5-2016

A Tabu Search, Augment-Merge Heuristic to Solve the Stochastic Location Arc Routing Problem

Tiffany L. Yang
University of Arkansas, Fayetteville

A Tabu Search, Augment-Merge Heuristic to Solve
the Stochastic Location Arc Routing Problem

A thesis submitted in partial
fulfillment of the requirements for the degree of
Master of Science in Industrial Engineering

by

Tiffany L. Yang
University of California, Los Angeles
Bachelor of Science in Chemical Engineering, 2009

May 2016
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

Dr. Kelly Sullivan
Thesis Director

Dr. Ashlea Milburn
Committee Member

Dr. Sarah Nurre
Committee Member

ABSTRACT

The location arc routing problem (LARP) is a network optimization problem combining strategic facility location decisions and tactical or operational vehicle routing decisions for customer demand located on arcs of a network. The LARP seeks to locate facilities, or depots, and create vehicle delivery routes to minimize costs. The total cost is comprised of three components: fixed facility locations costs, fixed route creation (or vehicle acquisition) costs, and variable arc traversal costs. The applications of the LARP are varied and often include public services such as mail delivery, garbage collection, and street sweeping. In all of these applications, the magnitude of customer demand may be unknown at the outset of the problem and realized uncertainty can greatly affect the final solution. To the author's knowledge, there is currently no discussion of formulating or solving a LARP with uncertainty.

This paper presents an iterative tabu search, augment-merge heuristic to solve the LARP with stochastic customer demand. Each realization of customer demand for a particular network, represented by an individual scenario, was generated using the deterministic *mval* instances (with 24-50 nodes and 44-138 arcs) created by Hashemi Doulabi and Seifi (2013) and a truncated normal probability distribution. The tabu search phase handles the depot location decisions and chooses a set of depots to be used across all scenarios. The augment-merge phase creates a set of vehicle routes for each scenario. One-third of the initial experiments resulted in stochastic solution costs less than their deterministic counterparts indicating the promising value of considering customer demand uncertainty using the proposed stochastic LARP algorithm.

TABLE OF CONTENTS

1. Introduction ... 1
- 1.1. Background and Motivation ... 1
- 1.2. Research Contributions ... 3

2. Literature Review ... 3
- 2.1. Problem Definition ... 4
- 2.2. Facility Location Problems (FLPs) ... 4
- 2.3. Vehicle Routing Problems (VRPs) ... 7
- 2.4. Arc-Routing Problems (ARPs) ... 11
  - 2.4.1. Chinese Postman Problems (CPPs) ... 11
  - 2.4.2. Rural Postman Problems (RPPs) ... 13
  - 2.4.3. Capacitated Arc Routing Problems (CARPs) ... 14
- 2.5. Location Arc-Routing Problems (LARPs) ... 18

3. Tabu Search, Augment-Merge Heuristic ... 22
- 3.1. Initialization ... 24
- 3.2. Tabu Search Phase ... 26
- 3.3. Augment-Merge Phase ... 28

4. Computational Results ... 31
- 4.1. Implementation ... 31
- 4.2. Instance Generation ... 31
- 4.3. Evaluation of Proposed Heuristic on Stochastic vs. Deterministic LARPs ... 32

5. Summary and Conclusions ... 42

6. References ... 44

1. INTRODUCTION

1.1. Background and Motivation

Location-routing problems (LRPs) are a category of network optimization problems combining a facility location problem (FLP) – typically a strategic decision to locate a set of facilities or depots and allocate customers to each opened facility – and a vehicle routing problem (VRP) – usually a tactical or operational decision to determine routes from the depots to multiple customers. Often, the objective is to minimize total facility opening costs, fixed vehicle or route establishment costs, and variable routing costs. While FLPs and VRPs are often viewed as two individual types of problems, considering the interdependency of locating facilities and creating routes can have a significant effect on an organization's logistics costs. Salhi and Rand (1989) first showed the suboptimal results of ignoring routing decisions when locating facilities.

A variation of the LRP is the location arc-routing problem (LARP) that seeks to locate facilities and create vehicle routes along arcs of a network instead of between nodes as is the case with the discrete LRP. Ghiani and Laporte (2001) also describe the LARP as an extension of three classical arc-routing problems (ARPs) – the Chinese postman problem (CPP), the rural postman problem (RPP), and the capacitated arc-routing problem (CARP) – with the added task of identifying facility locations. As the name implies, the CPP's origins are in the realm of mail delivery and aims to find the shortest route for a single postal carrier to traverse all streets (or edges) in a neighborhood. The RPP is similar to the CPP but only requires the carrier (or vehicle) to traverse a subset of edges. While the CPP and the RPP are single-depot, single-vehicle problems that ignore any customer demand magnitudes and vehicle capacities, the CARP is a single-depot, multiple-vehicle problem that incorporates the demand on each edge, accounts for a fleet of homogenous vehicles, and imposes a vehicle capacity constraint. Thus, in the same way the LRP can be seen as a combination of the FLP and the VRP, the LARP can be viewed as an integration of the FLP and the CARP.

In addition to mail delivery, the LARP can be applied to other situations where an organization needs to locate multiple facilities (e.g., depots, dump sites, transshipment points, mail relay boxes, refill vehicles) and create multiple vehicle routes to visit customers in the surrounding areas. Such applications of the LARP include newspaper delivery, garbage collection, road maintenance, power line or gas main inspections, street sweeping, snow plowing, winter road gritting, electric meter reading, and school bus routing. LARP may also be extended to airline crew scheduling where facilities represent hubs and arcs represent flight legs.

One of the first discussions of the LARP, originally named the arc oriented location routing problem, was by Levy and Bodin (1989). Building on an algorithm established by Laporte (1988), they created a location-allocation-routing (LAR) algorithm that first locates depots while minimizing the number of depots opened, then allocates customer arcs to each depot given some workload constraints, and finally creates vehicle routes to minimize the total deadhead time, or time traversing arcs without satisfying demand.

While there is a significant amount of research focusing on different variations of the FLP, the VRP, the ARP, and the LRP, the LARP is a much less studied problem and to the author's knowledge, there is currently no discussion of formulating or solving a LARP with uncertainty. Deterministic LARP models provide a solid foundation for an organization to make strategic decisions of citing depots and tactical or operational decisions of creating vehicle routes, but usually exact parameter values are unknown. Assuming constant values for parameters such as customer demand, travel times or distances, facility capacities, and number of vehicles can simplify the problem, but may result in suboptimal solutions. For example, the amount of mail in a neighborhood may change after the post office has been built and alter the effectiveness of the established delivery routes. Or, the traffic during a snowstorm may increase vehicle travel time and require more snowplows to clear the city's streets on a given day than previously determined. Incorporating the uncertainty of the input parameters to a LARP model may provide a better solution than a deterministic model that assumes the parameter values are constant.

1.2. Research Contributions

The objective of this thesis is to formulate a solution heuristic for a LARP with uncertain customer demand to determine the minimum cost option for locating facilities and creating vehicle routes. Below is a list of main contributions:

- Identify a fast CARP heuristic to use in the vehicle routing phase of the LARP.
- Extend benchmark deterministic LARP instances to include demand uncertainty.
- Create and implement heuristic software to solve problem instances.
- Compare the strength of the stochastic LARP solutions to their deterministic equivalents and quantify any benefits of considering demand uncertainty.
- Quantify total computational time and differentiate time spent identifying facility locations versus time spent creating vehicle routes.

2. LITERATURE REVIEW

The LARP is closely related to many existing problem formulations involving facility location and vehicle routing. The purpose of this section is to define the LARP, summarize the most relevant related problems, compare and contrast them to the LARP, examine some formulations and solution methods currently in the literature, and provide context for the contributions for the LARP with uncertain arc demand.

2.1. Problem Definition

The LARP can be defined on a graph $G = (V, A)$ where V is a set of vertices and A is a set of undirected arcs (or edges), directed arcs, or a combination of the two. Each potential depot location in $j \in J$, a subset of V, has an associated fixed establishment and operating cost f_j and capacity b_j. Each arc a has an associated non-negative traversal cost c_a and non-negative customer demand d_a and the arcs with a positive demand comprise R, a subset of A, that require service. The fleet of vehicles is homogenous with identical capacity Q and each new vehicle (or route) has an associated non-negative fixed cost F. Each vehicle will leave from a depot, traverse a route of arcs to serve customer demand, and return to its original depot.

The LARP seeks to determine a minimum-cost solution for simultaneously locating depots, allocating sets of customer arcs to each depot, and establishing vehicle traversal routes to satisfy customer demand given facility and vehicle capacity constraints. The total cost is the sum of fixed depot costs, fixed vehicle or route establishment costs, and variable route traversal costs.

2.2. Facility Location Problems (FLPs)

The FLP comprises one of the main goals of the LARP – to identify optimal locations for facilities or depots. Although a pure FLP is only concerned with siting facilities, this paper will use the term FLP interchangeably with the location-allocation problem (LAP) to also include the decision of allocating customer demands to each facility. Therefore, a LARP without the vehicle arc routing objective is reduced to a FLP. Owen and Daskin (1998) give a comprehensive review of this strategic problem and provide integer program formulations for three common static, deterministic FLPs: median problems, covering problems, and center (or minimax) problems.

One type of median problem, the P-median problem, measures the efficiency of a facility location using demand magnitude at each customer node and travel distance between customer nodes and facilities. The problem seeks to locate P facilities so as to minimize the total demand-weighted distance between each facility and the customers it serves. Applications include siting warehouses among customer regions or locating switching centers in a communications network. The general formulation of this problem is NP-complete but if potential facility locations are restricted to the nodes of a network and the value of P is fixed, it can be solved in polynomial time for a moderate numbers of customers and potential facilities. If P is variable, the problem is NP-hard. Daskin (2013) proposes several heuristic methods to solve the P-median problem including a myopic algorithm, an exchange heuristic, and a neighborhood search algorithm.

Covering problems are a formulation of the FLP that emphasize a facility's availability, not its average distance, to a customer. A facility is available, or covers customer demand, if its travel distance or time is within a maximum acceptable threshold. For example, locating post offices, unemployment centers, and emergency service facilities such as hospitals, fire stations, and ambulances necessitates this measure of locational efficiency. Two variations of covering problems assume uncapacitated facilities are the set covering problem and the maximal covering problem. The set covering problem ignores demand magnitude at each customer node and seeks to minimize facility location costs to cover all customer demand within a specified distance of a facility. The maximal covering problem incorporates customer demand magnitude at each node and aims to maximize the amount of customer demand covered within the acceptable service distance. The maximal covering problem can be transformed into a special case of the P-median problem by converting each of the facility-to-customer distances in the P-median network from continuous, non-negative values to binary parameters equal to 0 if the distance is within the maximum acceptable threshold and 1 otherwise. The modified P-median objective is to locate facilities to minimize the amount of unserved demand. Because these variants of the covering problem are NP-complete for general networks, they are typically solved by assuming potential facility locations coincide with the customer nodes in a network.

Center problems are a third form of the FLP. The P-center problem (or minimax problem) requires all customer demand to be satisfied and seeks to locate P facilities to minimize the maximum distance between any demand and its nearest facility. If the maximum distance of a P-center solution is less than a specified maximum acceptable threshold, then there exists a maximum covering solution with P facilities that serves all demand. Absolute center problems allow facilities to be located anywhere on the network while vertex center problems restrict facilities to the network nodes. For either variation, if the value of P is fixed, then the problem can be solved in polynomial time; but if P is variable, then it is NP-complete.

The three aforementioned formulations focus on distance or time between facilities and customers but the typical LARP objective is to minimize monetary cost. A fourth type of FLP, the fixed charge facility location problem, aligns the FLP objective accordingly. The fixed charge facility location problem applies a fixed cost to each facility that is open, removes the constraint that P facilities must be located, and aims to minimize the sum of the fixed facility costs and the variable travel costs.

In deterministic models, all parameters are assumed to be known. However, in real-life applications, every input to the model (e.g., customer demand, travel times or distances, facility capacities, costs) has some uncertainty. Owen and Daskin (1998) and Snyder (2006) provide thorough investigations into FLP decision-making under uncertainty. There are two established methods to incorporate uncertainty – stochastic optimization and robust optimization (or scenario planning).

If the probability distributions of the uncertain elements are known, then stochastic optimization can be used to optimize the expected value of the objective function. Often, the objective function for stochastic programs is to minimize the expected cost.

If there is not enough information to construct probability distributions, then a robust approach can be utilized by specifying a number of possible scenarios (or future states) and seeking to minimize the maximum regret. Regret is the difference between the optimal solution (i.e., the optimal solution for the true scenario that occurs when the uncertainty is realized) and the compromised solution. Other possible objectives of a robust FLP are to optimize expected performance across all scenarios or minimize the worst-case performance of any one scenario.

2.3. Vehicle Routing Problems (VRPs)

In addition to locating facilities on a network and allocating customers, the second main goal of the LARP is to establish vehicle routes from each facility to its corresponding customers to satisfy service demand. In the case of the LARP, customer demands exist on the network arcs. A VRP encompasses customer allocation and vehicle routing where customer demands exist on either nodes or arcs. Because the literature often uses VRP to refer specifically to the node routing problem (NRP), this paper follows the same naming convention. Thus, by this definition, removing the facility location aspect of a LARP and converting customer demand from arcs to nodes results in a VRP.

The classic VRP can be defined on a graph $G = (V, A)$ where V is vertex set and A is an arc set. The problem seeks to determine minimum cost routes for multiple vehicles from a single uncapacitated depot to satisfy customer demand at nodes on the network. The fleet is comprised of homogenous vehicles with capacity Q and customer j has an associated non-negative demand d_j that cannot be split between vehicles. Each arc (i, j) has a non-negative traversal cost c_{ij} (in e.g., distance, time, or monetary units) for a vehicle to travel from node i to node j. The objective is to minimize the sum of the route traversal costs while ensuring each vehicle starts and ends at the depot, each customer is visited by exactly one vehicle, and the sum of the customer demand of any particular route does not exceed the vehicle's load capacity.

Laporte (1992) and El-Sherbeny (2010) review exact solution methods for the VRP and Eksioglu et al. (2009) create a taxonomy for the vast VRP realm and identify more than half a dozen papers discussing exact methods. However, because the VRP is NP-hard, construction and improvement heuristics or metaheuristics are often employed.

Construction heuristics include route-first cluster second (RFCS) heuristics, cluster-first route-second (CFRS) heuristics, and the Clarke-Wright savings heuristic. The first phase of a RFCS heuristic creates a single route from the depot through all customer nodes ignoring vehicle capacity – i.e., it first solves the traveling salesman problem (TSP), a single depot, single uncapacitated vehicle variant of the VRP. In the second phase, it separates customers into clusters (or routes) to ensure the total demand of any given route does not exceed the designated vehicle capacity. TSP partitioning is an example of a RFCS algorithm. Prins et al. (2014) survey a more general form of RFCS heuristics called order first-split second (OFSC) heuristics. In the first phase, instead of creating a tour to visit all customers, OFSC heuristics allow the creation of a priority list of customers. Prins et al. (2014) also note some advantages of the RFCS (or OFSC) approach: a smaller solution space (because the set of lists is smaller than the set of routes), flexibility to include additional constraints, and efficiency.

Instead of routing and then allocating like RFCS heuristics do, CFRS is a two-phase construction heuristic that first allocates neighboring customers to ensure demand does not exceed vehicle capacity and then determines routes through each customer cluster. The sweep heuristic algorithm popularized by Gillett and Miller (1974) first orders the customers by imposing a ray from the depot and sweeping it in a circle through each customers. Then, it clusters adjacent customers to satisfy the vehicle capacity constraint and finally it creates the routes by solving a TSP for each customer cluster. Fisher and Jaikumar (1981) and Bramel and Simchi-Levi (1995) describe two other RFCS heuristics.

A third commonly used heuristic is the one introduced by Clarke and Wright (1964) and recently revisited by Segerstedt (2014). The Clarke-Wright savings heuristic can be divided into two phases: construction and improvement. The construction phase initializes the solution by creating a single route for each customer. To improve the solution, the algorithm seeks to combine separate tours that are feasible with respect to vehicle demand and results in positive cost savings. The cost savings for combining two tours is defined by summing the costs of the two arcs being removed (the last arc on the first tour and the first arc on the second tour) and subtracting the cost of the arc being added (the arc connecting the first tour's endpoint and the second tour's starting point). The cost savings is calculated for all pairs of nodes and sorted in non-increasing order. Starting at the top of the savings list (i.e., the pair of tours whose combination results in the greatest cost savings), the algorithm checks if the total customer demand of the two routes does not exceed the vehicle capacity. If so, then the combination is implemented. This procedure is repeated for every pair in the list with a positive cost savings.

Often, after an initial set of tours has been created for a VRP, improvement heuristics are applied to find a lower cost solution. The improvement heuristics can be move operators that are applied to all, or any subset, of routes from the initial solution. Some examples of move operators include λ-exchange (also known as λ-opt), remove and reinsert, and swap. A λ-exchange algorithm calculates the cost of removing λ arcs from a route and replacing them with a different set of λ arcs. If this cost is negative (i.e., if the move results in cost savings) and the resulting tour does not violate the vehicle capacity constraint nor does it result in sub-tours (tours that do not include the depot) or disconnected nodes, then the exchange is implemented.

A remove and reinsert algorithm removes a customer from its current tour and reinserts it in a new location. The new location can be a different location within the same tour or a location within another tour. If the cost of removing three arcs and adding three new arcs is negative and the total customer demands of each resulting tours does not violate the vehicle capacity, then the remove and reinsert move is implemented.

A third improvement algorithm swaps a customer from one route with another customer from a different route resulting in four removed arcs and four newly added arcs. If the cost of the swap improves the solution and the two new routes are feasible, then the swap is implemented.

A more efficient and effective alternative to exact and heuristic methods are metaheuristics, another category of approximate algorithms. Metaheuristics are iterative processes used to solve optimization problems and combine one or more heuristics with a guided search procedure to explore the solution space to find near-optimal solutions (Osman and Laporte 1996). Pisinger and Ropke (2007) propose an adaptive large neighborhood search metaheuristic to solve five different VRP variations. Laporte (1992), Malandraki and Daskin (1992), Laporte et al. (2000), Nagy and Salhi (2005), and El-Sherbeny (2010) summarize the details of different heuristic and metaheuristic algorithms for several VRP variations and name tabu search as one of the most powerful metaheuristics.

In addition to the deterministic formulations of the VRP, there exist a variety of stochastic formulations accounting for uncertainty. Gendreau et al. (1996), Berhan et al. (2014), and Sathyanarayan and Joseph (2014) systematize existing stochastic VRP research papers and present different formulations for VRPs with uncertain customer demand, customer presence and demand, and service time.

2.4. Arc-Routing Problems (ARPs)

Instead of creating routes to visit customers at nodes as the VRP does, the ARP seeks to satisfy customer demand existing along arcs of a network. Ghiani and Laporte (2001) describe the vehicle routing portion of the LARP as extensions of the following three classical ARPs: two uncapacitated problems, the Chinese postman problem (CPP) and the rural postman problem (RPP), and the capacitated arc-routing problem (CARP). CPPs seek to find a minimum cost route that traverses all arcs in the network at least once while RPPs traverse only a subset of arcs. CARPs incorporate vehicle capacities and thus can include a multi-vehicle fleet. Deadheading is when a vehicle traverses a required arc without servicing its demand (i.e., the arc has already been serviced by the same vehicle or is being serviced by another vehicle).

2.4.1. Chinese Postman Problems (CPPs)

The CPP was named after Chinese mathematician Kwan Mei-Ko (or Guan Meigu) who in 1962 studied the problem of devising the shortest cycle for a mailman to service his assigned segments before returning to the post office. Given a connected graph, the CPP seeks to find the least cost tour that passes through every edge at least once. Determining a CPP solution uses the concepts of Eulerian graphs and circuits.

An Eulerian circuit is one that starts at a node, visits each arc in the graph exactly once and each node at least once, and returns to the same node. Ford and Fulkerson (1962) outline the following necessary and sufficient conditions for Eulerian graphs. If the graph is undirected, then every vertex must have an even degree (i.e., each vertex must have an even number of incident edges). If the graph is directed, then every vertex must be symmetric (i.e., the number of edges and arcs entering and leaving each vertex must be equal). If the graph is mixed, then every vertex must have an even degree and the graph must be balanced (i.e., for every subset of vertices, the difference between the total number of arcs leaving the subset and the total number of arcs entering the subset must be less than or equal to the total number of arcs incident with the subset).

CPP algorithms consist of two phases: the first phase converts the graph into an Eulerian graph and the second phase determines the best Eulerian route (Eiselt et al. 1995a). If the original CPP graph is Eulerian, then the first phase can be skipped and the optimal solution is the cost of the Eulerian circuit (i.e., the sum of all arcs in the original graph). However, if the graph does not have an Eulerian circuit, then it must be augmented. Undirected graphs are augmented by listing all pairs of vertices that have an odd degree, identifying the least cost paths between each pair, and adding duplicate, artificial arcs for every arc included in the least cost paths. The augmented graph will result in all vertices having an even degree. The optimal cost of the CPP solution is the total cost of all edges in the augmented graph (i.e., the total cost of the original and additional edges). Augmenting undirected and directed graphs can be solved in polynomial time while augmenting mixed graphs is an NP-hard problem. Eiselt et al. (1995a) present methods for completing the first phase of a CPP solution for all three types of graphs. They also discuss two other NP-hard variations of the CPP: windy (traversal costs across arcs in an undirected graph depend on the direction of travel) and hierarchical (arcs are prioritized by required service levels). Thimbleby (2003) provides executable Java code to solve a directed CPP.

After phase one has been completed and the original CPP graph has been converted into an Eulerian graph, the second phase is to determine the actual traversal route. In an Eulerian graph, the route can be started at any node and return to the same node. Thus, as the graph size increases, the number of possible routes increases. One useful fact is the number of times each vertex will appear in a CPP is equal to one-half its degree – i.e., if a vertex has a degree of 6, then the vertex should be traversed 3 times in the CPP solution (Greenaway 2004). The only exception is the depot node that will be visited one additional time. Hertz and Mittaz (2010) present a straightforward route construction algorithm for an undirected CPP. First, identify a cycle C in the Eulerian graph. If C traverses all arcs in the graph, then the solution has been obtained. If not, then choose a vertex $v \in C$ and an arc incident to this node that is not in C and construct a second cycle C' such that they have no common arcs. Merge the two routes C and C' by starting at vertex v, traversing the arcs in C, then traversing the arcs in C', and ending at v. If the merged route traverses all arcs, then the CPP solution is the merged route. Otherwise, continue creating additional cycles until all arcs have been traversed.

The CPP is a single-facility, single-vehicle problem in which the vehicle is uncapacitated and all arcs must be traversed. This formulation can be applied to urban settings and provides a basis for understanding and developing solution methods for other ARPs, but it is not very realistic because organizations do not typically service all customer arcs in a region. The RPP is a variation of the CPP that is more applicable to real scenarios because it only requires traversal of a subset of arcs in the network.

2.4.2. Rural Postman Problems (RPPs)

The RPP consists of a single facility and a single uncapacitated vehicle and seeks to find the least cost tour visiting a subset of required arcs. Undirected, directed, and mixed RPPs are all NP-hard but are more realistic than CPPs.

Hertz and Mittaz (2010) present a route construction algorithm for a special type of undirected RPPs. First, a sub-network is created by removing all non-required arcs in the original graph. If the remaining required arcs in the sub-network are still connected, then this algorithm applies. Convert the sub-network into an Eulerian graph by listing pairs of vertices that have an odd degree, identifying the least cost paths between each pair using any arcs from the original graph, and adding duplicate, artificial arcs for every arc included in the least cost paths. Then, the problem can be solved with any CPP method.

Hertz and Mittaz (2010) also illustrate the following heuristics presented in other papers: a construction algorithm for solving an undirected RPP whose required arcs do not form a connected sub-network (Frederickson 1979), a construction algorithm utilizing add moves, an improvement algorithm to shorten an existing route, and an algorithm to drop arcs (Hertz 2009);

Eiselt et al. (1995b) discuss several useful applications of the RPP: street sweeping, snow plowing, garbage collection, mail delivery, school bus routing, and electric meter reading. They propose formulations for solving three variations: undirected, directed, and mixed (the stacker crane problem). Like solving the CPP, RPP solution algorithms include two phases: one to augment a graph to make it Eulerian and another to determine the least cost cycle. The first phase employs heuristics and the second phase can be solved exactly using TSP techniques.

While the CPP and RPP assume unlimited vehicle capacity, the CARP discussed in the next section accounts for this extra problem constraint present in most real-world problems.

2.4.3. *Capacitated Arc Routing Problems (CARPs)*

Initial discussions of CARP are attributed to Golden and Wong (1981) who expanded uncapacitated ARP formulations such as the CPP and RPP and imposed vehicle capacity constraints. The classic CARP is a single-facility, multiple-vehicle problem that seeks to find the least cost set of routes to traverse a subset of arcs in an undirected network. The fleet is comprised of homogenous, capacitated vehicles and each arc must be serviced by exactly one vehicle (i.e., arc demand cannot be split among multiple vehicles). CARPs are NP-hard and several classic construction heuristics include construct-strike (Christofides 1973), augment-merge (Golden and Wong 1981 and Golden et al. 1983), and path-scanning (Golden et al. 1983).

The CARP is the arc equivalent to the classical node VRP discussed in Section 2.3. The VRP can be transformed to a CARP (Golden and Wong 1981) and a CARP can be transformed into a capacitated VRP (Longo et al. 2006). In the CARP-to-VRP transformation, the problem is solved by introducing two vertices for every required arc in the graph, arbitrarily splitting arc demand between these two vertices, and applying any VRP heuristic. However, most CARP algorithms do not rely on this transformation.

One variation of the CARP is referred to as the capacitated Chinese postman problem (CCPP) where all arcs, instead of a subset of arcs, have positive demand. The construct-strike heuristic originally proposed by Christofides (1973) for the CCPP and revisited by Golden et al. (1983) constructs a list of feasible routes and then strikes (or removes) routes from the list to iteratively decrease overall cost of the solution.

Augment-merge is a construction algorithm comprised of three steps. First, it initializes the solution by assigning every demand arc to its own route. The initial routes are created by computing the shortest path from each arc's endpoints to the depot. Then, the augment phase sorts the list of routes in non-increasing cost order and determines whether a route can be augmented. Starting at the top of the list (i.e., the most expensive route), each route is compared to all of the remaining lower cost routes. If the higher cost route is already servicing the arcs in the lower cost route, then the lower cost route will be removed from the list and its arcs will be augmented into the higher cost route, provided the vehicle capacity constraints are not violated. The third phase, merge, is similar to the Clarke-Wright savings algorithm for the VRP and seeks to combine pairs of routes that result in positive cost savings. Any routes created in the augment phase that have already reached the vehicle capacity limit are ineligible to be merged – only routes whose total customer demand is below the capacity constraint can be modified in the third phase. A list of cost savings is created for every pair of possible route mergers and sorted in non-increasing order. The cost savings by merging two routes is the sum of the individual route costs minus the cost of the merged route. Starting at the top of the list, every feasible merger (i.e., merge every pair of routes that does not violate capacity constraints) is implemented until the cost savings are no longer positive. The third phase in its entirety is repeated –routes eligible for merging are identified, cost savings for each pair of eligible routes are computed, and feasible merges are implemented – until no more merges can be made.

The path-scanning algorithm constructs vehicle routes for a CARP by starting with a path, scanning adjacent arcs to determine the best arc to add to the current path, and repeating the scan-then-add process until vehicle capacity is reached. The path is converted into a route by adding the shortest path return to the depot. The route construction procedure is repeated until all required arcs have been serviced. This algorithm produces five different solutions and chooses the set of routes with the lowest cost. Each set of routes is created by using one of five optimization criteria. Ensuring the vehicle capacity is not violated, an adjacent arc is added to the current path such that 1) the distance per unit remaining demand is minimized; 2) the distance per unit remaining demand is maximized; 3) the return distance to the depot is minimized; 4) the return distance to the depot is maximized; or 5) the return distance to the depot is maximized if the vehicle is less than half-full, otherwise the return distance is minimized. The path-scanning algorithm utilizes five different optimization approaches to identify the lowest cost set of routes.

Wøhlk (2005) summarizes other CARP construction heuristics including parallel-insert, modified construct-strike, randomized path-scanning, and augment-insert. New problem-specific heuristics double outer scan, node duplication, and A-ALG algorithms are presented. Also included in the review are several metaheuristic methods such as simulated annealing, tabu search, genetic algorithms, ant colony system algorithms, and guided local searches.

Wøhlk (2008) reviews CARP literature from the preceding ten years and summarizes exact methods, heuristics, metaheuristics, and lower bounds developed for the classic CARP. The survey also discusses eight different formulations of the CARP: on directed/mixed graphs, with non-traditional objective functions, with time windows, with multiple depots, with mobile depots, with vehicle/site dependencies, with periodic considerations, and with stochastic elements.

Belenguer and Benavent (2003a) introduce an effective cutting plane algorithm to compute a lower bound for the CARP and identify four CARP benchmark instances: *bccm*, *gdb*, *kshs*, and *eglese* (Belenguer and Benavent 2003b). The first three sets consist of synthesized data while the fourth set of instances is derived from real winter gritting data in Lancashire, United Kingdom.

Perrier et al. (2005) review different CARP solution methods in the context of winter road maintenance, namely salt and abrasive spreading operations. They summarize sequential, parallel, and CFRS constructive heuristics used in these applications.

Belengeuer et al. (2006) discuss a formulation of the CARP on a mixed network with undirected edges and directed arcs. They present methods for determining a lower bound to the problem and mixed CARP (MCARP) variations of the path-scanning, augment-merge, and Ulusoy’s heuristic. The first two heuristics are as discussed above. Ulusoy’s heuristic is a RFCS algorithm that first creates a giant route traversing all required arcs ignoring vehicle capacity and then splits the route into several smaller feasible routes (Ulusoy 1985). They also discuss a memetic algorithm for solving the mixed CARP.

Polacek et al. (2008) present a variable neighborhood search (VNS) metaheuristic to solve a variation of the CARP with intermediate facilities (CARPIF). CARPIFs are situations where vehicles start at a depot, perform a service (e.g., waste collection or snow removal) along arcs of customer demand, visit intermediate facilities (e.g., specific dumpsites to transfer material) along the way, and return to the same depot.

Stochastic CARPs with uncertain demand have been studied by Christiansen et al. (2009) who present an exact branch-and-price algorithm and Laporte et al. (2010) who propose an adaptive large neighborhood search metaheuristic. Mei et al. (2010) investigate the effectiveness of applying deterministic CARP solution methods to situations with uncertainty and demonstrate the algorithms do not provide robust solutions to the stochastic formulation.

CARPs assume facility locations are fixed and only make vehicle routing and customer allocation decisions. To reduce overall logistics costs, LARPs incorporate the strategic decision of siting facilities while simultaneously considering vehicle routing.

2.5. Location Arc-Routing Problems (LARPs)

LARPs combine facility location decisions discussed in Section 2.2 and arc routing decisions presented in Section 2.4.3. These problems seek to locate depots and create vehicle routes to service customer demand on arcs in a network. The typical objective of a LARP is to minimize total costs comprised of fixed facility location costs, fixed vehicle or route establishment costs, and variable arc traversal costs. Levy and Bodin (1989) first presented the

LARP as the arc oriented location routing problem and proposed a location-allocation-routing algorithm.

Ghiani and Laporte (1999) formulate a version of the uncapacitated RPP considering location decisions, called the location rural postman problem (LRPP), on an undirected graph. They show if the LRPP has no upper bounds on the number of depots that can be located or if only one depot is located, then the LRPP can be reduced to a classic RPP. Otherwise, they outline an exact method of solving the LRPP with an upper bound on the number of depots.

Ghiani and Laporte (2001) provide a brief survey of LARPs in the context of mail delivery, garbage collection, and road maintenance and discuss some related VRP and CARPIF heuristics. They posit the best heuristic algorithms construct vehicle routes first and then locate facilities, instead of locating facilities and then creating sets of routes. Also, they recommend including customer allocation decisions in the routing phase instead of in the facility location phase.

Related to LARPs, the CARP with refill points (CARPRF) seeks to create routes for service vehicles to paint and repaint road markings along streets and to visit refill vehicles that carry replenishment paint. The refill vehicles are located before service vehicles embark on their routes and thus can be considered depots. Amaya et al. (2007) propose an integer linear programming model for the CARPRF.

Hashemi Doulabi and Seifi (2013) present mixed integer programming models and lower and upper bounds for single- and multiple-depot LARPs. They also propose an insertion heuristic to solve the CARP portion of the problem and combine it with a location-allocation simulated annealing heuristic.

Lopes et al. (2014) summarize several algorithms to solve the LARP including construction heuristics (extended augment-merge and extended merge), improvement heuristics (reverse and relocate), and metaheuristics (tabu search-variable neighborhood search, greedy randomized adaptive search procedure, and tabu search-greedy randomized adaptive search procedure).

The extended augment-merge (EAM) heuristic, a LARP construction algorithm, is based on the CARP augment-merge heuristic discussed in Section 2.4.3 and consists of three phases. The first phase, initialization, assigns every required arc (i.e., arc with customer demand) to its closest depot until depot capacity is reached. Any unused depots, or depots without assigned arcs, are closed. The augment phase sorts the routes in non-increasing cost order. Starting at the top of the list (i.e., the highest cost route), it checks if each route traverses any required arcs included in any lower cost routes. If so, and if the sum of customer demands of the two routes will not exceed the vehicle capacity, it augments (or absorbs) the required arcs of the lower cost route into the higher cost route. The third phase, merge, calculates the cost savings for merging each pair of routes (similar to the Clarke-Wright savings algorithm for the VRP) and reassigning the merged route to each depot. Sorting the cost savings in non-increasing order, it starts at the top of the list and merges the pair of routes if the combination does not exceed vehicle or facility capacities. After this step is repeated for every possible merger with positive cost savings, the EAM algorithm is complete.

Another LARP construction algorithm Lopes et al. (2014) propose is the extended merge (EM) heuristic similar to the algorithm used by Belenguer et al. (2006a) to solve a CARP. If not all of the arcs in the network require service, the augment phase of the EAM may introduce deadheading costs (or traversal costs along arcs where demand is not serviced) that cannot be overcome by the cost savings gained in the merge phase. Thus, the EM heuristic is like the EAM heuristic without the augment step.

Two improvement heuristics are the reverse heuristic and relocate heuristic. The reverse improvement heuristic is performed within each initially constructed feasible route and is analogous to the 2-exchange algorithm used to improve VRP solutions. It chooses a sub-sequence of arcs within a route, reverses its direction, and reinserts it into the original route. If the reverse operation results in a cost savings, then the route is updated with the change. This is continued within the route until no more cost savings can be found and is repeated for every route in the solution.

The relocate improvement heuristic can be performed within a route or between two routes. It chooses a sub-sequence of arcs within a route and relocates it to another position within the same route or, if it does not violate vehicle or facility capacities, inserts it into another route. The feasible relocation with the largest cost savings is implemented and the procedure is repeated for different sub-sequences of arcs and different routes.

In addition to the more simple construction heuristics, Lopes et al. (2014) present a general tabu search-variable neighborhood search (TS-VNS) metaheuristic where tabu search (TS) is used to make facility location decisions and VNS is used to make routing decisions.

Starting with a subset of depots equal to all possible depot locations, the algorithm is initialized using the EAM construction heuristic to obtain a feasible solution. Following the VNS methods proposed by Polacek et al. (2008) to solve CARPIF, the set of routes is updated by shaking using a cross-exchange move – randomly swapping sub-sequences of arcs between routes. A local search in the neighborhoods of these two changed routes is performed by using the reverse and relocate improvement heuristics until no more improvements can be made. Then, current best solution is used as input for the tabu search (TS) phase.

The first step in the TS phase is to represent the cumulative customer demand for each route determined in the VNS phase as a single client. The smallest insertion cost of each depot into each route is computed and used to represent the distance from a depot to a route. The FLP is solved to determine the best depot locations for the current routes and the subset of depots is updated to be passed back into the VNS phase.

The algorithm continues iterating between the VNS and TS improvement steps to find a better solution. If no improvement is found in the last five iterations, then the subset of depots is altered by opening one or two depots and closing another depot, ensuring depot capacities are not exceeded. The entire procedure is repeated until a specified number of iterations without improvement (suggested to be 10 times the number of arcs in the network) are executed.

Liu et al. (2008) provide another survey of LARPs and propose stochastic LARPs as an area of future research. Similarly, reviews of the related LRP by Nagy and Salhi (2007), Lopes et al. (2013), and Prodhon et al. (2014) note the relative lack of research of LARPs compared to their node LRP counterparts and suggest stochastic formulations of these problems due to their real-world applicability.

3. TABU SEARCH, AUGMENT-MERGE HEURISTIC

Given the existing landscape of problems involving depot location and routing decisions, two algorithms were chosen to solve the LARP with stochastic demand, uncapacitated facilities, and capacitated vehicles. The proposed solution method is an iterative framework with two phases to address the two components of the problem. The first phase uses a tabu search metaheuristic to identify the depot locations to be used across all the scenarios and was chosen for its existing applications to the uncapacitated FLP as studied by Michel and Van Hentenryck (2004) and Sun (2006). The second phase uses an augment-merge constructive heuristic to create a set of vehicle routes for each scenario and was chosen for its computation speed and ease of implementation for solving CARPs.

After a problem is initialized, the tabu search phase uses the sets of vehicle routes for each scenario and modifies the set of open depots to minimize total solution cost. This phase keeps the group of demand arcs and their respective traversal directions for each route intact, only cycling – never rearranging – the order of arcs visited. For example, if a route created during the initialization phase is comprised of demand arcs [5, 10, 7, 8 13, 0] with traversal directions [1, 1, 2, 1, 2, 2], then a possible rearrangement of the route could be [7, 8, 13, 0, 5, 10] with directions [2, 1, 2, 2, 1, 1].

Upon completion of one cycle of the tabu search phase, one set of depots is passed to the augment-merge phase where new vehicle routes are constructed. This phase does not make changes to the depot locations and creates vehicle routes with no regard to the set of vehicle routes used prior to this cycle.

The algorithm cycles back and forth between these two phases until the total elapsed solution time, *elapsedTime*, reaches a specified maximum time, or *maxTime*. Figure 1 shows a framework of the heuristic. A legend for many of the pertinent parameters is included in Figure 4 and pseudocode is defined in Figure 5.

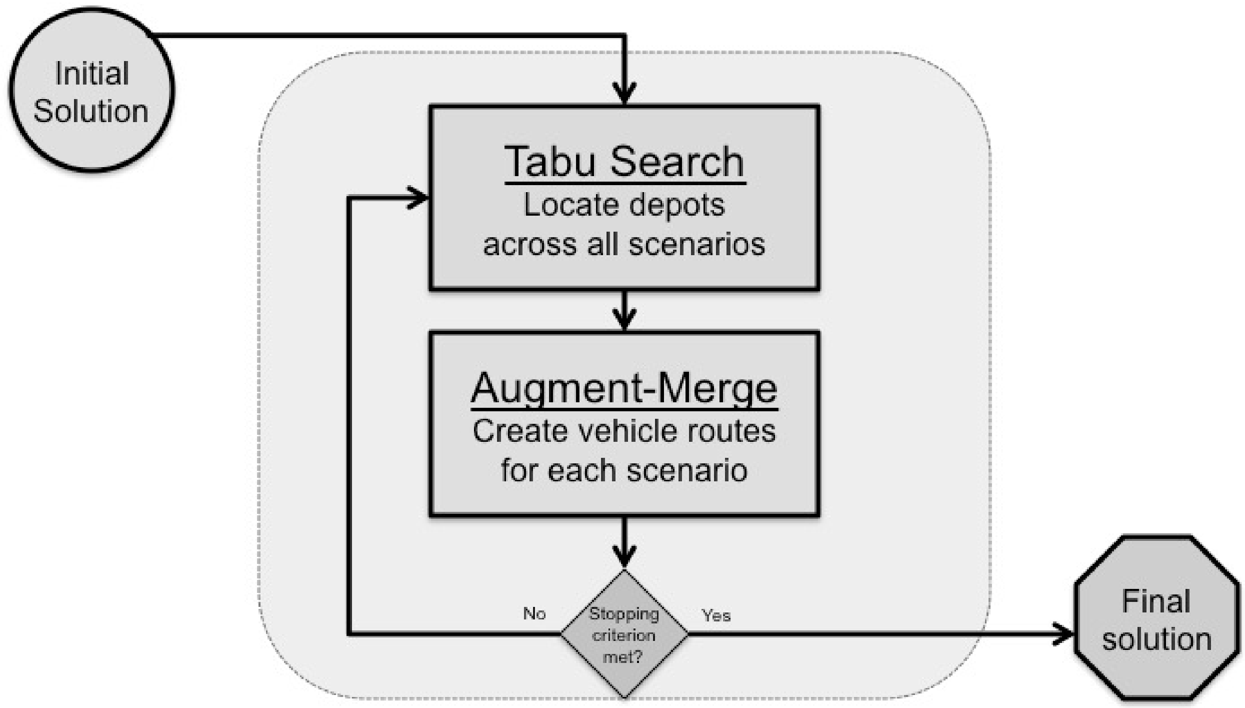

Figure 1: Tabu Search, Augment-Merge Heuristic Framework

3.1. Initialization

The problem is initialized by opening all depots and using the augment-merge heuristic to create sets of feasible vehicle routes for each scenario. Figure 2 shows an example of how the deterministic arc demands for a particular network are modified to create a stochastic instance with three scenarios. An individual route is created for each demand arc, starting from the demand arc's origin node, traversing the demand arc, and then traversing the shortest path to return to its origin node.

After creating these initial trivial routes, to augment the routes, the set of routes is sorted in non-increasing cost order. Starting at the top of the list, if a demand arc on a cheaper route is already traversed in the return trip of a more expensive route and the combined customer demand does not exceed the vehicle capacity, then this demand arc is augmented (or absorbed) into the more expensive route and the cheaper route is deleted from the list. This process is repeated for each route until the second-to-last route has been checked.

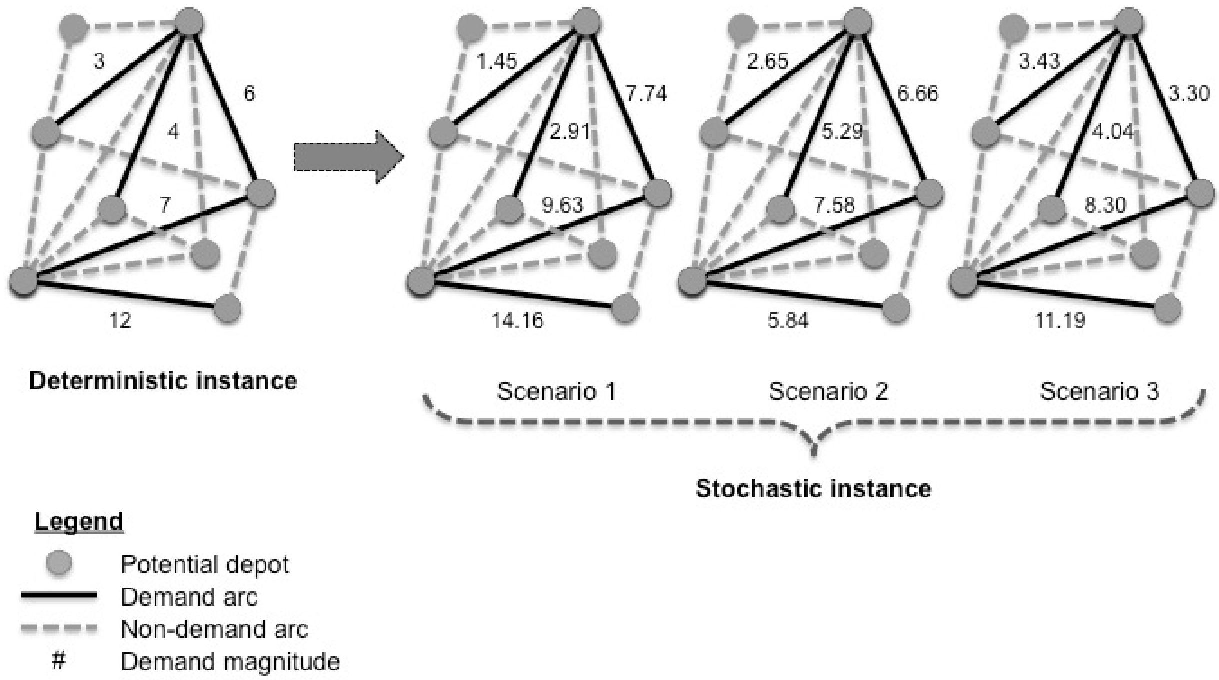

Figure 2: Example of Converting a Deterministic Instance to a Stochastic Instance Using Scenarios to Represent Demand Uncertainty

Pairs of adjacent routes (i.e., routes that share at least one node) are merged by building a cost savings matrix for each feasible route merger (i.e., if the combined demand of the two routes does not violate vehicle capacity). Two routes are merged by inserting the series of demand arcs of one route into the other route. If all arcs of either route are undirected, then the merge is performed by maintaining the original route direction (forward or reverse) and repeated in the opposite direction (reverse or forward). Figure 3 shows an example of a route merger for two adjacent routes. The cost savings of any route merger is the sum of the costs of the individual routes minus the cost of the merged route. The route merger with the largest cost savings is implemented, a new cost savings matrix is created for all new feasible route mergers, and the process is repeated until there are no more feasible route mergers with positive cost savings. If there are any ties for the largest cost savings, then the route merger that appears first in the matrix, or has the lower numbered routes, is chosen.

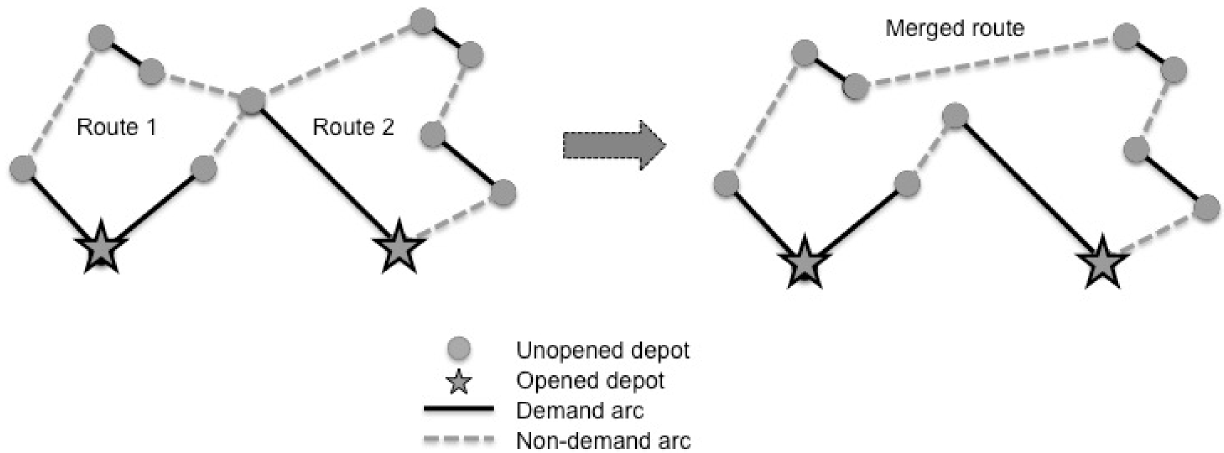

Figure 3: Example of Route Merger for Two Adjacent Routes

The total solution cost is the sum of the total routing costs (fixed and variable) for each scenario weighted by the scenario's probability of occurrence plus the total fixed depot location costs. After augmenting and merging routes, any depots unused across all scenarios are closed. The initialization information includes a vector with the status of each depot – open or closed (*depotStatuses*), an array of the sets of vehicle routes for each scenario (*vehRouteScenarios*), and an array of the sets of total demand to be serviced by each depot in each scenario (*depotDemandScenarios*). Each route in *vehRouteScenarios* includes the demand arcs to be serviced, the directions to traverse the demand arcs, the total cost of the route, the total demand to be serviced, the nodes to be visited, and the depot node. The global best solution, *bestSolution*, is initialized with the aforementioned information and is then passed to the iterative loop where the solution optimization occurs.

3.2. Tabu Search Phase

The tabu search phase takes the set of vehicle routes for each scenario and identifies one set of depots to be used across all scenarios. The neighborhood for a particular set of depots is created by flipping the status of any one depot at a time, i.e., closing an open depot or opening a closed depot. Starting with the set of depots and routes created from the initialization for the first tabu iteration (or from the augment-merge phase in the subsequent tabu iterations), the cost savings of performing all possible depot flips is calculated.

If a depot is closed, all routes previously assigned to it are reassigned to another open depot. To determine where the open depot should be located in a route, the open depot is inserted between each pair of demand arcs in a route and the resulting routing costs are computed. The depot insertion location resulting in the lowest cost is chosen. This process is repeated for all open depots and the route is assigned to the depot with the cheapest insertion cost.

If a depot is opened, the cost of moving each existing route to the newly opened depot is calculated. Again, the depot insertion cost is calculated for all possible locations in a route and the route is reassigned to the newly opened depot only if the routing cost decreases.

The cost savings of the current neighbor, *currNeighborSavings*, is the total routing and depot location costs of the previous solution minus the total costs of the new neighbor solution. The tabu list stores the flips that were performed in the last 10 iterations. If a flip results in the largest cost savings in the cycle (*bestNeighborSavings*) and the flip is not on the tabu list, then the flip is implemented and the best neighbor information, *bestNeighborSolution* and *bestNeighborCost*, are updated. The aspiration criterion, or criterion that overrides the non-tabu flip requirement, is if the new solution cost is less than the overall best-known solution. After a flip has been chosen, the neighborhood search procedure is repeated until the number of iterations within one cycle of the tabu search phase, *tabuIter*, reaches its maximum limit, *maxTabuIter*, before continuing to the augment-merge phase. To introduce diversification and encourage departure from a local optimum, a random eligible depot flip is chosen every five tabu iterations, similar to the method used by Lopes et al. (2014) in the tabu search phase of their tabu search-variable neighborhood search heuristic to solve the deterministic LARP.

After the end of one tabu search cycle, a set of opened depots and vehicle routes across all scenarios with the lowest total solution cost is identified and passed to the augment-merge phase. If this cost improves the overall best-known solution, then the best-known solution is updated with this new information.

3.3. Augment-Merge Phase

The augment-merge phase is very similar to that performed in the initialization phase with a few exceptions. The vehicle routes used in the tabu search phase are deleted and new trivial routes are created. However, instead of all depots eligible for use, only the open depots selected during the most recent tabu search cycle can be assigned routes. Thus, because each demand arc's origin node may not contain an open depot, each demand arc is assigned to the depot closest to its origin node. Again, for undirected arcs, the origin node is the lower-numbered node. After the trivial routes are created, the routes are augmented and merged as before, but unused depots across all scenarios are not closed. After the initialization, depot flips are only performed in the tabu search phase, not in the augment-merge phase. This process is repeated for *numScenarios*, the total number of scenarios generated for the instance. Once a set of vehicle routes for each scenario has been created, the heuristic revisits the tabu search phase if the time limit has not yet been reached, or it exits the iterative loop and ends the algorithm. If the set of depots and vehicle routes from the augment-merge phase, *currCost*, improves the overall best-known solution, then the best-known solution is updated with this new information.

- *bestNeighborCost*: Cost of best solution in current neighborhood
- *bestNeighborSavings*: Best savings in current neighborhood
- *bestNeighborSolution*: Best *depotStatuses*, *depotDemandScenarios*, and *vehRouteScenarios* in current neighborhood
- *currNeighborSavings*: Cost savings for current neighbor's depot flip
- *depotDemandScenarios*: Array of demand serviced by each depot across each scenario
- *depotStatuses*: Vector of statuses (open or closed) for each depot
- *elapsedTime*: Total elapsed time upon starting implementation of heuristic
- *maxTabuIter*: Maximum tabu iteration before exiting tabu search phase
- *maxTime*: Maximum elapsed time before exiting tabu search phase
- *numScenarios*: Total number of scenarios generated for the instance
- *tabuIter*: Counter for current iteration of tabu search phase

Figure 4: Legend of Tabu Search, Augment-Merge Heuristic Parameter Names

```
INITIALIZATION
  Generate scenarios from deterministic instance
  for numScenarios
    Create sets of trivial routes for each scenario
    Augment routes
    Merge routes
    Close any unused depots
  end
  Set bestSolution = currSolution
  Set bestSolutionCost = currSolutionCost

SOLUTION OPTIMIZATION
Tabu search locate depots
  while elapsedTime < maxTime
    if tabuIter < maxTabuIter
      bestNeighborSavings = -∞
      if mod(tabuIter, 5) equals 0
        Randomly choose an eligible depot flip
        if currNeighborSavings > bestNeighborSavings
          Update bestNeighborSavings, bestNeighborSolution, and tabuList
        end
      else
        for all depots
          Flip depot and calculate currNeighborSavings
          if (depot is not on tabuList and currNeighborSavings >
            bestNeighborSavings)
          or (depot is on tabuList and currNeighborSolutionCost <
            bestSolutionCost)
            Update bestNeighborSavings, bestNeighborSolution, and tabuList
          end
        end
      end
    end
  end

  if bestNeighborCost < bestSolutionCost
    Update bestSolution and bestSolutionCost
  end

Create vehicle routes
  for numScenarios
    Create sets of trivial routes for each scenario
    Augment routes
```

Figure 5: Tabu Search, Augment-Merge Heuristic Pseudocode

4. COMPUTATIONAL RESULTS

4.1. Implementation

The heuristic was implemented in MATLAB_R2015b using a Intel Core i7-5600U CPU with a 2.60-GHz processor, Windows 7 Enterprise, and 16.0 GB of RAM. Two built-in MATLAB functions in the Bioinformatics Toolbox were used: *graphallshortestpaths* to determine the shortest path distances between two nodes and *graphshortestpath* to determine the sequence of nodes along the shortest path between two nodes.

4.2. Instance Generation

Stochastic instances of the LARP were generated using existing deterministic instances. The set of *mval* instances were created by Hashemi Doulabi and Seifi (2013) and are based on CARP instances from Belenguer et al. (2006a). A subset of 10 of the 34 total instances were used: each instance with the subscript A (e.g., *mval1A*, *mval2A*,…, *mval10A*). The instances are comprised of mixed (i.e., directed and undirected) graphs with 24-50 nodes (i.e., potential depot locations), and 44-138 demand arcs. For each of the 10 instances, the pair of origin and destination nodes, the traversing (or traversal) cost, and the demand for each arc were used while the service cost was ignored. An indicator for directedness was also assigned to each arc – undirected arcs (or edges) were assigned a value of 0 and directed arcs were assigned 1.

To create scenarios for a particular instance, arc demands were generated from a truncated normal probability distribution similar to the method used by Laporte et al. (1989) for the stochastic LRP. For each arc, the distribution mean was equal to the deterministic demand value and the standard deviation was a coefficient times the mean. Any negative values generated were replaced by zero. Each scenario’s probability of occurrence was assumed to be equal.

4.3. Evaluation of Proposed Heuristic on Stochastic vs. Deterministic LARPs

To evaluate the effectiveness of the proposed tabu search, augment-merge heuristic, 9 experiments were conducted for each of the 10 instances, resulting in a total of 90 experiments. The parameters to be tested were divided into two groups: instance parameters and model parameters, summarized in Table 1.

The values for the three instance parameters were drawn from existing literature and were unchanged across the 90 experiments. The fixed vehicle cost (referred to as dumping cost by Belenguer et al. 2006a) was 0. As defined by Hashemi Doulabi and Seifi (2013), the fixed depot location cost for each instance was calculated using a depot fixed cost factor multiplied by the average cost of all arcs in the network. A depot fixed cost factor of 1 was used. To generate the scenarios, the variance of the truncated normal probability distribution of each arc demand was defined using a standard deviation factor of 1/3 multiplied by the mean.

There were three model parameters, two of which were varied for experimentation. The number of scenarios generated (*numScenarios*) had values of 10, 30, or 50. The maximum time allotted to solve each LARP (*maxTime*) was 1 hr. The maximum number of tabu search iterations (*maxTabuIter*) was set to 32, 125, or 500. Table 2 summarizes the parameters that were changed for each of the 90 experiments.

One replication of each of the 90 stochastic LARP experiments was conducted using the full tabu search, augment-merge heuristic. Similarly, using the deterministic arc demands for each instance, the full heuristic was used to solve 90 comparison experiments without stochastic scenarios. The depot locations for both stochastic and deterministic versions of each experiment were stored and used for further analysis. Figure 6 shows the total solution cost versus time for the *mval1A* experiment with *numScenarios* = 10 and *maxTabuIter* = 30.

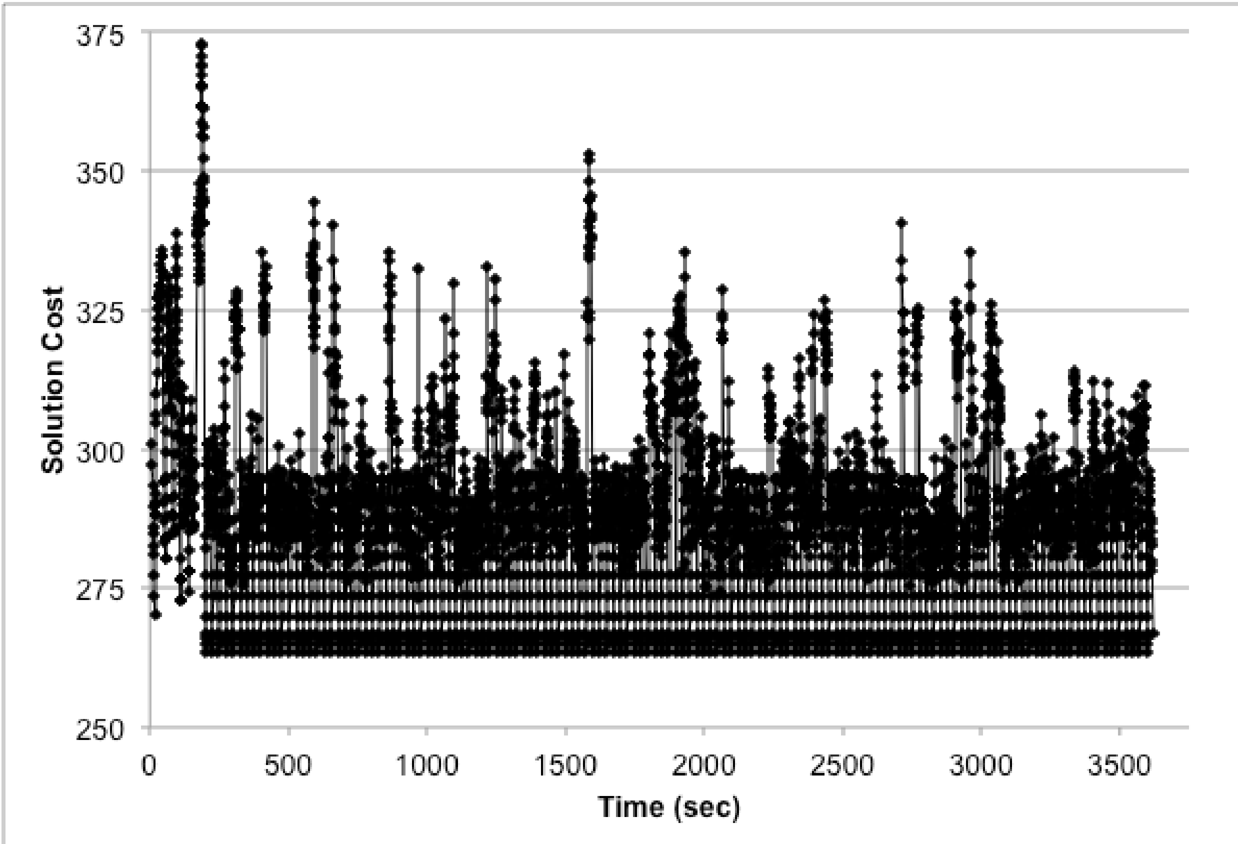

Figure 6: Solution Cost versus Time for Experiment #1 (*mval1A*, *numScenarios* = 10, *maxTabuIter* = 30)

Table 1: Summary of Experiment Parameters

Category	Names	Values
Instance parameters	*vehFixedCost*	0
	depotFixedCostFactor	1
	stdDevFactor	1/3
Model parameters	*numScenarios*	10, 30, 50
	maxTime	1 hr
	maxTabuIter	32, 125, 500

To better understand the robustness of these initial solutions to the stochastic instances, arc demands for 100 scenarios were generated for each of the 10 instances. Then, for each experiment, using the set of depots obtained from the full stochastic heuristic, vehicle routes were created using only the augment-merge phase across all 100 scenarios and an updated expected total cost was computed. This procedure was repeated using the set of depots obtained from the full deterministic heuristic and its solution cost was compared to its stochastic equivalent.

Of the 90 initial experiments, 30 resulted in a deterministic solution cost higher than its corresponding stochastic solution cost, indicating the value of considering variance in customer demand. The ratios of deterministic solution cost to stochastic solution cost for these 30 experiments range from 100.3 – 112.3% as shown in Figure 7. The same 30 experiments were observed across 9 of the 10 *mval* instances showing the potential benefits across a range of possible depot locations and numbers of demand arcs. For 2 of the 10 instances (*mval5A* and *mval9A*), the minimum solution cost using the depots obtained from the stochastic experiment was less than its deterministic counterparts.

Looking at the effect of the model parameters, each of the *numScenarios* and *maxTabuIter* values are equally represented in the 30 experiments. For the 10 highest solution cost ratios, *numScenarios* of 10 occurs the most frequently with 6 occurrences and *maxTabuIter* of 32 and 125 both appear 4 times.

As expected, the time spent in the tabu search phase exploring the neighborhood and choosing depot locations constituted the majority of the total solution processing time. For the 90 stochastic experiments, on average, the initialization step took 1.7% of the total solution time, the tabu search phase took 84.7%, and the augment-merge phase took 13.6%.

With customer demand magnitude being the primary difference between the stochastic and deterministic instances, the depot locations for the deterministic instances tended to perform well for their corresponding stochastic instances if the route capacities were unlikely to violate their respective vehicle capacities. In the initial 90 deterministic experiment solutions using the full heuristic, less than 3% of the 2045 routes had a probability greater than 0.000001 of exceeding the vehicle capacity. Of this subset, the average probability a route would exceed its vehicle capacity was only 32%. Thus, to further investigate the benefits of incorporating customer demand uncertainty in the depot location decisions, another set of 90 experiments was conducted using the same *mval* instances with modified vehicle capacity. The vehicle capacity in each instance was replaced with a capacity 1/8 times its original value. The same procedures listed above were conducted: 1) a stochastic version of each experiment was solved using the full tabu search, augment-merge heuristic; 2) a deterministic version of each experiment was solved using the full heuristic; 3) the set of depots obtained from the stochastic version was applied across 100 scenarios to evaluate its robustness; and 4) the set of depots obtained from the deterministic version was applied across 100 scenarios.

The modified instances resulted in 33 of the 90 experiments with stochastic solutions costs lower than their deterministic counterparts. They also spanned 9 of the 10 instances and ranged from just over 100.0% to 110.1%. Compared to the original instances, the number of instances in which the minimum solution cost using the depots obtained from the stochastic experiment was less than its deterministic counterparts doubled from 2 to 4. This highlights the importance of instance information, such as vehicle capacity, in determining whether considering customer demand uncertainty is a valuable endeavor.

Table 2: List of Experiments

Expt. #	Instance Name	*numScenarios*	*maxTabulter*
1	*mval1A*	10	125
2			32
3			500
4		30	125
5			32
6			500
7		50	125
8			32
9			500
10	*mval2A*	10	125
11			32
12			500
13		30	125
14			32
15			500
16		50	125
17			32
18			500
19	*mval3A*	10	125
20			32
21			500
22		30	125
23			32
24			500
25		50	125
26			32
27			500
28	*mval4A*	10	125
29			32
30			500
31		30	125
32			32
33			500
34		50	125
35			32
36			500
37	*mval5A*	10	125
38			32
39			500
40		30	125
41			32
42			500
43		50	125
44			32
45			500

Expt. #	Instance Name	*numScenarios*	*maxTabulter*
46	*mval6A*	10	125
47			32
48			500
49		30	125
50			32
51			500
52		50	125
53			32
54			500
55	*mval7A*	10	125
56			32
57			500
58		30	125
59			32
60			500
61		50	125
62			32
63			500
64	*mval8A*	10	125
65			32
66			500
67		30	125
68			32
69			500
70		50	125
71			32
72			500
73	*mval9A*	10	125
74			32
75			500
76		30	125
77			32
78			500
79		50	125
80			32
81			500
82	*mval10A*	10	125
83			32
84			500
85		30	125
86			32
87			500
88		50	125
89			32
90			500

Table 3: Comparison of Stochastic and Deterministic Solution Costs Using Depots from Original Instances with Full Vehicle Capacity Across 100 Scenarios

				STOCHASTIC COSTS				DETERMINISTIC COSTS				
Expt. #	Instance Name	*numScenarios*	*maxTabuIter*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	% of Stochastic Cost
1	*mval1A*	10	125	283.53	246.80	0.00	36.73	***261.03***	242.67	0.00	18.36	92.1%
2	*mval1A*	10	32	***263.41***	248.72	0.00	14.69	278.88	271.53	0.00	7.35	105.9%
3	*mval1A*	10	500	295.35	280.66	0.00	14.69	271.24	252.88	0.00	18.36	91.8%
4	*mval1A*	30	125	283.53	246.80	0.00	36.73	292.56	277.87	0.00	14.69	103.2%
5	*mval1A*	30	32	283.53	246.80	0.00	36.73	287.54	269.18	0.00	18.36	101.4%
6	*mval1A*	30	500	277.37	255.33	0.00	22.04	262.31	258.64	0.00	3.67	94.6%
7	*mval1A*	50	125	299.31	284.62	0.00	14.69	***261.03***	242.67	0.00	18.36	87.2%
8	*mval1A*	50	32	300.47	263.74	0.00	36.73	272.48	265.13	0.00	7.35	90.7%
9	*mval1A*	50	500	295.12	254.72	0.00	40.40	293.39	260.34	0.00	33.05	99.4%
10	*mval2A*	10	125	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
11	*mval2A*	10	32	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
12	*mval2A*	10	500	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
13	*mval2A*	30	125	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
14	*mval2A*	30	32	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
15	*mval2A*	30	500	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
16	*mval2A*	50	125	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
17	*mval2A*	50	32	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
18	*mval2A*	50	500	***366.28***	327.94	0.00	38.34	***366.28***	327.94	0.00	38.34	100.0%
19	*mval3A*	10	125	142.69	123.94	0.00	18.75	145.64	126.89	0.00	18.75	102.1%
20	*mval3A*	10	32	143.79	126.91	0.00	16.88	143.79	126.91	0.00	16.88	100.0%
21	*mval3A*	10	500	143.79	126.91	0.00	16.88	135.41	127.91	0.00	7.50	94.2%
22	*mval3A*	30	125	***135.11***	121.98	0.00	13.13	***133.61***	127.98	0.00	5.63	98.9%
23	*mval3A*	30	32	143.79	126.91	0.00	16.88	142.69	123.94	0.00	18.75	99.2%
24	*mval3A*	30	500	143.79	126.91	0.00	16.88	143.79	126.91	0.00	16.88	100.0%
25	*mval3A*	50	125	142.69	123.94	0.00	18.75	142.62	138.87	0.00	3.75	100.0%
26	*mval3A*	50	32	142.69	123.94	0.00	18.75	133.72	128.09	0.00	5.63	93.7%
27	*mval3A*	50	500	142.69	123.94	0.00	18.75	142.69	123.94	0.00	18.75	100.0%
28	*mval4A*	10	125	639.00	599.34	0.00	39.66	655.45	610.83	0.00	44.62	102.6%
29	*mval4A*	10	32	653.13	623.38	0.00	29.75	653.13	623.38	0.00	29.75	100.0%
30	*mval4A*	10	500	***624.55***	589.84	0.00	34.71	661.94	617.32	0.00	44.62	106.0%
31	*mval4A*	30	125	644.67	619.88	0.00	24.79	***607.81***	573.10	0.00	34.71	94.3%
32	*mval4A*	30	32	662.83	633.08	0.00	29.75	638.64	618.81	0.00	19.83	96.4%
33	*mval4A*	30	500	675.33	615.84	0.00	59.49	639.00	599.34	0.00	39.66	94.6%
34	*mval4A*	50	125	639.00	599.34	0.00	39.66	653.13	623.38	0.00	29.75	102.2%
35	*mval4A*	50	32	698.49	663.78	0.00	34.71	652.95	608.33	0.00	44.62	93.5%
36	*mval4A*	50	500	662.35	597.90	0.00	64.45	670.15	655.28	0.00	14.87	101.2%
37	*mval5A*	10	125	***676.03***	637.75	0.00	38.28	757.71	664.74	0.00	92.97	112.1%
38	*mval5A*	10	32	747.66	665.63	0.00	82.03	757.71	664.74	0.00	92.97	101.3%
39	*mval5A*	10	500	734.08	668.45	0.00	65.63	817.00	778.72	0.00	38.28	111.3%
40	*mval5A*	30	125	732.80	667.17	0.00	65.63	729.49	685.74	0.00	43.75	99.5%
41	*mval5A*	30	32	692.85	643.63	0.00	49.22	***694.98***	667.64	0.00	27.34	100.3%
42	*mval5A*	30	500	747.82	682.19	0.00	65.63	753.43	704.21	0.00	49.22	100.8%
43	*mval5A*	50	125	739.64	663.08	0.00	76.56	723.11	652.02	0.00	71.09	97.8%
44	*mval5A*	50	32	719.87	681.59	0.00	38.28	725.86	665.70	0.00	60.16	100.8%
45	*mval5A*	50	500	727.57	661.94	0.00	65.63	703.49	648.80	0.00	54.69	96.7%

Table 3 <continued>: Comparison of Stochastic and Deterministic Solution Costs Using Depots from Original Instances with Full Vehicle Capacity Across 100 Scenarios

				STOCHASTIC COSTS				DETERMINISTIC COSTS				
Expt. #	Instance Name	*numScenarios*	*maxTabuIter*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	% of Stochastic Cost
46	mval6A	10	125	390.52	363.64	0.00	26.88	386.52	355.80	0.00	30.72	99.0%
47	mval6A	10	32	***385.52***	354.80	0.00	30.72	***357.95***	338.75	0.00	19.20	92.8%
48	mval6A	10	500	***385.52***	354.80	0.00	30.72	390.40	355.83	0.00	34.57	101.3%
49	mval6A	30	125	389.39	362.51	0.00	26.88	386.51	374.99	0.00	11.52	99.3%
50	mval6A	30	32	***385.52***	354.80	0.00	30.72	393.00	381.48	0.00	11.52	101.9%
51	mval6A	30	500	396.45	365.73	0.00	30.72	390.57	359.85	0.00	30.72	98.5%
52	mval6A	50	125	424.91	340.42	0.00	84.49	390.40	355.83	0.00	34.57	91.9%
53	mval6A	50	32	390.37	355.80	0.00	34.57	392.13	353.72	0.00	38.41	100.5%
54	mval6A	50	500	***385.52***	354.80	0.00	30.72	390.40	355.83	0.00	34.57	101.3%
55	mval7A	10	125	***459.30***	425.08	0.00	34.22	491.61	453.59	0.00	38.02	107.0%
56	mval7A	10	32	461.39	427.17	0.00	34.22	483.43	441.60	0.00	41.83	104.8%
57	mval7A	10	500	475.19	429.56	0.00	45.63	***419.99***	404.78	0.00	15.21	88.4%
58	mval7A	30	125	482.56	440.73	0.00	41.83	503.09	461.26	0.00	41.83	104.3%
59	mval7A	30	32	480.45	434.82	0.00	45.63	474.51	440.29	0.00	34.22	98.8%
60	mval7A	30	500	467.10	402.46	0.00	64.64	463.18	425.16	0.00	38.02	99.2%
61	mval7A	50	125	484.88	446.86	0.00	38.02	480.58	438.75	0.00	41.83	99.1%
62	mval7A	50	32	480.58	438.75	0.00	41.83	444.35	421.54	0.00	22.81	92.5%
63	mval7A	50	500	471.97	395.92	0.00	76.05	432.15	416.94	0.00	15.21	91.6%
64	mval8A	10	125	718.23	685.29	0.00	32.94	652.07	619.13	0.00	32.94	90.8%
65	mval8A	10	32	***664.34***	631.40	0.00	32.94	***650.71***	639.73	0.00	10.98	97.9%
66	mval8A	10	500	693.86	655.43	0.00	38.43	714.63	687.18	0.00	27.45	103.0%
67	mval8A	30	125	723.71	685.28	0.00	38.43	652.07	619.13	0.00	32.94	90.1%
68	mval8A	30	32	***664.34***	631.40	0.00	32.94	714.63	687.18	0.00	27.45	107.6%
69	mval8A	30	500	715.98	677.55	0.00	38.43	714.63	687.18	0.00	27.45	99.8%
70	mval8A	50	125	701.32	657.40	0.00	43.92	714.63	687.18	0.00	27.45	101.9%
71	mval8A	50	32	686.42	620.54	0.00	65.88	715.98	677.55	0.00	38.43	104.3%
72	mval8A	50	500	705.54	623.20	0.00	82.34	655.49	628.04	0.00	27.45	92.9%
73	mval9A	10	125	593.47	525.34	0.00	68.13	583.25	518.08	0.00	65.17	98.3%
74	mval9A	10	32	***568.74***	509.50	0.00	59.24	583.22	518.05	0.00	65.17	102.5%
75	mval9A	10	500	596.99	534.79	0.00	62.20	582.84	517.67	0.00	65.17	97.6%
76	mval9A	30	125	592.31	527.14	0.00	65.17	579.31	564.50	0.00	14.81	97.8%
77	mval9A	30	32	587.80	522.63	0.00	65.17	586.80	521.63	0.00	65.17	99.8%
78	mval9A	30	500	577.39	518.15	0.00	59.24	580.39	541.88	0.00	38.51	100.5%
79	mval9A	50	125	587.80	522.63	0.00	65.17	579.87	517.67	0.00	62.20	98.7%
80	mval9A	50	32	600.52	529.43	0.00	71.09	***578.27***	542.72	0.00	35.55	96.3%
81	mval9A	50	500	590.76	522.63	0.00	68.13	606.61	544.41	0.00	62.20	102.7%
82	mval10A	10	125	798.01	754.57	0.00	43.44	784.95	717.81	0.00	67.14	98.4%
83	mval10A	10	32	804.03	740.84	0.00	63.19	799.96	732.82	0.00	67.14	99.5%
84	mval10A	10	500	794.46	747.07	0.00	47.39	746.20	706.71	0.00	39.49	93.9%
85	mval10A	30	125	784.95	717.81	0.00	67.14	792.68	749.24	0.00	43.44	101.0%
86	mval10A	30	32	797.38	746.04	0.00	51.34	753.06	701.72	0.00	51.34	94.4%
87	mval10A	30	500	***761.69***	686.65	0.00	75.04	778.11	703.07	0.00	75.04	102.2%
88	mval10A	50	125	799.96	732.82	0.00	67.14	***735.52***	696.03	0.00	39.49	91.9%
89	mval10A	50	32	820.54	753.40	0.00	67.14	753.88	734.13	0.00	19.75	91.9%
90	mval10A	50	500	807.48	740.34	0.00	67.14	795.39	728.25	0.00	67.14	98.5%

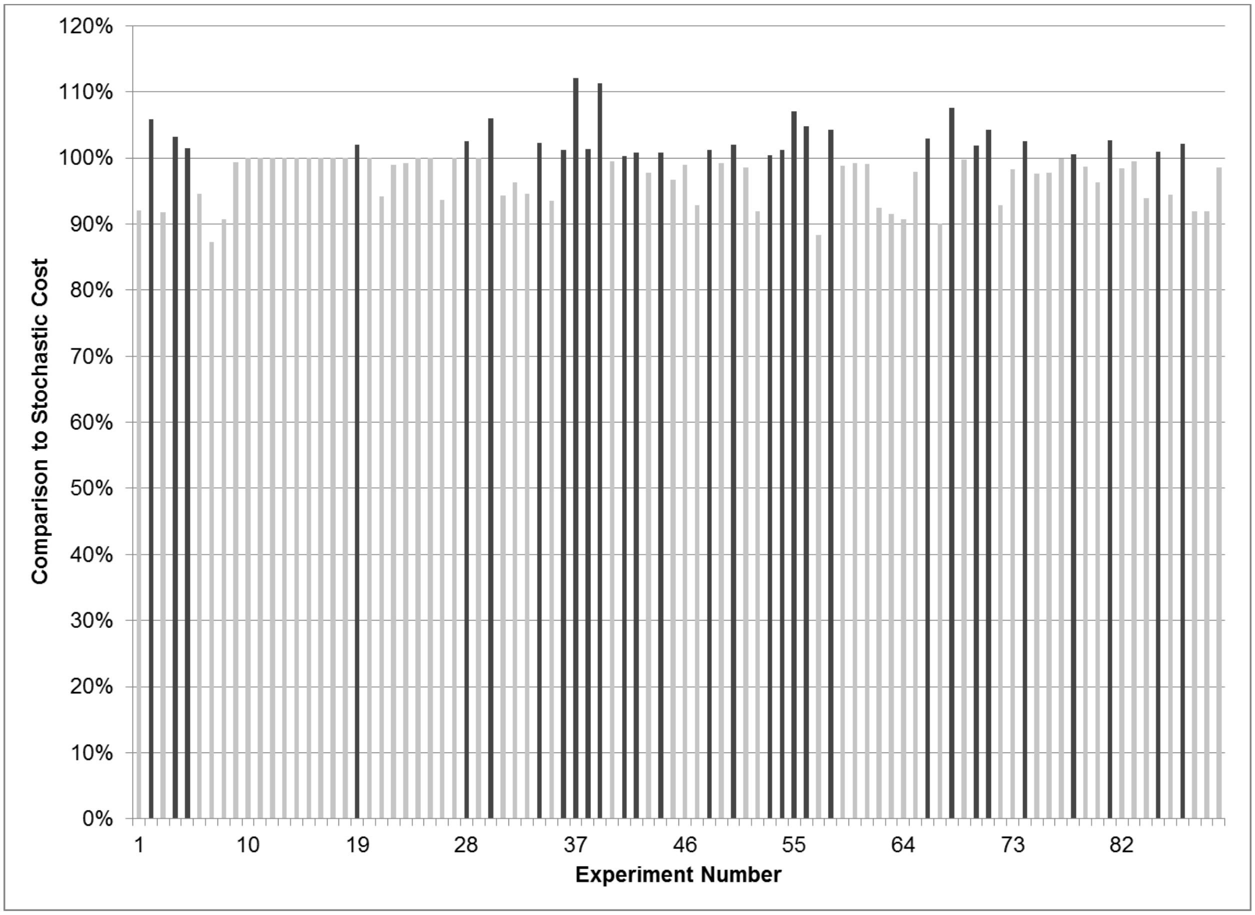

Figure 7: Comparison of Stochastic and Deterministic Solution Costs Using Depots from Original Instances with Full Vehicle Capacity Across 100 Scenarios

Table 4: Comparison of Stochastic and Deterministic Solution Costs Using Depots from Modified Instances with 1/8 * Vehicle Capacity Across 100 Scenarios

				STOCHASTIC COSTS				DETERMINISTIC COSTS				
Expt. #	Instance Name	*numScenarios*	*maxTabuIter*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	% of Stochastic Cost
1	*mval1A*	10	125	322.63	296.92	0.00	25.71	328.04	302.33	0.00	25.71	101.7%
2	*mval1A*	10	32	309.73	280.35	0.00	29.38	334.40	290.33	0.00	44.07	108.0%
3	*mval1A*	10	500	329.60	289.20	0.00	40.40	329.59	292.86	0.00	36.73	100.0%
4	*mval1A*	30	125	320.77	295.06	0.00	25.71	331.80	291.40	0.00	40.40	103.4%
5	*mval1A*	30	32	337.40	297.00	0.00	40.40	331.95	295.22	0.00	36.73	98.4%
6	*mval1A*	30	500	322.08	296.37	0.00	25.71	329.99	285.92	0.00	44.07	102.5%
7	*mval1A*	50	125	***300.40***	271.02	0.00	29.38	330.73	290.33	0.00	40.40	110.1%
8	*mval1A*	50	32	320.39	283.66	0.00	36.73	***322.65***	285.92	0.00	36.73	100.7%
9	*mval1A*	50	500	331.38	294.65	0.00	36.73	336.33	295.93	0.00	40.40	101.5%
10	*mval2A*	10	125	460.30	405.53	0.00	54.77	454.26	410.44	0.00	43.82	98.7%
11	*mval2A*	10	32	461.26	428.40	0.00	32.86	447.62	409.28	0.00	38.34	97.0%
12	*mval2A*	10	500	451.94	408.12	0.00	43.82	439.48	401.14	0.00	38.34	97.2%
13	*mval2A*	30	125	473.95	435.61	0.00	38.34	431.17	387.35	0.00	43.82	91.0%
14	*mval2A*	30	32	461.94	407.17	0.00	54.77	438.36	394.54	0.00	43.82	94.9%
15	*mval2A*	30	500	***441.30***	392.00	0.00	49.30	433.59	389.77	0.00	43.82	98.3%
16	*mval2A*	50	125	457.96	408.66	0.00	49.30	***428.21***	384.39	0.00	43.82	93.5%
17	*mval2A*	50	32	462.06	412.76	0.00	49.30	433.59	389.77	0.00	43.82	93.8%
18	*mval2A*	50	500	469.18	414.41	0.00	54.77	***428.21***	384.39	0.00	43.82	91.3%
19	*mval3A*	10	125	162.76	142.13	0.00	20.63	164.39	149.39	0.00	15.00	101.0%
20	*mval3A*	10	32	155.09	141.96	0.00	13.13	159.35	142.47	0.00	16.88	102.7%
21	*mval3A*	10	500	163.15	144.40	0.00	18.75	159.38	146.25	0.00	13.13	97.7%
22	*mval3A*	30	125	***151.30***	140.05	0.00	11.25	***158.67***	145.54	0.00	13.13	104.9%
23	*mval3A*	30	32	163.09	144.34	0.00	18.75	158.67	145.54	0.00	13.13	97.3%
24	*mval3A*	30	500	158.14	139.39	0.00	18.75	164.39	149.39	0.00	15.00	104.0%
25	*mval3A*	50	125	157.04	147.66	0.00	9.38	154.27	143.02	0.00	11.25	98.2%
26	*mval3A*	50	32	153.62	142.37	0.00	11.25	164.39	149.39	0.00	15.00	107.0%
27	*mval3A*	50	500	168.79	146.29	0.00	22.50	151.39	134.51	0.00	16.88	89.7%
28	*mval4A*	10	125	***712.28***	642.87	0.00	69.41	730.35	690.69	0.00	39.66	102.5%
29	*mval4A*	10	32	763.08	693.67	0.00	69.41	716.69	652.24	0.00	64.45	93.9%
30	*mval4A*	10	500	722.55	668.01	0.00	54.54	736.49	686.91	0.00	49.58	101.9%
31	*mval4A*	30	125	740.51	671.10	0.00	69.41	731.82	702.07	0.00	29.75	98.8%
32	*mval4A*	30	32	727.28	657.87	0.00	69.41	710.79	646.34	0.00	64.45	97.7%
33	*mval4A*	30	500	731.27	656.90	0.00	74.37	728.59	659.18	0.00	69.41	99.6%
34	*mval4A*	50	125	712.81	648.36	0.00	64.45	***695.52***	650.90	0.00	44.62	97.6%
35	*mval4A*	50	32	747.53	668.20	0.00	79.33	740.40	690.82	0.00	49.58	99.0%
36	*mval4A*	50	500	725.56	651.19	0.00	74.37	757.01	702.47	0.00	54.54	104.3%
37	*mval5A*	10	125	878.02	790.52	0.00	87.50	814.69	738.13	0.00	76.56	92.8%
38	*mval5A*	10	32	837.51	760.95	0.00	76.56	821.57	766.88	0.00	54.69	98.1%
39	*mval5A*	10	500	876.51	794.48	0.00	82.03	***806.04***	740.41	0.00	65.63	92.0%
40	*mval5A*	30	125	838.26	767.17	0.00	71.09	838.33	761.77	0.00	76.56	100.0%
41	*mval5A*	30	32	840.61	758.58	0.00	82.03	848.86	772.30	0.00	76.56	101.0%
42	*mval5A*	30	500	839.98	768.89	0.00	71.09	810.32	755.63	0.00	54.69	96.5%
43	*mval5A*	50	125	838.44	767.35	0.00	71.09	831.16	771.00	0.00	60.16	99.1%
44	*mval5A*	50	32	856.33	779.77	0.00	76.56	819.37	742.81	0.00	76.56	95.7%
45	*mval5A*	50	500	***815.25***	744.16	0.00	71.09	810.43	744.80	0.00	65.63	99.4%

Table 4 <continued>: Comparison of Stochastic and Deterministic Solution Costs Using Depots from Modified Instances with 1/8 * Vehicle Capacity Across 100 Scenarios

				STOCHASTIC COSTS				DETERMINISTIC COSTS				
Expt. #	Instance Name	*numScenarios*	*maxTabuIter*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	*totCost*	*varRouteCost*	*fixedRouteCost*	*fixedDepotCost*	% of Stochastic Cost
46	mval6A	10	125	461.94	419.69	0.00	42.25	448.95	399.02	0.00	49.93	97.2%
47	mval6A	10	32	451.63	409.38	0.00	42.25	441.27	414.39	0.00	26.88	97.7%
48	mval6A	10	500	457.81	423.24	0.00	34.57	465.13	422.88	0.00	42.25	101.6%
49	mval6A	30	125	460.36	410.43	0.00	49.93	441.99	392.06	0.00	49.93	96.0%
50	mval6A	30	32	***432.78***	390.53	0.00	42.25	446.66	392.89	0.00	53.77	103.2%
51	mval6A	30	500	472.14	426.05	0.00	46.09	***441.27***	399.02	0.00	42.25	93.5%
52	mval6A	50	125	458.63	408.70	0.00	49.93	455.07	432.03	0.00	23.04	99.2%
53	mval6A	50	32	458.22	415.97	0.00	42.25	455.12	405.19	0.00	49.93	99.3%
54	mval6A	50	500	450.65	400.72	0.00	49.93	***441.27***	399.02	0.00	42.25	97.9%
55	mval7A	10	125	543.29	463.44	0.00	79.85	525.18	456.74	0.00	68.44	96.7%
56	mval7A	10	32	502.33	464.31	0.00	38.02	534.38	469.74	0.00	64.64	106.4%
57	mval7A	10	500	519.09	473.46	0.00	45.63	526.43	484.60	0.00	41.83	101.4%
58	mval7A	30	125	***490.56***	444.93	0.00	45.63	515.73	466.30	0.00	49.43	105.1%
59	mval7A	30	32	535.13	493.30	0.00	41.83	527.39	462.75	0.00	64.64	98.6%
60	mval7A	30	500	514.77	453.93	0.00	60.84	534.40	469.76	0.00	64.64	103.8%
61	mval7A	50	125	540.23	464.18	0.00	76.05	519.32	481.30	0.00	38.02	96.1%
62	mval7A	50	32	537.99	492.36	0.00	45.63	529.06	494.84	0.00	34.22	98.3%
63	mval7A	50	500	513.98	449.34	0.00	64.64	***504.37***	454.94	0.00	49.43	98.1%
64	mval8A	10	125	802.71	736.83	0.00	65.88	805.36	744.97	0.00	60.39	100.3%
65	mval8A	10	32	832.15	760.79	0.00	71.36	824.79	742.45	0.00	82.34	99.1%
66	mval8A	10	500	791.51	714.66	0.00	76.85	820.32	759.93	0.00	60.39	103.6%
67	mval8A	30	125	805.93	734.57	0.00	71.36	798.49	738.10	0.00	60.39	99.1%
68	mval8A	30	32	***781.61***	721.22	0.00	60.39	811.99	746.11	0.00	65.88	103.9%
69	mval8A	30	500	805.49	739.61	0.00	65.88	795.78	713.44	0.00	82.34	98.8%
70	mval8A	50	125	810.92	739.56	0.00	71.36	***781.09***	715.21	0.00	65.88	96.3%
71	mval8A	50	32	784.14	707.29	0.00	76.85	808.58	742.70	0.00	65.88	103.1%
72	mval8A	50	500	799.15	722.30	0.00	76.85	815.40	749.52	0.00	65.88	102.0%
73	mval9A	10	125	639.25	568.16	0.00	71.09	610.56	548.36	0.00	62.20	95.5%
74	mval9A	10	32	***592.52***	530.32	0.00	62.20	607.49	542.32	0.00	65.17	102.5%
75	mval9A	10	500	632.92	567.75	0.00	65.17	599.05	533.88	0.00	65.17	94.6%
76	mval9A	30	125	615.87	547.74	0.00	68.13	627.20	562.03	0.00	65.17	101.8%
77	mval9A	30	32	616.61	548.48	0.00	68.13	610.42	545.25	0.00	65.17	99.0%
78	mval9A	30	500	623.95	561.75	0.00	62.20	608.50	546.30	0.00	62.20	97.5%
79	mval9A	50	125	610.39	551.15	0.00	59.24	635.93	573.73	0.00	62.20	104.2%
80	mval9A	50	32	658.80	587.71	0.00	71.09	***587.53***	525.33	0.00	62.20	89.2%
81	mval9A	50	500	614.14	548.97	0.00	65.17	600.07	526.02	0.00	74.05	97.7%
82	mval10A	10	125	810.33	731.34	0.00	78.99	***788.45***	748.96	0.00	39.49	97.3%
83	mval10A	10	32	819.59	748.50	0.00	71.09	808.93	749.69	0.00	59.24	98.7%
84	mval10A	10	500	805.69	730.65	0.00	75.04	792.30	721.21	0.00	71.09	98.3%
85	mval10A	30	125	828.16	753.12	0.00	75.04	824.24	757.10	0.00	67.14	99.5%
86	mval10A	30	32	830.14	755.10	0.00	75.04	824.61	749.57	0.00	75.04	99.3%
87	mval10A	30	500	821.16	746.12	0.00	75.04	800.01	760.52	0.00	39.49	97.4%
88	mval10A	50	125	844.21	769.17	0.00	75.04	814.80	743.71	0.00	71.09	96.5%
89	mval10A	50	32	***795.01***	727.87	0.00	67.14	831.83	760.74	0.00	71.09	104.6%
90	mval10A	50	500	808.84	733.80	0.00	75.04	814.50	755.26	0.00	59.24	100.7%

5. SUMMARY AND CONCLUSIONS

This paper presented a framework for a tabu search, augment-merge heuristic to solve a stochastic LARP – a problem whose objective is to site depots and create vehicle routes given uncertain customer demand and minimize overall depot location and routing costs. Different scenarios represent the potential realizations of customer demand. The heuristic is split into two phases: the tabu search phase seeking to minimize the depot location costs or depot removal/insertion costs from/into a given route of customer arcs across scenarios and the augment-merge phase aiming to minimize the vehicle routing costs for each scenario given a fixed set of depot locations.

A set of 10 existing deterministic LARP instances were transformed into stochastic versions of the problem and 9 experiments with varying number of scenarios (10, 30, 50) and maximum tabu iterations (32, 125, 500) were conducted for each. One-third of the initial 90 experiments showed the potential value of considering stochastic customer demand with a ratio of deterministic solution cost to stochastic solution cost greater than 1.

A second set of 90 experiments was conducted using value the same instance and model parameters as the first set of experiments and a modified vehicle capacity 1/8 times its original value. Compared to the original instances, the modified versions resulted in twice as many instances in which the depots obtained from the stochastic solution outperformed the depots obtained from the deterministic solution, indicating the algorithm's sensitivity to the instance input information, such as vehicle capacity.

These initial experiments show promising value for the proposed stochastic LARP algorithm. Because each experiment was conducted only once, performing a larger number replications using the full tabu search, augment-merge heuristic for both stochastic and deterministic experiments may help to mitigate any effects of a particularly skewed set of scenarios and demonstrate even greater potential benefits.

Another suggestion for future research is to explore other intensification and diversification techniques and investigate different types of neighborhoods to continue improving the tabu search phase. For example, when choosing a flip to implement in the neighborhood search, instead of choosing the first flip resulting in the lowest savings, a depot can be randomly chosen among all flips resulting in positive cost savings.

The current heuristic spends the majority of its time in the tabu search phase seeking the best depot locations, and the augment-merge phase was chosen for its relative solution speed. However, the augment-merge portion of the heuristic can also be further improved by allowing pairs of non-adjacent routes (i.e., two routes that do not share a common node) to be merged. Furthermore, for the scope of this project, the augment-merge heuristic was chosen for its computational speed in creating vehicle routes, but given the existing selection of CARP algorithms, other methods can be substituted.

In addition to improving the details of the two phases of the algorithm, instead of examining the demand magnitudes in a given instance, demand presence (or absence) on arcs in the network is another area for further exploration and could be represented using binary variables.

6. REFERENCES

Amaya, A., Langevin, A., Trépanier, M. (2007). The capacitated arc routing problem with refill points. *Operations Research Letters, 35*, 45-53.

Amaya, A., Langevin, A., Trépanier, M. (2010). A heuristic method for the capacitated arc routing problem with refill points and multiple loads. *Journal of the Operational Research Society, 61*, 1095-1103.

Belenguer, J. M. & Benavent, E. (2003a). A cutting plane algorithm for the capacitated arc routing problem. *Computers & Operations Research, 30*, 705-728.

Belenguer, J. M. & Benavent, E. (2003b.) Capacitated arc routing problem instances. Retrieved April 28, 2016, from http://www.uv.es/~belengue/carp.html.

Belenguer, J. M., Benavent, E., Lacomme, P., Prins, C. (2006a). Lower and upper bounds for the mixed capacitated arc routing problem. *Computers & Operations Research, 33*, 3363-3383.

Belenguer, J. M., Benavent, E., Lacomme, P., Prins, C. (2006b). The mixed capacitated arc routing problem. Retrieved April 28, 2016, from http://www.uv.es/belengue/mcarp/.

Berhan, E., Beshah, B., Kitaw, D. (2014). Stochastic Vehicle Routing Problem: A Literature Survey. *Journal of Information & Knowledge Management, 13(3)*, 1450022-1-12

Bramel, J. B., Simchi-Levi, D. (1995). A location based heuristic for general routing problems. *Operations Research, 43(4)*, 649-660.

Christiansen, C. H., Lysgaard, J., Wøhlk, S. (2009). A branch-and-price algorithm for the capacitated arc routing problem with stochastic demands. *Operations Research Letters, 37*, 392-398.

Christofides, N. (1973). The optimum traversal of a graph. *Omega, 1(6)*, 719-735.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research, 12(4)*, 568–581

Daskin, M. S. (2013). *Network and discrete location: Models, algorithms, and applications (2nd ed.)*. Hoboken, New Jersey: Wiley.

Eiselt, H. A., Gendreau, M., Laporte, G. (1995a) Arc routing problems, part I: The Chinese postman problem. *Operations Research, 42(2)*, 231-242.

Eiselt, H. A., Gendreau, M., Laporte, G. (1995b) Arc routing problems, part II: The rural postman problem. *Operations Research, 42(3),* 399-414.

Eksioglu, B., Vural, A.V., Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers and Industrial Engineering, 57(4)*, 1472-1483.

El-Sherbeny, N. A. (2010). Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. *Journal of King Saud University (Science), 22,* 123–131.

Fisher, M. L., Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. *Networks, 11*, 109-124.

Ford, L. R. & Fulkerson, D. R. (1962). *Flows in Networks*. Princeton, N.J.: Princeton University Press.

Frederickson, G. N. (1979). Approximation algorithms for some postman problems. *Journal of the ACM, 26*, 538-554.

Gendreau, M., Laporte, G., Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research, 88,* 3-12.

Ghiani, G. & Laporte, G. (1999). Eulerian location problems. *Networks, 34*: 291–302

Ghiani, G. & Laporte, G. (2001). Location-arc routing problems. *Opsearch, 38(2),* 151-9.

Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations Research, 22(2),* 340–349.

Golden, B. L., DeArmon, J. S., Baker, E.K. (1983). Computation experiments with algorithms for a class of routing problems. *Computers & Operations Research, 10(1)*, 47-59.

Golden, B. L., Wong, R. T. (1981). Capacitated arc routing problems. *Networks,11(3)*, 305–15.

Greenaway, M. (2004). Chinese postman problem. Retrieved April 28, 2016, from http://www.suffolkmaths.co.uk/pages/Maths%20Projects/Projects/Topology%20and%20Graph%20Theory/Chinese%20Postman%20Problem.pdf.

Hashemi Doulabi, S. H. & Seifi, A. (2013). Lower and upper bounds for location-arc routing problems with vehicle capacity constraints. *European Journal of Operational Research, 224,* 189-28.

Hertz, A. & Mittaz, M. (2008). *Pickup and delivery problems with services on nodes or arcs of a network*. In: Finke, G., editor. Operations Research and Networks. London, UK: ISTE, Chapter 6, 1-22.

Hertz, A., Laporte, G., Nanchen P. (1999). Improvement procedure for the undirected rural postman problem. *INFORMS Journal on Computing, 11*, 53-62.

Laporte, G. (1988). *Location-routing problems*. In: Golden, B. L., Assad, A. A., editors. Vehicle routing: methods and studies. Amsterdam, The Netherlands: Elsevier, 163–198.

Laporte, G. (1992). The vehicle routing problem: an overview of exact and approximate algorithms. *European Journal of Operational Research, 59*, 345-358.

Laporte, G., Gendreau, M. , Potvin, J. Y., Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research, 7,* 285-300

Laporte, G., Louveaux, F., 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., Musmanno, R., Vocaturo, F. (2010). An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands. *Transportation Science, 44(1),* 125-135.

Levy, L., Bodin, L. (1989). The arc oriented location routing problem. *INFOR, 27(1),* 74–94.

Liu, T., Jiang, Z., Chen, F., Liu, R., Liu, S. (2008). Proceedings from the IEEE/SOLI 2008. IEEE International Conference: *Combined location-arc routing problems: A survey and suggestions for future research*, 2336-2341.

Longo, H., Poggi de Aragão, M., Uchoa, E. (2006). Solving capacitated arc routing problems using a transformation to the CVRP. *Computers & Operations Research, 33*, 1823-1837.

Lopes, R. B., Ferreira, C., Santos, B. S., Barreto, S. (2013). A taxonomical analysis, current methods and objectives on location-routing problems. *International Transactions in Operational Research, 20,* 795-822.

Lopes, R. B., Plastria, F., Ferreira, C., Santos, B. S. (2014). Location-arc routing problem: Heuristic approaches and test instances. *Computers & Operations Research, 43,* 309-317.

Mei, Y., Tang, K., Yao, X. (2010). Proceedings from the 2010 IEEE Congress Evolutionary Computation (CEC): *Capacitated arc routing problem in uncertain environments,* 2336-2341.

Michel, L., Van Hentenryck, P. (2004). A simple tabu search for warehouse location. *European Journal of Operational Research, 157*, 576-591.

Nagy, G. & Salhi, S. (2005). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research, 162*, 126-141.

Nagy, G. & Salhi, S. (2007). Location-routing: issues, models and methods. *European Journal of Operational Research, 177*, 649-672.

Owen, S. H., Daskin M. S. (1998). Strategic facility location: A review. *European Journal of Operational Research, 111*, 423-447.

Perrier, N., Langevin, A., Campbell, J. (2007). A survey of models and algorithms for winter road maintenance. Part III: Vehicle routing and depot location for spreading. *Computers & Operations Research, 24*, 211-257.

Polacek, M., Doerner, K. F., Hartl, R.F., Maniezzo, V. (2008). A variable neighborhood search for the capacitated arc routing problem with intermediate facilities. *Journal of Heuristics, 14(5),* 405–23.

Prins, C., Lacomme, P., Prodhon, C. (2014). Order-first split-second methods for vehicle routing problems: A review. *Transportation Research Part C, 40*, 179-200.

Prodhon, C. & Prins, C. (2014). A survey of recent research on location-routing problems. *European Journal of Operational Research, 238*, 1-17.

Salhi, S. & Rand, G. K. (1989). The effect of ignoring routes when location depots. *European Journal of Operational Research, 39*, 150-6.

Sathyanarayanan, S. & Suresh Joseph, K. (2014). Proceedings from International Conference on Information Communication and Embedded Systems (ICICES): *A survey on stochastic vehicle routing problem*. Chennai, India.

Segerstedt, A. (2014). A simple heuristic for vehicle routing – A variant of Clarke and Wright's saving method. *International Journal of Production Economics, 157,* 74-79.

Snyder, L. V. (2005). Facility location under uncertainty: a review. *IIE Transactions, 38,* 537-554.

Sun, M. (2006). Solving the uncapacitated facility location problem using tabu search. *European Journal of Operational Research, 33*, 2563-2589.

Thimbleby, H. (2003). The Directed Chinese Postman Problem. *Software: Practice & Experience, 33(11),* 1081–1096.

Ulusoy, G. (1985). The fleet size and mix problem for capacitated arc routing. *European Journal of Operational Research, 22(3),* 329-337.

Wøhlk, S. (2005). *Contributions to Arc Routing – Chapter 2: State of the Art*. (Unpublished doctoral dissertation). University of Southern Denmark, Odense, Denmark.

Wøhlk, S. (2008). A *decade of capacitated arc routing*. In: Golden, B., Raghavan, S., Wasil, E. editors. The vehicle routing problem: Latest advances and new challenges. United States: Springer, 29-48.