Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2007)

Solving Multicommodity Capacitated Network Design Problems using a Multiobjective Evolutionary Algorithm

Mark P. Kleeman, Gary B. Lamont, Kenneth M. Hopkinson and Scott R. Graham
Air Force Institute of Technology
Department of Electrical and Computer Engineering
Graduate School of Engineering & Management
Wright Patterson Air Force Base, Dayton, OH 45433

Abstract— **Evolutionary algorithms have been applied to a variety of network flow problems with acceptable results. In this research, a multiobjective evolutionary algorithm (MOEA) is used to solve a variation of the multicommodity capacitated network design problem (MCNDP). This variation represents a hybrid communication network as found in network centric models with multiple objectives including costs, delays, robustness, vulnerability, and reliability. Nodes in such centric systems can have multiple and varying link capacities, rates and information (commodity) quantities to be delivered and received. Each commodity can have an independent prioritized bandwidth requirement as well. Insight to the MCNDP problem domain and Pareto structure is developed. The nondominated sorting genetic algorithm (NSGA-II) is modified and extended to solve such a MCNDP. Since the MCNDP is highly constrained, a novel initialization procedure and mutation method are also integrated into this MOEA. Empirical results and analysis indicate that effective solutions are generated very efficiently.**

I. Introduction

The basis of military doctrine is presented in the Department of Defense document entitled Joint Vision (JV) 2020. The document outlines goals that attempt to provide the formation of a joint military force that is dominant in all aspects of military operations. Information superiority is one of the key tenets outlined in the document. Having information superiority allows for network centric warfare (NCW). NCW attempts to translate an information advantage into military advantage through the use of a geographically dispersed robust network. This network has many nodes, each of which has information that may be of value to another node. Therefore the design of robust networks is paramount in the development of associated NCW.

This research focuses on the design of robust communications networks. We use a variant of the fairly detailed network design problem (NDP) model called the multicommodity capacitated network design problem (MCNDP). The MCNDP is an NP-complete problem [1] that models many key aspects of a communications network. This detailed model enables us to generate solutions that can effectively meet mission objectives in an efficient manner.

The views expressed in this article are those of the authors and do not reflect the official policy of the United States Air Force, Department of Defense, or the United States Government.

This research consists of the following sections. Section II presents a high level overview of the network design problem and the methods that researchers have used to solve it. The section then delves deeper into the MCNDP approach and discusses some of the contemporary research. The complexity of the problem is discussed, as well as some of the other stochastic approaches that researchers have used. Section III discusses the MCNDP problem domain in more detail. Specific problem details are discussed including the mathematical problem formulation, problem objectives, and the problem difficulty. Section IV presents the MOEA design. The section briefly touches on MOEAs and the specific MOEA used, a modified NSGA-II (M-NSGA-II). The section concludes with a discussion of how the problem domain is integrated into the algorithm. Section V presents the results of the experiments and provides analysis of the results.

II. Network Design Problem Background

Historically, there are many types of NDPs that researchers have modelled in order to solve real world problems. They range from fairly simplistic models to more in depth models that allow for more flexible network design parameters. The main issue with all networks is being able to move a commodity from its source node to its destination node. The commodity can be information or some type of product. If the network being modelled is very simplistic, a variation of the network flow problem can be implemented. If the model is more complex, then a variation of the MCNDP would be more appropriate. This section briefly discusses several approaches for solving the NDP and lists the advantages and disadvantages of each.

A. Network Flow Problems

Network flow problem have been researched as far back as 1736, when the famous Swiss mathematician Leonhard Euler proved that the Königsberg bridges problem could not be solved. There are many other examples of network flow problems, such as the Chinese Postman Problem, but most of these are not well suited to model a communication network. Many models incorporate flows that start and stop at the same node, whereas communication networks model flows from multiple points to other points in the network. Speed and reliability are two of the major criteria when designing a communication network. In general, communication network models typically examine one or more of the following components [2]:

1) The physical hardware - nodes and arcs (computers, routers, communication lines, etc)
2) The information packets (commodities) - bandwidth required, priority, etc.
3) The limited capability of the physical hardware to handle the information - capacity limits on the arcs, buffers in the nodes, etc.

Given the specifications of the communication network model, a researcher can use a variety of network flow problem models to solve a specific problem. This section briefly discusses a few of the network flow models that can be employed to aid in the design of a communication network. Approaches to solving these problems are presented, the complexity of the problem is addressed, as well as stochastic approaches researchers have previously applied to the problem. First, we present the approaches to the problem.

1) Minimum Cost Flow Problem: One of the simplest communication network models is the minimum cost flow problem. This problem finds the minimum cost between two nodes, based on the arc costs. For this problem, each network arc has a predetermined length or cost. The goal is to simply connect the nodes via the arcs to produce the shortest aggregate paths for the network. This simplistic model does not take into account any constraints at the nodes or arcs, so it is not an ideal choice to use for designing a communications network.

2) The Shortest Path Problem: The shortest path problem is similar to the minimum cost flow problem, but it introduces upper and lower bounds on the arc flows. Like the minimum cost flow problem, this problem simply finds the shortest path between two nodes. Each network arc has a predetermined length or cost. This model does have some constraints on the arcs, so it is more realistic than the minimum cost flow problem, but it doesn't take into account the commodities that need to flow on the network.

B. Uncapacitated Network Design Problem

In the uncapacitated NDP Ahuja et al [3] the network must route multiple commodities. Each commodity k has a single source node s^k and a single destination node d^k. This problem differs slightly from our problem in that each arc introduced into the network has enough capacity to route the flow of all of the commodities present on the network. This research is geared more toward the real-world, where network connections have finite capacities, and the arcs may not be able to handle large amounts of commodity flow.

A mathematical description of the uncapacitated NDP model is [3–5]:

Let x^k denote the vector of flows of commodity k on the network.

Let x_{ij}^k denote the fraction of the required flow of commodity k to be routed from the source s^k to the destination d^k that flows on arc (i,j).

Let c^k denote the cost vector for commodity k (c_{ij}^k is the per unit cost for commodity k on arc (i,j) multiplied by the flow requirement of that commodity).

Let f denote the fixed cost vector for the construction of each arc in the network.

Let y_{ij} be a zero-one variable indicating whether arc (i,j) is selected as part of the network design.

Given these definitions, Equations 1 through 5 mathematically describe the problem.

$$\text{Minimize} \sum_{1\le k\le K} c^k x^k + fy \tag{1}$$

subject to

$$\sum_{\{j:(i,j)\in A\}} x_{ij}^k - \sum_{\{j:(j,i)\in A\}} x_{ji}^k = \begin{cases} 1 & \text{if } i = s^k \\ -1 & \text{if } i = d^k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$\forall i \in N, k = 1,2,\ldots,K$$

$$x_{ij}^k \le y_{ij} \qquad \forall (i,j) \in A, k = 1,2,\ldots,K \tag{3}$$

$$x_{ij}^k \ge 0 \qquad \forall (i,j) \in A, k = 1,2,\ldots,K \tag{4}$$

$$y_{ij} \text{is binary} \qquad \forall (i,j) \in A \tag{5}$$

Simply stated, the goal of the uncapacitated network design problem is to find the lowest cost of a network based on both fixed and variable costs. Fixed costs are the number of arcs fy that need to be constructed for the network. The variable costs are a summation of the costs of the commodities flowing over the necessary arcs to reach their destination. The constraints make this problem more difficult. Node balance constraints, listed in Equation 2, ensure that 100% of each commodity leaves its source node and arrives at its destination. It also ensures that the intermediate nodes are balanced, i.e. they send the same amount of a commodity that they receive. Constraints 3 through 5 ensure that a link is present for every commodity flow, the commodity flows are non-negative, and arcs are either present or not. As stated earlier, a limitation of this model is that arcs have unlimited capacity. This unlimited capacity allows for all commodities to flow across the network. In reality, it may be necessary to drop a commodity in order to meet arc capacity constraints. This leads into the multicommodity capacitated network design problem.

C. Multicommodity Capacitated Network Design Problem (MCNDP)

The MCNDP is similar to the uncapacitated NDP with capacity limits included with each arc. A good mathematical formulation of the problem is presented in [1] and [6]. The MCNDP formulation is similar to the variant used in this research, and discussed in further detail in Section III-B. The MCNDP formulation views dropped commodities as constraints. In this research, a dropped commodity is allowed, but a steep penalty is applied for every dropped commodity. This type of implementation allows the algorithm to find solutions to networks that may otherwise be unsolvable due to the capacity limits on the arcs.

D. MCNDPs and the Network Centric Framework

MCNDPs can have as objectives costs, delays, robustness, vulnerability, and reliability with associated constraints of network flow capacities, rates, and quantities of information. The optimization of such complex MCNDPs is an integral but low-level element of *network centric systems* design. These high-level robust centric systems by definition must provide improved information sharing and collaboration between network elements. Such systems enhance the quality of information awareness, improving sustainability, and mission effectiveness and efficiency. The centric network includes all dynamic information elements and applied so as to maximize the desired decision and action impact. Near instantaneous cooperation between all network forces and elements is of course desired to effectively accomplish each mission. Moreover, in real-world operations, an integrated and distributed network centric operating system (NCOS) is required to manage information. The NCOS is a key element in network centric warfare (NCW). However, we only focus on the generation of an ”optimized” flow of quality information given some of the above objectives and constraints of a static network. Since we show that static MCNDPs can be solved quickly using an MOEA, future NCOSs could include this process in dynamic environments where various information channels become unavailable, change their characteristics, or information priorities are modified.

E. Approaches for Solving these Problems

Researchers have approached the NDP in various ways. Depending on the requirements of the problem, the networks can be modelled as a probabilistic undirected graph, degree constrained spanning tree (DCMST), and a mixed integer linear program (MILP) to name a few.

Dengiz et al [7] and Flores et al [8] solved their NDPs using a probabilistic undirected graph. In this approach, the network is modelled after a graph that consists of nodes, arcs, and the reliability of the links. The model does not take into account the capacity of the links and the models presented in the above papers only looked at fixed costs (total cost of all arcs in the network). Since commodities are not a part of this model, they did not include variable costs (costs to send commodities over the arcs) as part of their objective function.

Several researchers [4, 6, 9] model the NDP using a DCMST. This approach consists of nodes, arcs, and degree constraints for each node. The degree constraint limits the number of arcs that can be attached to each node. The objective function only calculates the fixed cost of the network and commodities are not part of the problem formulation.

Erwin [4] applied the MILP approach. This approach uses matrices to represent the node, arc, and commodity values. The goal is to minimize the total cost of the network, which includes fixed costs, variable costs, and a penalty function for dropping commodities. This method incorporates more network detail than the previously mentioned methods. Given the extra detail, it is capable of producing better solutions. But a disadvantage of using this method is the amount of time it takes to search for all possible solutions. This research extends Erwin’s by applying an MOEA to the MILP and looking at the total flow results and the average number of hops as the two objectives.

F. Complexity of Problem

Some models of network design problems can be implemented using polynomial time algorithms. The classical implementation of the minimum spanning tree problem is an example of a problem model that can be solved in polynomial time [10]. But the multicommodity capacitated network design problem is an NP-complete problem [1, 11]. This means there exists no polynomial deterministic algorithm that can solve a realistically sized problem instance in a reasonable amount of time. For this reason, stochastic approaches are often applied to the problem.

G. Stochastic Approaches

Erwin [4] attempted to use some heuristic methods to solve the MCNDP problem. To do this he simplified the problem into a DCMST, similar to the ones implemented by [6, 9]. Since the DCMST ignores traffic requirements, it must be combined with the MILP approach. The MILP phase of the heuristic starts with a minimal solution generated by the DCMST. Using this method no longer guarantees optimality, but it is able to generate a solution that would be considered a ”good” solution to the decision maker, in a more efficient manner.

Alvarez et al [1] used a greedy randomized adaptive search procedure (GRASP) embedded scatter search to solve the MCNDP. GRASP [12] is a two phase algorithm that is used to solve large optimization problems. The first phase, the construction phase, iteratively generates a feasible solution that is created using an adaptive greedy function and random selection. Once the construction phase completes, it is improved through the use of a local search technique. Scatter search is a population based algorithm that constructs solutions by combining previous solutions. This is similar to some forms of crossover, but uses more bookkeeping in order to create high quality and diverse solutions [1]. Their research objective is to minimize the total cost of the network, but considers dropped commodities as hard constraints.

Zaleta et al [13] used a Tabu search-based algorithm to solve the MCNDP. Tabu search uses long and short-term memory. The short-term memory allows the algorithm to exploit a region in the search space. The long-term search is used to explore the search space in an effort to avoid convergence on a local optimal solution instead of the global optimal solution. Their research solved the same variant of the MCNDP as [1], where dropped commodities are not penalized and considered for a solution but are considered hard constraints and as such, are invalid solutions.

Chou et al [9] applied a genetic algorithm (GA) to the MCNDP. Their research used a DCMST to model the problem. As stated earlier, the DCMST model does not utilize commodities when determining the solution. Their research focuses on the impact that mutation, crossover, and encoding have on the performance of their GA. They use a repair function as well as a penalty function to handle any solutions that fail to meet constraints. Our research differs in that we use commodities and we attempt to generate valid or "near valid" solutions with our initialization process and mutation operator.

Lo et al [6] applied a multiobjective hybrid genetic algorithm (MOHGA) to solve the MCNDP. Their work also uses a DCMST model, so commodities are not part of their equation. Their goal is to minimize the fixed cost of connecting nodes via arcs and to minimize the average delay of each arc. The MOHGA algorithm uses four subpopulations which are generated using a variety of methods [6]. These subpopulations are then mixed to produce the next generation. The algorithm fairs well compared to the vector evaluated genetic algorithm (VEGA) [14] and the single-objective genetic algorithm (SOGA) [15]. Since the SOGA is a single-objective algorithm, equal weighting of each objective was applied in order to come up with a solution. Their research differs from our in that commodities are not part of the problem model.

III. Problem Domain Approach

This section presents our approach to solve the MCNDP problem. Section III-A gives an overview of the variant of the MCNDP. Section III-B presents the mathematical formulation of the problem. Section III-C details the specific objectives associated with the problem. Section III-D discusses the difficulty of the problem.

A. Problem Overview

The specific goal of this research is to design a network given specific network requirements and limitations. These requirements and limitations are passed as input files that describe the network characteristics and the commodity requirements. From this input, the objective is to design a network that optimizes user defined objectives. For this research, the total cost and average number of hops are optimized. The next section presents the mathematical formulation of the problem.

B. Problem Formulation

This research utilizes the MILP problem formulation created by Erwin [4]. This formulation is a modification of the uncapacitated NDP by Ahuja et al [3] (discussed in Section II-B). It is a variant of the MCNDP. This formulation differs in two key respects. First, the Erwin formulation includes degree and interface constraints to accommodate a hybrid network. Secondly, the formulation includes capacity constraints on all the arcs. This is done in an effort to mirror real-world telecommunications networks that have multiple types of links and capacities. The problem formulation, as presented initially by Erwin [4], is as follows:

Let N denote the set of nodes, K the number of commodities, and F the number of interface types.

Let (i,j,f) denote the arc connecting node i to node j by interface type f.

Let A denote the node-incidence matrix where $a_{ijf}=1$ if node i is incident to node j via interface type f, and $a_{ijf}=0$ otherwise.

Let x^k_{ijf} denote the fraction of the required flow of commodity k to be routed from the source s^k to the destination d^k that flows on arc (i,j,f).

Let denote the y_{ijf} variable indicating whether arc (i,j,f) is selected as part of the network topology.

Let v^k_{ijf} denote the per unit cost for commodity k on arc (i,j,f) multiplied by the flow requirement for that commodity.

Let c_{ijf} denote the fixed cost of including arc (i,j,f) in the network.

Let u_{if} denote the number of interfaces of type f at node i.

Let b^k denote the required bandwidth for commodity k.

Let cap_{ijf} denote the capacity of arc (i,j,f).

With these definitions, Equations 6 through 14 describes the problem mathematically.

Minimize

$$\sum_{\{k,(i,j,f):a_{ijf}=1\}} \mathrm{v}^k_{ijf}\mathrm{x}^k_{ijf} + \sum_{\{(i,j,f):a_{ijf}=1\}} \mathrm{c}_{ijf}\mathrm{y}_{ijf} + \sum_k 1000 r^k m^k \tag{6}$$

subject to

$$\sum_{\{j,f:a_{ijf}=1\}} x^k_{ijf} - \sum_{\{j,f:a_{ijf}=1\}} x^k_{jif} = \begin{cases} 1-m^k & \text{if } i=s^k \\ -1+m^k & \text{if } i=d^k \\ 0 & \text{otherwise} \end{cases} \quad \forall i\in N, k=1,\ldots,K \tag{7}$$

$$\sum_k r^k x^k_{ijf} \le cap_{ijf} \quad \forall (i,j,f)\in A \ni a_{ijf}=1 \tag{8}$$

$$\sum_k y_{ijf} \le u_{if} \quad \forall i\in N, f=1,\ldots,F \tag{9}$$

$$x^k_{ijf} \le y_{ijf} \quad \forall (i,j,f)\in A, \ni a_{ijf}=1, k=1,\ldots,K \tag{10}$$

$$y_{ijf} = y_{jif} \quad \forall (i,j,f)\in A \ni a_{ijf}=1 \tag{11}$$

$$x^k_{ijf} \ge 0 \quad \forall (i,j,f)\in A \ni a_{ijf}=1, k=1,\ldots,K \tag{12}$$

$$y_{ijf} \text{is binary} \quad \forall (i,j,f)\in A \ni a_{ijf}=1 \tag{13}$$

$$m^k \text{is binary} \quad \forall k=1,\ldots,K \tag{14}$$

Equation 6 is the objective function for the total cost. The goal is to minimize the total cost, which is an aggregation of the variable costs, the fixed costs, and the penalty function for any commodities that are dropped. There are numerous constraints that must be met as well. Equation 7 is the node balance constraint, which varies slightly from the one presented in Equation 2, since this problem definition takes into account dropped commodities. Equation 2 is for an uncapacitated NDP, and since the arcs had unlimited capacities, there was no need to drop commodities. Equation 7 takes this into account. Constraint 8 ensures that the total amount of commodity flow over an arc doesn't exceed the arcs capacity. Constraint 9 ensures the number of arcs adjacent to a node doesn't exceed the number of interfaces assigned to that node.

Constraint 10 ensures that there is an arc available for each commodity flow. Constraint 11 ensures that the directional arcs allow commodity flow to and from both nodes. Constraint 12 ensures that all flows are non-negative. Constraints 13 and 14 simply state that arcs and commodities are represented as binary variables.

Figure 1 shows a graphical example of two nodes. Note that the arcs between the nodes are directed arcs and each is has its own cost and capacity. Also note that each node can have multiple interfaces. Therefore a commodity can have multiple flow options from a node. To minimize the variable cost, it is important to send the commodity along the lowest cost arc. But given the capacity constraints of each arc, that may not be possible.

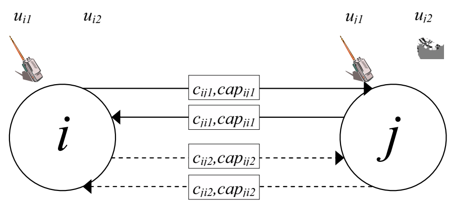

Fig. 1. An arc example for the Network Design Problem [4]

Figure 2 depicts an example of a four-node instance that has two possible interfaces per node. Note that node 3 has two outgoing satellite links while node 4 has zero. All the arcs have the same cost (4) and capacity (4). This does not have to be the case. The arcs can have varying costs and varying capacities, just as the nodes can have varying interfaces.

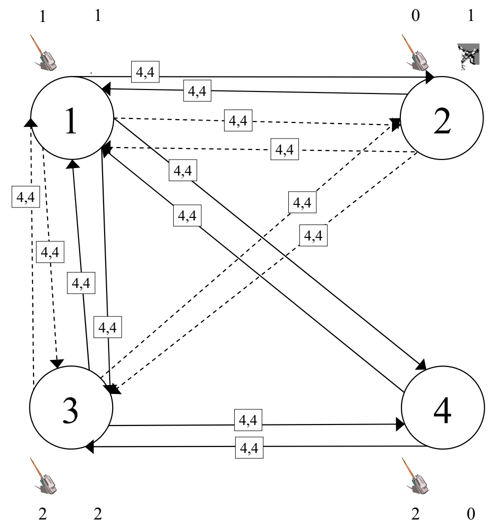

Fig. 2. Four node instance of the Network Design Problem [4]

To go along with the four node example in Figure 2, Table I lists the commodities that are required to flow on the network. The total number of commodities possible commodities is $N(N-1)$, in this instance, 12. Each commodity has a source, destination, and a bandwidth requirement. For source-destination pairs that have no commodity flow between them, the bandwidth required would be zero. This bandwidth, in conjunction with the amount of flow over an arc, is used to calculate the variable cost.

TABLE I
COMMODITY LIST FOR THE FOUR NODE INSTANCE OF THE NDP [4]

Commodity	*Source*	*Destination*	*BW Required*
1	1	2	4
2	1	3	1
3	1	4	2
4	4	1	3
5	2	1	1
6	2	3	4
7	2	4	2
8	4	2	2
9	3	1	5
10	3	2	1
11	3	4	2
12	1	3	3

C. Specific Problem Objectives

Our program calculates the following objectives for each network: total cost, fixed cost, variable cost, average number of hops, and diameter. For this research we examine only two of these objectives:

- Minimize total cost
- Minimize average number of hops

By minimizing the total cost, the goal is to design a network as cheaply as possible. But at the same time, we want to minimize the average number of hops, which aids in the efficiency of the network. Fewer hops for a commodity equate to a faster delivery time.

These objectives were chosen in order to compare with previous results. Other objective can be easily used that help to categorize the network based on efficiency, robustness, or any other network metric.

D. Problem Difficulty

For a relatively small number of nodes, the search space for this problem is quite large. Equations 15 to 17 show the required size of each matrix for the problem, where X_n is the number of possible commodity flows, Y_n lists whether an arc is present between the specific interface of two nodes, S_n lists whether commodities have been dropped, N is the number of nodes, K is the number of commodities, and F is the number of interfaces from each node. Observe $K = N(N-1)$, so as N grows, so does K. This problem grows in a non-linear fashion as the number of nodes increases, making the problem intractable for deterministic algorithms as the number of nodes increase.

$$X_n = N(N-1)KF \tag{15}$$

$$Y_n = N(N-1)F \tag{16}$$

$$S_n = K-1 \tag{17}$$

But the matrix size is only part of the problem. Not only are there a lot of values for the X matrix, but the values represent the percentage of flow for a commodity from one node to another. In order to limit the search space somewhat, we chose to limit the percentages at 20% increments. This limits the search space somewhat, but is still granular enough to allow the network to divide commodities and send them across different arcs. Table II shows the search space size for a four node and 10 node problem, given the 20% increments. Granted, most of the possible values do not meet the given restraints, so a researcher can greatly decrease the effective search space by avoiding the generation of invalid solutions.

TABLE II
SEARCH SPACE SIZE EXAMPLES

	4 node	10 node
Possible X values	1.28×10^{224}	1.12×10^{12606}
Possible Y values	1.68×10^{7}	1.53×10^{54}
Possible S values	2048	6.19×10^{26}
Solution space size	4.40×10^{234}	1.06×10^{12687}

To visualize the search space, we attempted to generate valid solutions using a Monte Carlo approach. We generated $10,000,000$ instances and every one of them failed to meet constraints. We then modified the Monte Carlo search so that our solutions would meet all the node balance constraints. We generated $50,000$ solutions and roughly 80% were valid solutions. Figure 3 shows the results of a modified Monte Carlo run using the second problem instance found in Erwin's initial work [4]. Circled in the figure are the points that create the known Pareto front for the valid solutions generated. The Pareto front is the set of nondominated, valid solutions generated by the algorithm. Figure 3 shows that the algorithm found 15 valid solutions that were not dominated by another solution. Note that some invalid solutions were generated that were better than the valid solutions, but for the most part, valid and invalid solutions are all intermingled in the search space.

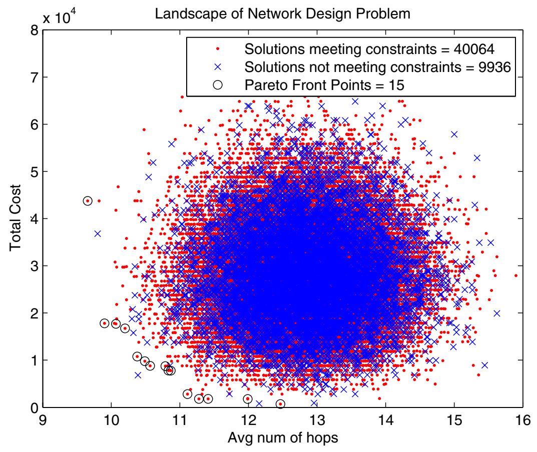

Fig. 3. Modified Monte Carlo results for the 2nd instance of the 10 node problem

IV. ALGORITHM DESIGN

This problem is highly constrained, such that a random initialization process and standard genetic operators almost always generate infeasible networks [7]. To overcome this, a researcher needs to either apply a repair function such as in [16], or ensure that individuals generated by the operators and the initialization process are able to meet the most restrictive constraints. This research takes the latter approach and generates only feasible and near feasible solutions (solutions that break only minor constraints). This section briefly discusses MOEAs in general, provides an overview of the NSGA-II MOEA, and then discusses specific algorithm enhancements.

A. EA and MOEA Overview

Evolutionary algorithms (EA) include genetic algorithms (GA), evolution strategies (ES), genetic programming (GP), and evolutionary programming (EP). EAs consist of a class of algorithms that use the concepts of genetics which enable them to explore the search space. In an evolutionary algorithm, there is a collection of individuals, each one is known as a chromosome. A group of chromosomes are created and compared to one another. This group is known as a population. Chromosomes consist of alleles that can be encoded into a variety of datatypes: binary, integer, real-valued, etc. These allele values are altered by EA operators such as mutation and recombination. Mutation occurs by inserting new genetic material into the population by modifying allele values. Recombination is accomplished by exchanging allele values between two or more individuals of the population. There are many varieties of genetic operators each with a different set of parameters that may be modified given a particular EA type [17]. After the chromosomes are modified, a selection process occurs where a new population is determined for the next generation. There many are ways to select one chromosome over another [18], but the main objective of the selection process is to steer the EA toward the solution. Regardless of the selection process used, all EAs have a fitness function that they assign to a chromosome. The selection process compares the fitness functions of the chromosomes in order to determine the new population.

A multi-objective EA (MOEA) differs from an EA in that there is more than one fitness function for each chromosome. This often creates a situation where there is a vector of answers that can be considered optimal. To determine which answer is best, the researcher must either weight the fitness values or must pick one point out of the known Pareto front via an inspection process. The goal of an MOEA is move the population toward the true Pareto front while maintaining a diverse set of solutions along the known Pareto front. The book by Coello Coello, Lamont, and Van Veldhuizen [19] explains many of the important aspects of MOEAs in more detail.

B. Overview of the NSGA-II Algorithm

The NSGA-II algorithm is an MOEA developed by Deb [20]. The algorithm is an implicit building block (BB) MOEA based on its predecessor, the NSGA algorithm [21]. The

NSGA-II is similar to most MOEAs, in that it utilizes recombination, mutation, and tournament selection operators. The niching operator that it uses is a rank based niching which is based on Pareto dominance and a crowding operator. The algorithm gathers results for all population members and then runs through the entire population to find the individuals that are non-dominated. These individuals are assigned a rank of 0 and are removed from the search population. Then the algorithm finds the next group of non-dominated individuals and gives them a rank of 1 and removes them from the population. The algorithm continues in this fashion until it has sorted all the members based on their Pareto dominance. To determine the next generation, the algorithm pulls individuals with the lowest Pareto ranking into the population. The crowding operator comes into play when the algorithm must pull in a portion of a ranked set of individuals into the population. In this case, the algorithm accepts only the least crowded points in that ranking. A density estimator picks the two nearest points on either side of the point being measured and creates a cuboid. The crowding distance for the point is the average side length of the cuboid.

C. Algorithm Implementation

Since this problem is highly constrained, generating random values to put into the matrices creates invalid solutions. To overcome this situation, the initialization process is done using a propagation mutation operator for each commodity. The operator starts with the first commodity listed in the matrix and ends with the last commodity listed. This introduces some bias into the problem, where the earlier commodities have a higher probability of meeting the arc capacities than the last ones. But this bias can be seen as a priority bias, where the first commodities listed are the highest priority. As such, they are allowed to utilize arc capacity first and lower priority commodities are able to fill any remaining arc capacity.

Figure 4 shows an example of how the propagation mutation algorithm generates the flow of a commodity from the source node to the destination node. The example is for a four-node problem and the source node is 1 and the destination node is 4. To limit confusion, the example assumes only one interface type. In reality, there can be multiple arcs between each node.

In the example, the algorithm initially generates a random commodity flow of 60% from node 1 to node 3. Since the outflow of the commodity from the source is not equal to 100%, the algorithm generates two more random outflows to nodes 2 and 4 until the outflow constraint is met for the source node. The algorithm then takes the last flow generated and determines if the node is the destination node. Since 4 is the destination node, that flow path is complete and the algorithm goes back the previously generated path. Since node 2 is not the destination, the algorithm randomly generates the next node 3 and the outflow from 2 (20%). Since node 2 has the same inflow and outflow (20%), the algorithm progresses down to the next node. The algorithm continues in this fashion until all the flows end up at the destination node. By initializing the commodities in this way, we ensure that the source and destination nodes have 100% inflow and outflow, respectively. We also ensure that the intermediate nodes are balanced. This process greatly reduces the number of individuals not meeting constraints.

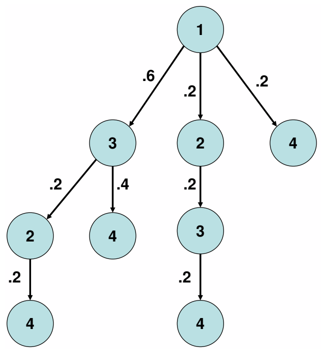

Fig. 4. Example of the initialization of a commodity flowing from a source node (1) to a destination node (4) in a four node problem

Table III lists the probability that a particular flow percentage is generated. Note that the distribution is not uniform. Instead it is heavily biased toward allowing 100% of the commodity to flow from one node to the next. This bias was added to better represent the real world. Typically, it is more efficient for the nodes to send all of a commodity out on one node and it is easier for the destination node to handle all the commodity coming in all at once instead of reassembling it at the end. But link capacities may require that some commodities be divided in order to be passed from one node to the next, so the algorithm must allow for some way to vary the flow percentages of the commodity. But in doing this, the search space can be greatly increased. This research implements an approach that allows commodities to be divided into 20% increments. While this may prevent the algorithm from getting the optimal solution, it greatly reduces the amount of space the algorithm would have to search if each of the commodities could flow in 1% increments.

TABLE III
INITIALIZATION OF FLOW PERCENTAGES

Commodity flow percent	Likelihood of generation
20%	11.1%
40%	11.1%
60%	11.1%
80%	11.1%
100%	55.6%

For the initialization process, the propagation mutation operator is applied on all the commodities. But for the mutation process, random commodities are picked based on the mutation probability. For this research, only mutation is used, so the mutation probability is set to to a relatively high level. Table IV lists some of the key parameters used in this research. 200 individuals and 250 generations are chosen to create a total of $50,000$ fitness evaluations. This allows for a better comparison of how the M-NSGA-II algorithm compares to a Monte Carlo method that generates $50,000$ solutions.

TABLE IV
M-NSGA-II PARAMETERS

Parameter	Value
Number of individuals	200
Number of generations	250
Total evaluations	50,000
Mutation	10.0%
Crossover	0.0%

V. RESULTS AND ANALYSIS

The goal of this research is three-fold:

1) Develop better understanding of the problem domain search space.
2) Develop an MOEA that can generate solutions for the MCNDP quickly and effectively.
3) Generate solutions that are competitive with the work done by Erwin [4].

As shown in Figure 3 and discussed in Section III-D, we first developed a better understanding of the search space by applying several Monte Carlo approaches to the problem. We found that due to restrictive constraints, a purely randomized initialization process failed to generate a single good solution. Either a repair procedure or guided random initialization process was needed. We chose the latter, using the propagation mutation algorithm to generate commodity flows. This process produced a population with 80% of the solutions valid. We then found the Pareto front created by the Monte Carlo process and used it to compare the results of the M-NSGA-II algorithm.

Figure 5 shows a comparison of a single run of the M-NSGA-II with the Monte Carlo method. Figure 6 shows a close-up view of the lower Pareto front points to show that they are nondominated.

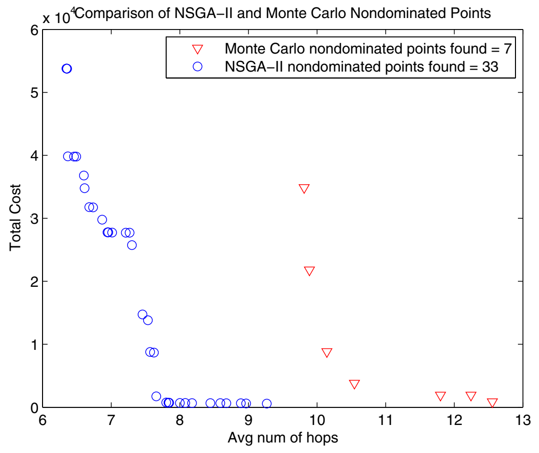

Fig. 5. Comparison of 10 node problem (test10a) Monte Carlo results with M-NSGA-II results

When comparing the results of the Monte Carlo method and M-NSGA-II, there are similarities between the generated Pareto fronts. They both seem to conform to the same shape.

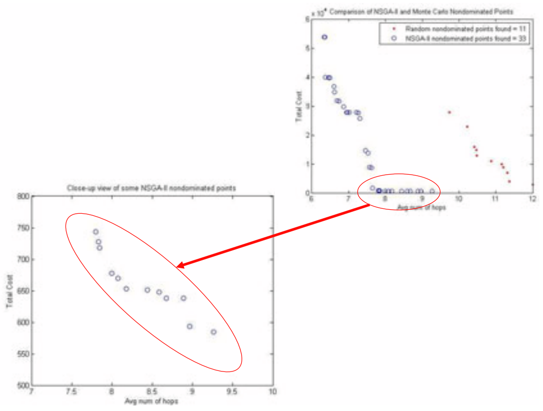

Fig. 6. Close-up view of lower M-NSGA-II nondominated points

It is interesting to note that all of the M-NSGA-II solutions performed better than the Monte Carlo solutions. The Monte Carlo technique appears to be hindered by the initialization process, whereas the M-NSGA-II is able to use mutation and selection to greatly improve its results. Figure 7 shows the initial points generated by the M-NSGA-II algorithm as well as the nondominated points generated from both the M-NSGA-II and Monte Carlo methods. The initial M-NSGA-II values are worst than the Monte Carlo method, which would be expected, since both populations are generated in the same fashion, but the Monte Carlo method generated 50,000 individuals versus 200 for the M-NSGA-II. The propagation mutation operator was then able to generate individuals that had fewer average hops and networks with smaller total costs. But since the mutation operator is a random process, it also generated solutions that were worst than the parents. The selection operator is an elitist operator, so the algorithm propagates all the best solutions along with other good solutions, based on dominance and diversity.

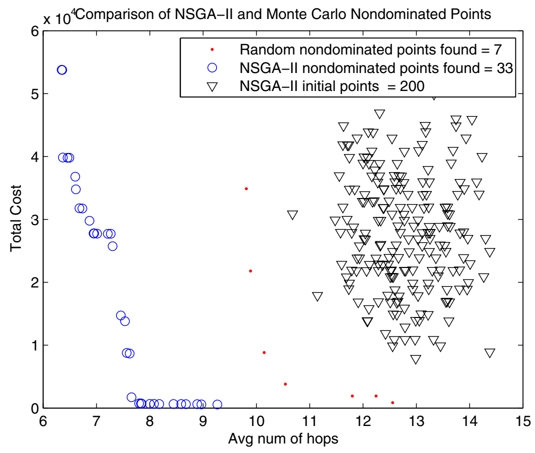

Fig. 7. Where the initial M-NSGA-II points are with respect to the Monte Carlo technique and final M-NSGA-II solutions (trial 1)

The M-NSGA-II was run 30 times for each of the 10 instances of the 10 node MCNDP. Table V compares our total cost results with those found in [4]. Since we are dealing with costs, the lower the number, the better result. Not only did our mean results perform better. But even the worst result from our thirty runs performed better than Erwin's best result. Erwin used Xpress-MP optimization software and he solver the problem using the Newton Barrier method, the Primal Simplex method, and the Dual Simplex method. He used these deterministic approaches that find his best solutions in the 10 node instance. Since our stochastic method outperformed his solved deterministic methods, our results were surprising. After careful review of all objective functions and constraints, we found that our implementation was coded exactly as it was for Erwin. We determined that Erwin's black box implementation was the probable fault. His optimization algorithms do not have any parameters specifying the granularity of his commodity flows. His programmed limitations may have made it impossible for his deterministic algorithms to find the best solutions.

TABLE V

COMPARISON OF TOTAL COST FOR M-NSGA-II AND ERWIN'S RESULTS

	M-NSGA-II (30 runs each trial)			Erwin's [4]
Trial	mean / std dev	best run	worst run	best run
1	**585.19** / 10.63	567.80	620.60	815.9
2	**516.21** / 12.24	493.40	535.00	823.45
3	**519.63** / 13.59	489.20	539.40	763
4	**485.87** / 13.00	444.40	507.00	898
5	**535.27** / 15.20	507.20	579.60	766.8
6	**544.71** / 11.32	521.60	565.00	813.42
7	**527.45** / 17.94	499.60	591.40	736.8
8	**543.73** / 15.48	516.00	575.80	879.65
9	**514.36** / 18.36	484.80	576.40	857.33
10	**510.79** / 12.95	485.40	533.00	888
Mean	**528.32**	500.94	562.32	824.235
Std. Dev.	**29.49**	31.73	33.53	56.3262

VI. CONCLUSIONS

In this research, we discuss some of the ways that researchers model NDPs. The advantages and disadvantages of each model are discussed. We present the MCNDP and discuss some of the research done on this problem. A specific variant of the MCNDP is introduced. We attempt to solve the MCNDP using two objectives - total cost and average number of hops. The search space is analyzed and the NSGA-II MOEA is modified to solve this problem. The results show that the M-NSGA-II is able to improve upon the best results reported in [4]. Future work includes investigating larger real-world problem instances. We also plan to include more objectives functions and analyze the solutions using a larger variety of MOEA evaluation metrics. Our overall objective is to create a tool that can play a vital role in the development of network centric systems. This research is a big step in that direction.

REFERENCES

[1] Ada M. Alvarez, José Luis Gonázlez-Velarde, and Karim De-Alba. Grasp embedded scatter search for the multicommodity capacitated network design problem. *Journal of Heuristics*, 11(3):233–257, 2005.

[2] Roger Guimera, Alex Arenas, Albert Diaz-Guilera, and Francesc Giralt. Dynamical properties of model communication networks. *Physical Review E*, 66:026704, 2002.

[3] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[4] Michael C. Erwin. Combining quality of service and topology control in directional hybrid wireless networks. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH, March 2006.

[5] Kaj Holmberg and Johan Hellstrand. Solving the uncapacitated network design problem by a lagrangean heuristic and branch-and-bound. *Oper. Res.*, 46(2):247–259, 1998.

[6] Chi-Chun Lo and Wei-Hsin Chang. A multiobjective hybrid genetic algorithm for the capacitated multipoint network design problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 30(3):461–470, 2000.

[7] Berna Dengiz, Fulya Altiparmak, and Alice E. Smith. Local search genetic algorithm for optimal design of reliable networks. *IEEE Trans. Evolutionary Computation*, 1(3):179–188, 1997.

[8] Susana Duarte Flores, Benjamín Barán Cegla, and Diana Benítez Cáceres. Telecommunication network design with parallel multi-objective evolutionary algorithms. In *LANC '03: Proceedings of the 2003 IFIP/ACM Latin America conference on Towards a Latin American agenda for network research*, pages 1–11, New York, NY, USA, 2003. ACM Press.

[9] Hsinghua Chou, G. Premkumar, and Chao-Hsien Chu. Genetic algorithms for communications network design - an empirical study of the factors that influence performance. *IEEE Trans. Evolutionary Computation*, 5(3):236–249, 2001.

[10] Corinne Feremans, Martine Labbé, and Gilbert Laporte. Generalized network design problems. *European Journal of Operational Research*, 148:1–13, 2003.

[11] D. S. Johnson, J. K. Lenstra, and A. H. G. Rinnooy Kan. The complexity of the network design problem. *Networks*, 8:279–285, 1978.

[12] Thomas A. Feo and Mauricio G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

[13] Nadia Cobos Zaleta and Ada Margarita Alvarez Socarrás. Tabu search-based algorithm for capacitated multicommodity network design problem. In *CONIELECOMP*, pages 144–148. IEEE Computer Society, 2004.

[14] J. David Schaffer. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.

[15] Hisao Ishibuchi and Tadahiko Murata. Multi-Objective Genetic Local Search Algorithm. In Toshio Fukuda and Takeshi Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 119–124, Nagoya, Japan, 1996. IEEE.

[16] Jesse Zydallis. *Explicit Building-Block Multiobjective Genetic Algorithms: Theory, Analysis, and Development*. PhD thesis, Air Force Institute of Technology, Wright Patterson AFB, OH, March 2003.

[17] Thomas A. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York - Oxford, 1996.

[18] T. Bäck, D. B. Fogel, and Z. Michalewicz, editors. *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, Bristol, 2000. Contains excerpts from the Handbook of Evolutionary Computation.

[19] Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.

[20] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[21] N. Srinivas and Kalyanmoy Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, Fall 1994.

[22] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.