improvement: After generating the initial population, a TSP heuristic is used to improve the vehicle routes of the solutions. The heuristic algorithm is due to Lin (1965) and the C code that uses Lin’s algorithm is due to Dunn (2005). This heuristic is also used to improve the routes of the populations generated later in the execution of the GA. Since applying the improvement heuristic at each generation will be time consuming, it is only applied to some of the generations. For instance one option is applying the route improvement to populations at every 50th generation. The frequency of the improvement is a parameter that can be adjusted by the user.

Annealing Acceptance for Mutation Operators: Given that the mutation operators cause random changes in the solutions, these changes might deter a good solution and create an inferior solution. This is acceptable during the initial generations but as the GA proceeds and the population evolves, mutations that increase the cost of a solution seem to be harmful for the population. This observation lead us to use an annealing type of approach to control the mutation operators. The increase in the cost due to a mutation is limited by a parameter called In order to test our algorithm on benchmark instances and compare our results with previous results, “tolerance”, which decreases as the number of generations increase. If the increase in the cost exceeds the tolerance, the new solution obtained after a mutation is rejected and the original solution is restored. The tolerance parameter is decreased at each iteration using an exponentially decaying function dependent on the generation number. Thus, after several generations, the tolerance becomes close to zero. This prevents almost any of the mutations to be executed, given that the probability of mutation is low. To balance this effect, the probability of executing a mutation operator is increased exponentially as the number of generations increase. So as the number of generations increase the algorithm tries to mutate more solutions and at the same time the proportion of mutations accepted becomes less as the tolerance become closer to zero.

3.5 The Algorithm

1. Set $t = 0$
2. Form $N_{initial}$ solutions to form the set $CurrentSolutions(t)$.
3. Optimize Routes of solutions in $CurrentSolutions(t)$.
4. While the *Stopping Criteria* is not met do:
 - (a) Set $t = t + 1$
 - (b) Create $N_{initial}/2$ new solutions by:
 - i. Randomly select two solutions from the current population of solutions and designate one of them as donor and the other as receiver.

- ii. Randomly pick one of the Reproduction (or Crossover) Operators, according to their corresponding probabilities, and apply it on the two selected solutions to create a(two) new solution(s).
- iii. If the new solution(s) created exist in $TempSolutions(t)$ or in $CurrentSolutions(t-1)$, discard it and goto Step 4(b)i. Otherwise add new the solution(s) into the set of $TempSolutions(t)$
- (c) For every solution S in $TempSolutions(t)$ and $CurrentSolutions(t-1)$ do:
 - i. Apply Displacement(Customer Swap) to S with $DisplacementProbability$ ($CustomerSwapPropability$) and create S' . If $Cost(S') - Cost(S) \leq Tolerance$, replace S with S' . Otherwise discard S' and restore S .
- (d) Replace the inferior solutions in $CurrentSolutions(t-1)$ with the superior solutions in $TempSolutions(t)$ and create $CurrentSolutions(t)$
- (e) If $mod(t, RouteOptFrequency) = 0$ Optimize Routes of solutions in $CurrentSolutions(t)$..

4 Experimental Results

In this section, first the details of the implementation are explained. Then, a description of the test data sets is given. Finally, computational results obtained are presented.

4.1 Implementation Details

The implementation and experimentation is divided into two phases. The first phase is the ‘‘LRP Phase’’, where the aim is to test the proposed GA on some instances available in the literature and compare the results with previous works’ results. After comparing the results obtained by the GA with the previous works’ results, the next step is to go on to the ‘‘HDM-LRP Phase’’, where there does not exist any benchmark instances and there exist only one previous study. The experimental results given in this paper are the result of the LRP Phase. The results for the HDM-LRP Phase are deferred to an upcoming study on HDM.

In order to test our algorithm on benchmark instances and compare our results with previous results, we modified some of the characteristics of HDM-LRP. The basic characteristics of the problem considered in this paper is as follows: The objective is only to minimize the total cost. Full demand satisfaction is imposed as a constraint. We impose a vehicle capacity constraint that restricts the length of each route instead of the time limit constraint of HDM-LRP. Moreover, as opposed to HDM-LRP, there is no limit on the number of the vehicles available.

Since the problems considered is single objective, and full demand satisfaction is a constraint, the crossover operators described in earlier sections are not suitable. Because these operators do not guarantee to generate offspring that serve all customers. Thus, only the reproduction operators are used to solve the LRP instances.

The parameters of the GA have been set empirically: We used an initial population size $N_{initial} = 500$. At each generation we created 250 new solutions by applying Reproduction Operators to the current solutions. At iteration t , a new solution is created by using Facility Reproduction is used with probability $0.5/t^{0.25}$ and by using Route Reproduction with probability $1 - 0.5/t^{0.25}$. Displacement is applied to a solution with $DisplacementProbability = 0.1 * t^{0.33}$ and Customer Swap is applied with $CustomerSwapPropability = 0.2 * t^{0.33}$. A mutation is accepted if the change in the cost of the solution do not exceed $Tolerance$, which is initially set equal to %2 of the cost of best solution in the initial population, $CurrentSolutions(0)$. For $t > 0$, $Tolerance = Tolerance/t^{0.5}$. The algorithm is terminated if the best solution cost is not improved in 50 consecutive generations.

4.2 Instance Description

The first set of instances that the GA is tested are due to Tuzun and Burke (1996). These instances are randomly generated with different characteristics such as problem size, spatial distribution of customers and route structure. The set is composed of instances with number of customers equal to 100, 150 or 200 and number of potential facilities equal to 10 or 20.

The second set of instances are obtained from the compilation of Barreto (2005). These instances are due to many different authors from various articles in the literature. We tested the GA on the larger instances, where the number of potential facilities is at least 8 and the number of customers is at least 75.

4.3 Summary of Results

In order to demonstrate the difficulty of the problem, we first formulated the LRP as a mixed-integer-program (MIP) and run it on OPL studio 3.7, where CPLEX MIP solver is used to solve the problem. We compared the results obtained by the MIP with the results of the GA. We then tested the GA on benchmark instances described in Section 4.2 and compared the results with the results obtained by other algorithms. We performed our computational tests an on a PC with 1.6 GHz Pentium M Processor and 1GB of RAM.

Comparison of GA with the Mixed Integer Program

The instance that we solved is one of the smallest instances this paper considers and is due to Christofedes et. al(1969) with 75 customers and 10 possible facility locations. This instance is also one of the problems solved by the GA, where the related results are demonstrated in Table 1. The MIP formulation has 68850 constraints and 86786 variables. The computer ran out of memory after 11 hours of execution. The MIP could not even find a feasible solution before termination. The lower bound it found(485.7) is far less than the current known best lower bound (744.7).

After having this experience on one single instance, we tested the performance of MIP on the subproblems of this instance. Starting from a subproblem with 10 customers and 3 potential facilities, we increased the number of customers and facilities to create new instances. The results are demonstrated in Table 1. Except for the smallest instance, MIP could not obtain an optimal solution for none of the instances, before the computer ran out of memory. The gap between the best feasible solution found and the best lower bound increases as the instance size increases. The GA found better solutions than the MIP for all the subproblems except the smallest two. In those two smallest subproblems, the difference between the costs is very small. For the smallest size subproblem, the solution found by the GA has a cost almost equal to the optimal cost. In terms of time, the GA is multiple times faster than the MIP.

Problem	n	m	vc	CPLEX					Genetic Algorithm	
				Variables	Constraints	Cost (Best Sol.)	Lower Bound	Time(sec)	Cost	Time(sec)
Ch69Cli75x10	75	10	140	86786	68850	--	485.6	44234.54	851.72	59.63
Ch69Cli45x6	45	6	140	23461	18844	964.5	353.68	26378.71	600.48	28.87
Ch69Cli30x4	30	4	140	6971	5691	569.3	296.25	17944.23	459.77	1.64
Ch69Cli15x4	15	4	140	2186	1536	292.9	191.55	13527.85	282.61	0.21
Ch69Cli15x3	15	3	140	991	772	282.2	209.9	12187.45	287.85	0.13
Ch69Cli10x3	10	3	140	351	258	219.5	219.5	109.39	221.99	0.07

Table 1. The columns in the table consist of the name of the instance, number of customers (n), number of potential facilities (m), vehicle capacity (vc), number of variables of the IP, the number of costs of the IP, cost of the best feasible solution found by CPLEX, amount of time CPLEX ran, cost of the solution found by the GA, and amount of time the GA ran.

Comparison of the GA with Other Algorithms

Table 2 presents the results of the GA and two other heuristics, SAV1 and Tabu Search, on the first problem set. The results for the SAV1 and Tabu Search (TS) are due to Tuzun and Burke (1996). In their study, they implemented the SAV1 heuristic, which was initially proposed by Srivastava (1993), to compare their Two-Phase TS Algorithm. The SAV1 heuristic, also known as savings heuristic, assumes all potential facilities to be open initially, and uses approximate routing costs for open facilities to determine the facility to be closed. The TS algorithm, on the other hand, coordinates two tabu search mechanisms. One seeking a good facility configuration, the other a good routing that corresponds to this configuration.

The GA found better solutions than the SAV1 for all but one of the instances. The average improvement of GA over the SAV1 is %3.5. The GA obtained better solutions than the Tabu Search for 28 instances and obtained a solution with the same cost for one instance and worse solutions for the remaining 7 instances. The average improvement of GA over the Tabu Search is %1.2. In

Problem	n	m	SAV1 cost	SAV1 time	TS cost	TS time	GA cost	GA time	% Impr. Over SAV1	% Impr. Over TS
P111112	100	10	1596.0	1.0	1556.6	5.0	1498.7	179.8	6.1	3.7
P111122	100	20	1557.1	3.0	1531.9	3.0	1480.5	83.9	4.9	3.4
P111212	100	10	1457.0	1.0	1443.4	3.0	1428.8	167.3	1.9	1.0
P111222	100	20	1555.9	3.0	1511.4	4.0	1481.3	200.8	4.8	2.0
P112112	100	10	1245.8	1.0	1231.1	4.0	1206.5	62.6	3.2	2.0
P112122	100	20	1140.1	3.0	1132.0	2.0	1127.1	351.0	1.1	0.4
P112212	100	10	830.5	1.0	825.1	3.0	787.3	44.4	5.2	4.6
P112222	100	20	744.7	3.0	740.6	3.0	741.3	189.9	0.5	-0.1
P113112	100	10	1326.1	1.0	1317.0	3.0	1278.4	139.0	3.6	2.9
P113122	100	20	1316.5	3.0	1274.5	4.0	1259.1	248.2	4.4	1.2
P113212	100	10	927.2	1.0	920.8	4.0	917.2	368.4	1.1	0.4
P113222	100	20	1109.3	3.0	1042.2	3.0	1038.1	128.2	6.4	0.4
P131112	150	10	2066.2	3.0	2001.0	12.0	1986.2	429.7	3.9	0.7
P131122	150	20	1977.4	8.0	1892.8	12.0	1887.5	293.3	4.5	0.3
P131212	150	10	2113.6	2.0	2022.1	14.0	2037.7	435.0	3.6	-0.8
P131222	150	20	1931.6	8.0	1855.0	13.0	1856.2	145.5	3.9	-0.1
P132112	150	10	1653.4	4.0	1555.8	9.0	1479.4	565.1	10.5	4.9
P132122	150	20	1554.5	9.0	1478.8	12.0	1498.8	380.3	3.6	-1.4
P132212	150	10	1238.0	2.0	1231.3	9.0	1230.7	637.6	0.6	0.1
P132222	150	20	953.3	9.0	948.3	9.0	945.5	648.9	0.8	0.3
P133112	150	10	1848.4	3.0	1762.5	9.0	1755.3	618.5	5.0	0.4
P133122	150	20	1496.4	7.0	1488.3	9.0	1442.0	366.6	3.6	3.1
P133212	150	10	1247.3	3.0	1264.6	10.0	1243.4	263.5	0.3	1.7
P133222	150	20	1192.6	8.0	1182.3	9.0	1182.4	652.9	0.9	0.0
P121112	200	10	2463.2	5.0	2379.5	22.0	2349.9	162.9	4.6	1.2
P121122	200	20	2289.4	15.0	2211.7	22.0	2224.6	373.9	2.8	-0.6
P121212	200	10	2395.9	4.0	2288.2	23.0	2292.8	425.1	4.3	-0.2
P121222	200	20	2417.3	19.0	2355.8	26.0	2320.0	448.3	4.0	1.5
P122112	200	10	2203.4	9.0	2158.6	20.0	2130.6	692.0	3.3	1.3
P122122	200	20	1805.9	20.0	1787.0	18.0	1775.0	633.6	1.7	0.7
P122212	200	10	1560.8	8.0	1549.8	18.0	1504.9	508.6	3.6	2.9
P122222	200	20	1122.9	19.0	1113.0	18.0	1110.8	680.1	1.1	0.2
P123112	200	10	2312.8	5.0	2056.1	23.0	2010.5	370.3	13.1	2.2
P123122	200	20	2046.5	15.0	2002.4	20.0	1999.3	483.3	2.3	0.2
P123212	200	10	1849.8	6.0	1877.3	20.0	1833.1	640.3	0.9	2.4
P123222	200	20	1423.8	15.0	1414.8	17.0	1429.0	184.0	-0.4	-1.0
Average			1610.3	6.4	1566.8	11.5	1549.2	366.7	3.5	1.2

Table 2. Results of the GA on the first set. The columns in the table consist of the name of the instance, number of customers (n), number of potential facilities (m), cost found and time in seconds for SAV1, Tabu Search (TS) and Genetic Algorithm (GA), % improvement obtained by the GA over SAV1 and TS.

terms of solution time, the GA is a slower algorithm than the SAV1 and the Tabu Search Algorithms. Considering that the GA generates a population of solutions, the increased amount of time needed by the GA is not surprising and can be considered reasonable. Moreover, the solution times for the GA are all less than twelve minutes and the average time for the GA is approximately 6 minutes, which makes the GA a practical algorithm.

Table 3 presents the results of the GA on the second problem set. The GA obtained solutions that have lower costs than the current best upper bounds for all six instances. The average improvement of GA over the upper bounds is %3.5.

Problem	n	m	vc	LB	UB	GA	% Imp.	time
Christofides69-75x10	75	10	140	744.7	886.3	851.7	3.9	59.6
Daskin95-88x8	88	8	9000000	356.4	384.9	375.7	2.4	64.3
Christofides69-100x10	100	10	200	788.6	889.4	868.2	2.4	69.1
Or76-117x14	117	14	150	12048.4	12474.2	12071.9	3.2	451.0
Min92-134x8	134	8	850	-	6238.0	6040.5	3.2	34.0
Daskin95-150x10	150	10	8000000	43406.0	46642.7	44011.3	5.6	129.6
Average							3.5	

Table 3. Results of the GA on the second set. The columns in the table consist of the name of the instance, number of customers(n), number of potential facilities(m), currently known lower(LB) and upper bounds(UB), cost found by the GA,% improvement obtained by the GA over the UB, and the time in seconds(time) for the GA.

The Bi-Objective LRP

Although the focus of the experimental results in this paper is for the single objective version of the LRP, we want to briefly discuss the bi-objective version of the problem as well. As stated in Section 3, the GA has the advantage of generating a population solutions after a single run. Thus, for a multi-objective problem, it gives a decision maker the opportunity to choose from a set of solutions with different objective values. This is especially crucial for public sector problems, such as the HDM problem considered in this study, where a decision maker has to consider budget restrictions as well as customer service levels. In order to demonstrate this advantage of the GA, we run it using the Facility Crossover operator described in Section 3.1, along with the Reproduction and Mutation operators, on one instance from the first set of instances, namely the P122112. The tradeoff curve for this instance is illustrated in Fig. 4. To obtain this curve, the GA kept only the non-dominated solutions at each generation. A solution S_1 is called *dominated* if there exists a solution S_2 that has a lower cost than or serves more customers than S_1 . There are a few points worth mentioning about this trade-off curve:

- the overall shape of the curve resembles a hyperbola, where the rate of decrease in total cost decreases as the number of unserved customers increase,
- the graph itself consists of several small curves that have a similar shape with the graph. Moreover, there is significant cost change at the end of these segments, which is most likely due to a difference in the number of open facilities at the two neighboring end-of-segment solutions.
- the total cost for the solution that serves all customers is 2179.54. The costs found by the SAV1, TS and the GA by solving the single objective version of LRP are 2203.4, 2158.6, and 2130.6, respectively. So the solution found by the bi-objective GA is better than the cost of SAV1 and has a cost close to the costs of the TS and single-objective GA.

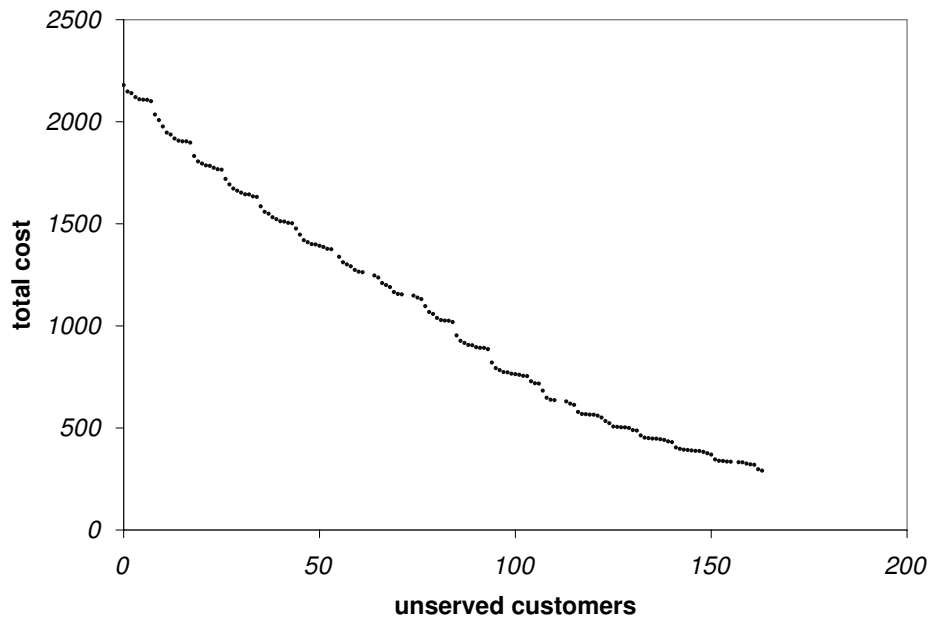


Fig. 4. The trade-off curve for instance P122112, with $n=200$ and $m=10$.

5 Conclusion

This study proposes a Genetic Algorithm for the Home-Delivered Meals Location-Routing Problem. Although several metaheuristics are applied to LRP, this appears to be the first study that uses GAs to solve the LRP.

Two different representations are proposed, namely Facility-Route and Binary Representations with corresponding genetic operators. The F-R represen-

tation is tested on two sets of benchmark problems with the resulting computational results presented. We have shown that the GA performs better than the SAV1 and Tabu Search methods for most of the instances in the first set and it finds solution that have a cost lower than the current best upper bound for the instance in the second set.

We also demonstrated the trade off curve for one instances for the bi-objective version of the problem. We argue that such curves are very useful for decision makers who has to plan for Home-Delivered Meals or similar services that can be formulated as a LRP. The tradeoff curves helps them to see the tradeoff between the level of demand satisfaction and the total cost and gives them the opportunity to pick a solution they think more appropriate. The quality of the solution obtained by the the GA for both the single objective and bi-objective LRPs is very promising.

Future research directions for this research includes to fully test the GA, by using all the operators, for the multi-objective problem and then apply the GA on larger, real-world data sets.

References

- Barreto, S. (2007) Location-Routing Problems:
<http://sweet.ua.pt/iscf143/private/SergioBarretoHomePage.htm>
- Bartholdi, J.J., III, Platzman, L.K.(1982). An $O(n \log n)$ planar travelling salesman heuristic based on spacefilling curves. *Operations Research Letters*, 1(4): 121 - 125.
- Bartholdi, J.J., III, Platzman, L.K., Collins, R.L. and Warden, W.H. III. (1983). A Minimal Technology Routing System for Meals on Wheels. *Interfaces*, 13(3): 1 - 8.
- Berger, R. (1997). Location-Routing Models for Distribution System Design. Unpublished dissertation. Northwestern University, Department of Industrial Engineering and Management Sciences, Evanston, Ill., USA.
- Bookbinder, J.H., and Reece, K.E. (1988). Vehicle routing considerations in distribution system design. *European Journal of Operational Research*, 37, 204 - 213.
- Cornuejols, G., Fisher, M.L., and Nemhauser, G.L., (1977). Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Science*, 23(8): 789 - 810.
- Correa, E.S., Steiner, M.T.A., Freitas, A.A., and Carnier, C.(2001). A Genetic Algorithm for the P-Median Problem. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pp. 1268 - 1275, San Francisco, USA, July 2001.
- Deb, K. (2001) Multi-objective optimization using evolutionary algorithms. Chichester, UK: John Wiley.

Dunn, K.L. (April, 2005) Euclidian Traveling Salesman Problem Solver: <http://fly.hiwaay.net:8000/kdunn/problems/tsp.shtml>

Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning. New York: Addison Wesley.

Hansen, P.H., Hegedahl, B., Hjortkjær, S., and Obel, B. (1994). A heuristic solution to the warehouse location-routing problem. *European Journal of Operational Research*, 76 (1), 111-127.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Johnson, M.P., Gorr, W.P., and S. Roehrig. (2002). Location/Allocation/Routing for Home-Delivered Meals Provision: Models and Solution Approaches. *International Journal of Industrial Engineering* 9(1): 45 - 56.

Karp, R., (1972). Reducibility among combinatorial problems, in: Miller, R., Thatcher, J.(Eds.), *Complexity of computer Computations*. Plenum Press, New York, pp.85-104.

Laporte, G. (1988). Location-Routing Problems. *Vehicle Routing: Methods and Studies*. (Eds.: B.L. Golden and A.A. Assad), North-Holland, Amsterdam, pp. 163 - 197.

Laporte, G. and Dejax, P.J. (1989). Dynamic Location-Routing Problems. *Journal of the Operational Research Society*, 40(5): 471 - 482.

Laporte, G., Louveaux, F. and Mercure, H. (1989). Models and Exact Solutions for a Class of Stochastic Location-Routing Problems. *European Journal of Operational Research*, 39: 71 - 78.

Laporte, G., and Norbert, Y. (1981). An exact algorithm for minimizing routing and operating costs in depot location. *European Journal of Operational Research*, 6: 224 - 226.

Laporte, G., Norbert, Y., and Arpin, D. (1986). An exact algorithm for solving a capacitated location-routing problem. *Annals of Operations Research*, 6: 293 - 310.

Laporte, G., Norbert, Y., and Pelletier, P. (1983). Hamiltonian location problems. *European Journal of Operational Research*, 12: 80 - 87.

Laporte, G., Norbert, Y. and Taillefer, S. (1988). Solving a Family of Multi-Depot Vehicle Routing and Location-Routing Problems. *Transportation Science*, 22(3): 161 - 172.

Lenstra, J.L. and Rinnooy Kan, A.H.G. (1981). Complexity of Vehicle Routing and Scheduling Problems. *Networks*, 11: 221 - 227.

- Lin, C.K.Y., Chow, C.K., and Chen, A. (2002). A location-routing-loading problem for bill delivery services. *Computers and Industrial Engineering*, 43: 5-25.
- Lin, S. (1965). Computer Solutions of the Traveling Salesman problem. *Bell System Technical Journal*, 44: 2245-2269
- Madsen, O.B.G. (1983). Methods for Solving Combined Two Level Location-Routing Problems of Realistic Dimensions. *European Journal of Operational Research*, 12: 295 - 301.
- Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin: Springer.
- Nagy, G., and Salhi, S. (1996). Nested Heuristic Methods for the Location-Routing Problem. *Journal of the Operational Research Society*, 47: 1166-1174.
- Or, I. and Pierskalla, W.P. (1979). A Transportation Location-Allocation Model for Regional Blood Banking. *AIIE Transactions*, 11: 86 - 95.
- Perl, J. and Daskin, M.S. (1985). A Warehouse Location-Routing Problem. *Transportation Research*, 19B: 381 - 396.
- Renaud, J., Laporte, G. and Boctor, F.F. (1996). A Tabu Search Heuristic for the Multi-Depot Vehicle Routing Problem. *Computers and Operations Research*, 23(3): 229 - 235.
- Srivastava, R., (1993). Alternate solution procedures for the location-routing problem. *Omega International Journal of Management Science* 21(4): 497-506
- Tavares, J., Pereira, F.B., Machado, P., Costa, E. (2003) On the Influence of GVR in Vehicle Routing. *Proceedings of the 2003 ACM Symposium On Applied Computing (SAC 2003) - Evolutionary Computation And Optimization Track*, pp.753-758, Melbourne, FL, 9-13 March, 2003.
- Tuzun, D., and Burke, L. (1999). A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, 116 (1), 87-99.
- Wong, D.W.S. and Meyer, J.W. (1993). A Spatial Decision Support System Approach to Evaluate the Efficiency of a Meals-on-Wheels Program. *Professional Geographer*, 45(3): 332 - 341.
- Wu, T., Low, C., Bai, J. (2002). Heuristic solutions to multi-depot location-routing problems. *Computers and Operations Research*, 29(10): 1393-1415.

Appendix A

Model Parameters:

$J = \{j \mid j = 1, \dots, M\}$ is the set of potential kitchen locations

$I = \{i \mid i = M+1, \dots, N+M\}$ is the set of customers

$K = \{k \mid k = 1, \dots, P\}$ is the set of vehicle routes

c_{ij} = distance from point i to point j ; $i, j \in I \cup J$

t_{ij} = travel time from point i to point j ; $i, j \in I \cup J$

τ_i = maximum time to unload meals for customer $i \in I$

d_i = number of meals required by customer $i \in I$

f_j = fixed cost of establishing a kitchen at site $j \in J$

v_j = variable cost per meal produced at kitchen $j \in J$

g_{ij} = cost per mile of travel from points i to j ; $i, j \in I \cup J$

T_k = maximum allowable duration of route $k \in K$

W_j = number of vehicles available at kitchen $j \in J$

M_j = number of meals that can be served by kitchen $k \in K$

$D = \sum_{i \in I} d_i$ = total demand throughout service area

Decision Variables:

$x_{ijk} = \begin{cases} 1, & \text{if point } i \text{ immediately precedes point } j \text{ on route } k, i, j \in I \cup J, k \in K ; \\ 0, & \text{otherwise.} \end{cases}$

$y_j = \begin{cases} 1, & \text{if a kitchen is located at site } j, j \in J ; \\ 0, & \text{otherwise.} \end{cases}$

$z_{ij} = \begin{cases} 1, & \text{if customer } i \in I \text{ is served from kitchen } j \in J ; \\ 0, & \text{otherwise.} \end{cases}$

U_{ik} are auxiliary variables used in subtour elimination constraints.

Model HDM-LRP:

$$\text{optimize} \left\{ \sum_{j \in J} f_j y_j + \sum_{i \in I} \sum_{j \in J} v_j d_i z_{ij} + \sum_{k \in K} \sum_{i \in I \cup J} \sum_{j \in I \cup J} g_{ij} c_{ij} x_{ijk}, \sum_{i \in I} \sum_{j \in J} d_i z_{ij} \right\} \quad (1)$$

s.t.

$$\sum_{k \in K} \sum_{j \in I \cup J} x_{ijk} = 1 \quad \forall i \in I \quad (2)$$

$$\sum_{i \in I} \sum_{j \in J} d_i z_{ij} \leq M_j y_j \quad \forall j \in J \quad (3)$$

$$\sum_{i \in I} \tau_i \sum_{j \in I \cup J} x_{ijk} + \sum_{i \in I \cup J} \sum_{j \in I \cup J} t_{ij} x_{ijk} \leq T_k \quad \forall k \in K \quad (4)$$

$$U_{ik} - U_{jk} + N x_{ijk} \leq N - 1 \quad \forall i, j \in I, k \in K \quad (5)$$

$$\sum_{i \in I \cup J} x_{ijk} - \sum_{i \in I \cup J} x_{jik} = 0 \quad \forall j \in I, k \in K \quad (6)$$

$$\sum_{j \in J} \sum_{i \in I \cup J} x_{ijk} \leq 1 \quad \forall k \in K \quad (7)$$

$$\sum_{h \in I \cup J} x_{ihk} + \sum_{h \in I \cup J} x_{jhk} - z_{ij} \leq 1 \quad \forall i \in I, j \in J, k \in K \quad (8)$$

$$\sum_{i \in I} \sum_{k \in K} x_{ijk} \leq W_j \quad \forall j \in J \quad (9)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i \in I \cup J, j \in I \cup J, k \in K \quad (10)$$

$$y_j \in \{0, 1\} \quad \forall j \in J \quad (11)$$

$$z_{ij} \in \{0, 1\} \quad \forall i \in I, j \in J \quad (12)$$

$$U_{ik} \geq 0 \text{ and integer} \quad \forall i \in I, k \in K \quad (13)$$

Objective (1) is composed of two competing measures. The first, a measure of efficiency, is a total cost term equal to the sum of fixed kitchen location costs, variable kitchen throughput costs and variable delivery costs. This objective is minimized. (Note that by varying the variable costs v_j and g_{ij} , more or less weight is placed on these volunteer-donated activities.) The second objective,

an effectiveness measure, represents the total demand served, which is to be maximized.

Constraints (2) ensure that each customer is served on a single route by a single kitchen. Constraints (3) ensure that a kitchen, if located, cannot serve more than its maximum capacity of meals. Constraints (4) ensure that the duration of each route does not exceed the maximum allowable duration. Constraints (5) require that every delivery route be connected to a kitchen. They also act as subtour elimination constraints. Constraints (6) ensure that a vehicle leaves every node that it enters. Constraints (7) ensure that a route is assigned to at most one kitchen. Constraints (8) are linking constraints that ensure that a customer may be allocated to a kitchen only if there is a route assigned to that kitchen that serves that customer. Constraints (9) ensure that the number of routes assigned to each kitchen does not exceed the maximum allowable number of routes. Constraints (11), (12), and (13) ensure that the routing variables, location variables, and allocation variables can take on values 0 or 1. Constraints (14) ensure that the auxiliary variables are nonnegative integers.