2009-01-01

Multi-Objective Network Reliability Optimization Using Evolutionary Algorithms

Franciso Oswaldo Aguirre
University of Texas at El Paso, faguirre@miners.utep.edu

MULTI-OBJECTIVE NETWORK RELIABILITY OPTIMIZATION USING EVOLUTIONARY ALGORITHMS

Oswaldo Aguirre

Department of Industrial Engineering

APPROVED:

_____
Heidi Taboada Ph.D., Chair

_____
Virgilio Gonzalez, Ph.D.

_____
Jose F. Espiritu, Ph.D.

_____

Patricia D. Witherspoon Ph. D.
Dean of the Graduate School

For my family, and friends, for their friendship and support

MULTI-OBJECTIVE NETWORK RELIABILITY OPTIMIZATION USING EVOLUTIONARY ALGORITHMS

By

Oswaldo Aguirre Ortega

THESIS

Presented to the Faculty of Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Industrial Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2009

ACKNOWLEDMENTS

Thanks to my parents to awake in me the inquietude to start this journey and give the support to complete it.

When I started this adventure, I never imagined what will happened or what I could feel when I finish it. I had considered a Master's degree just as any other curse to be taken. However, since the beginning of this journey, I realized that it will be many things but just another curse. At the beginning, I was alone in a new country with no friends, but that changed quickly. There were many people that gave me support, advices and guidance in this project. A very important person that makes this work possible is my advisor Dr Heidi Taboada. She always had the correct words at the correct moment to make this work a reality

Study away of your hometown always implies that you need to leave friends and family. This situation is not only hard to the person who leaves but also to the people that stay. I am grateful to my parents that always supported me even f that meant to be away of them. I want to thank also my friends because when I needed the more their support they always were there for me with their words of encouragement giving me the strength that I needed to finish this work.

Now that I am at the end of this work I remember all the help that I received for my professors. That dedicate their lives to teach others, and my friends and classmates of the ergo-Lab that always were there when a needed them. Last, but not least, thanks to god for showing me that with hard work and faith everything is possible.

ABSTRACT

This work presents a new multiple objective evolutionary algorithm to solve three well known network reliability allocation problems considering different conflicting objectives to be optimized simultaneously. The new algorithm is applied in the design of a telecommunication network that is formed for several stations or nodes interconnected by telecommunication links or paths. The problem presented in this work involves finding which links to activate in order to obtain connectivity in the nodes. The number of nodes that need to be connected depends of the case that is being evaluated. The three network reliability problems considered are: all-terminal, *k*-terminal, and two-terminal. Network reliability evaluation represents an NP-hard problem. In literature two approaches have been presented: exact reliability calculation and reliability estimation. Because of the impracticality of exact calculation in networks of moderate to large size Monte Carlo simulation is used to estimate network reliability. The new algorithm presented in this work is based on evolutionary computation and the objective functions considered are the maximization of system reliability, the minimization of system cost and, the minimization of system weight. The solution to this multiple objective problem is a set of Pareto solutions.

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: Introduction

Network theory is used to model different types of problems in many fields and solve a variety of problems such as the telecommunication network design problem. In the information era, being connected to the world, in order to share information, increase productivity, and make business around the world, is a very important issue that requires to develop a strategy that help to design a robust network. The objective of this thesis is to develop a flexible and efficient multiple objective evolutionary algorithm to design a reliable and cost efficient telecommunication network.

1.1 Telecommunication networks:

A telecommunication network is a type of network that is used to transmit voice or data to one part of the network to another. Nodes represent terminals in the network such as computers, servers, routers, switches, etc. Communication paths or links are represented by wires, radio frequencies, wireless signals, etc. Some examples of telecommunication networks are computer networks and the Internet network. The first electrical communications network was the telegraph. Here the network consisted of telegraph operators who transmitted the message efficiently using Morse code and routed the message so that it took the shortest possible path to its destination while taking into account internal network failures. Modern telecommunication networks normally involve the use of computational systems as an infrastructure of the network. However, no matter what kind of telecommunication network is implemented or how new is the technology implemented, the main idea behind the design of a

telecommunication network remains the same. Build a network as reliable as possible at the minimum cost and in some cases considering additional objective functions.

As mentioned before, there are several examples and types of telecommunication networks. However, this thesis focuses in wireless telecommunication networks although the algorithm developed can be applied to a wired network. Wireless telecommunication systems have been becoming popular recently. They allow people to communicate everywhere in the world. Wada & Fukushima [47] presented a research that combined the features of Internet, the largest world-wide communication platform, and digital wireless systems considering on mobile terminals with not only has global wireless connectivity but also IP reachability. Wada & Fukushima's work focuses more in the computational technical part of a wireless network than in the topology of the network. A similar approach is presented by Hayashi & Abe [48] where they proposed a reliability model for representing telecommunication networks that does not focus on topological information, but rather on traffic path information.

The reliability network allocation problem has been addressed in literature using different approaches. However, one aspect remains the same cost is always present along to reliability where a telecommunication network is designed The problems that are being presented in this research pertain to the cases in which (*i*) all the network nodes have to be connected, (*ii*) only some of the nodes have to be connected, and (*iii*) only two nodes need to be connected for the system to function.

In several previous approaches, the network reliability problem has been solved considering only one single objective function to be optimized subject to constraints. For instance, Ramirez-Marquez *et al.* [33] solved the problem using a probabilistic approach

to maximize the reliability subject to a cost constraint. Dengiz *et al.* [34] proposed a Genetic Algorithm (GA) and later a neural network [30] to solve the all-terminal reliability problem, in both cases as a single objective problem. Dengiz [50] developed a hybrid ant colony approach to design a telecommunication network to obtain the optimal network configuration with the least cost achieving certain level of reliability.

Considering the problem as a multiple objective optimization model, Azaron *et al.* [51] presented a GA to solve the time-cost trade-off problem in PERT networks and Ho *et al.* [52] proposed a method using an integrated multiple criteria decision making approach to solve a distribution network design. However there is not too many studies that consider the telecommunication network design as a multiple objective optimization problem.

1.2 Multiple objective optimization.

The existence of multiple criteria objective problems is very common in engineering. Almost all the optimization problems in real life involve more than one objective to be optimized, and normally those objectives are in conflict to each other. This situation is very easy to observe in real life. For instance, a very high performance product is also a high cost product, and a customer always seeks a product with high performance, but at the lowest cost. This dilemma is also found in engineering problems. Hence, the development of a method to find a solution to multiple objective optimization problems has been the topic of many research papers and books.

There are several approaches to deal with multiple objective optimization problems such as goal programming, the lexicographic method, the weighted sum method and utility functions. The mentioned methods are mathematical approaches that have been

proposed to deal with multi- criteria decision making problems. These methods are easy to implement. However, they not always really optimize all the objectives simultaneously. The other general approach is to use metaheuristic methods. These methods mimic several natural behaviors such as human brain, evolution, animal behavior, memory, etc. Meta-heuristics methods approximate solutions to the true Pareto front. However, they cannot guarantee to obtain a global optimal, but it has been proven that they get good solutions in very complex problems. Some of the most well known heuristic methods are: Tabu-search, neural networks, particle swarm, ant colony, genetic algorithms, evolutionary algorithms, etc

Generally, there are two primary approaches for the solution of multi-objective problems. The first approach involves the aggregation of the attributes into some kind of overall composite objective function; while the second approach involves populating a number of feasible solutions along the Pareto frontier, and the final solution is a set of non-dominated solutions. The solution to multiple objective problems using evolutionary algorithms is a set of Pareto optimal solutions based in the Pareto dominance concept. Then, a solution is said to be *Pareto-optimal* if it is not dominated by any other solution in the solution space. Thus, the Pareto-optimal solutions to a multi-objective optimization problem form the Pareto front or Pareto-optimal set [54].

Among all the meta-heuristic methods, evolutionary algorithms have been applied to solve many variety of prblems. Evolutionaty algortihms and genetic algorithms are based in Dawin's evolutionary theory. Basically, an evolutionary algorithm is a population based seach method in which a population of solutions to an optimization problem evolve toward better solutions. The evolution usually starts from a population of randomly generated individuals or posible solutions. At each generation, the fitness of

every solution in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. There are a specia kind of evolutionary algorithms that are applied to multple objective problems and are called muultiple objective evlolutionary algorithms(MOEAs) .

Some of the recent developed multi-objective evolutionary algorithms are: SPEA (strength Pareto evolutionary algorithm) [53]; PAES (Pareto-Archived Evolutionary Strategy) [55] and NSGA-II [56], among others. Unfortunately, these MOEAs cannot be directly applied to solve the network reliability design problem. Moreover, being general algorithms, imply that they do not have the strength of a problem-oriented technique. Thus, as part of this research a new MOEA was develeoped in order to solve the telecommunication design problem.

1.3 Thesis objective.

It is important to explain the problem that is being solved in this thesis. A telecommunication network is formed by nodes interconnected by telecommunication paths. Three different cases are considered: all-terminal, *k*-terminal, and two-terminal. A solution for the problem is a network configuration or network vector that consists in a list of the links that have to be active in order to maximize the reliability of the network and minimize cost and weight. A method to solve this problem is to evaluate all the posible solutions that the problem has and selet the ones that are nondominated. However, due to the large amount of possible solutions and the nature of the problem doing this in a large network is totally imparactical and infeasible. In order to solve the telecomunication network design problem a new MOEA was developed.

Therefore, the objective of the present work is to develop a new multiple objective evolutionary algortithm to solve the all-terminal, *k*-terminal, and two-terminal network reliability problem considering the following objectives to be optimized simultaneosly

- Maximize reliability
- Minimize total network cost
- Minimize weight

The present thesis is divided in 6 sections. Section 2 presents a literature review about the most common methods used to solve multiple objective optimization problems. In this section, also some basic concepts about single and multiple optimization are discussed.

Section 3 presents the different kinds of networks that can be used in engineering to model different problems. Section 3 also introduces the telecommunication network design problem. There are many type of telecommunication networks and chapter 3 also give a brief description of the most used in the field. Reliability is always an objective when working with network optimization problems, and in this chapter, the most common methods to evaluate network reliability are clearly described.

Chapter 4 presents the new multiple objective evolutionary algorithm developed to solve the telecommunication network design problem for the three cases mentioned before, considering three objectives to be optimized simultaneously. These objectives are: maximize network reliability, minimize network cost, and minimize weight. The algorithm can roughly divide in four main stages: initialization, evaluation, selection, and reproduction. Each part of the algorithm is explained in detail in section 4.

Section 5 presents two different numerical examples for the three network reliability cases where the developed algorithm was applied. Each example is explained in detail and the results obtained are presented and discussed in chapter 5.

Finally chapter 6 presents some conclusions and discusses some important points that can be considering as part of future research.

CHAPTER 2: Multi objective Optimization

Real world problems inevitably involve multiple objective optimization. Multiple objective optimization requires the simultaneous optimization of all objective functions. However, many of the objectives are usually conflicting with each other. Selection of one solution over others may be a complicated decision. Empirically, those decisions are made on the basis of intuition, common scene, experience, and a combination of all of these. However, in several areas as engineering, implementation of a methodology for multiple objective optimization is needed. These kinds of problems that involve several objectives or goals to be optimized simultaneously are known as multi-objective optimization problems or multi-criteria decision making problems (MCDM) [1].

In literature, several methodologies have been developed to address optimization problems that involve multiple conflictive objectives. These methods can be divided in two big groups: Mathematical methods and meta-heuristic methods. Mathematical methods basically transform the problem in to a single objective problem and in the majority of the cases linear programming is implemented to solve the problem. When using metaheuristic methods an optimal solution is not guarantee, however they have proved to obtain very good approximation to a global optimal in very complex problems when mathematical methods cannot be implemented.

The main characteristic when using metaheuristic methods for multiple objective optimization is that not unique solution exists. Instead, a set of mathematically equal good solutions is obtained. These selected solutions are known as nondominated solutions or Pareto optimal solutions based on the concept of Pareto optimality. Pareto optimality is a term for describing solutions for multiple objectives, when no part of a

Pareto optimal solution can be improved without making some other part worse. For instance Figure 1 shows an example of Pareto optimality and shows that more than one solution can be obtained.

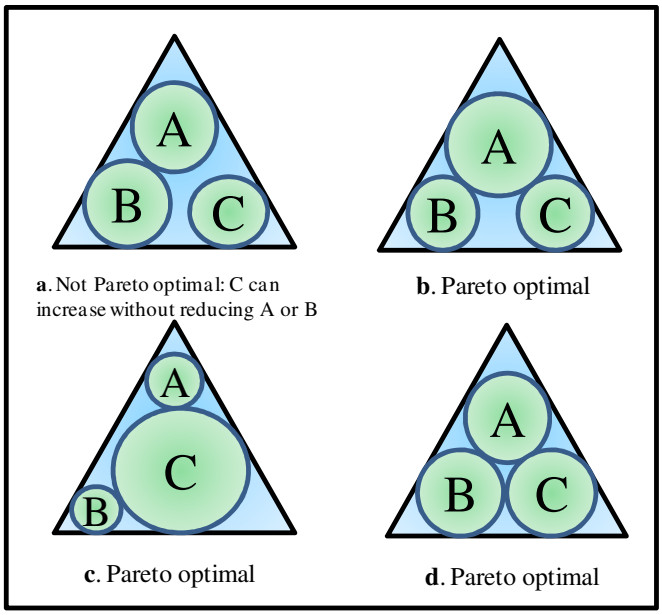

Figure 1: Pareto Optimality

The mathematical model for a multi-objective optimization problem is presented in Equation 1.

$$Min\ , Max\ \ (f_1(x), f_2(x), .., f_n(x)) \tag{1}$$

$$x\ \in\ D\ \subseteq\ X$$

$$subject\ to\ k\ constraints$$

$$h_i(x)\ =\ 0\quad (equality\ constraints)\qquad i\ =\ 1, .., r$$

$$g_i(x)\ \leq\ 0\quad (inequality\ constraints)\quad i\ =\ r+1, .., k$$

Where:

$n \geq 2$, number of objectives

D, feasible region of solutions

X, decision variable space

$f_1(x), f_2(x), .., f_n(x)$, represent the objectives functions that need to be optimized. These objectives are often in conflict with each other.

As mentioned before the general approaches to solve multiple objective optimization problems (MOOP) can be divided as follows:

- Mathematical methods
    - Goal programming
    - Weighted sum method
    - Lexicographic method
    - Utility functions
- Meta-heuristic methods
    - Tabu-search
    - Simulated annealing
    - Evolutionary algorithms
    - Ant colony
    - Swarm intelligence

2.1 Mathematical methods

2.1.1 Goal Programming

Goal programming (GP) was first used by Charnes, Cooper and Ferguson in 1955. It is based in linear programming optimization and it is one of the first methods developed specifically to solve MOOP. At first, it was thought to solve multi-objective linear problems (MOLP). In GP, aspiration levels or also called contribution coefficients are assigned for each objective that is to be optimized, and then the deviations from these aspiration levels are minimized. In other words, in the GP model the objective function is the summation of all the contribution coefficients and the objectives are constrains of

the problem. The mathematical formulation for a general GP problem is presented in Equation 2.

$$Z = \sum_{i=1}^{n} c_j x_j \tag{2}$$

$$s.t.$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq b_i, \qquad for\ i = 1, \dots, m$$

$$x_j \geq 0, \qquad for\ j = 1, \dots, n$$

Where x_1, $x_{2,\dots}$, x_n are the decision variables, and c_1, $c_{2,\dots}$, c_n are the contribution coefficients that represent the contribution to Z for each objective function. In GP a MOOP is converted in to a single objective problem seeking to minimize the Z function. In the constrains, $a_{i,j}$ are coefficients that represents the per unit usage by x_i of the right-hand side coefficient of b_j.

For this linear programming (LP) model some assumptions are considered:

- Each unit for each decision variable x_i contributes c_i units to the objective function and a_{ij} in the constraint.
- The contribution to the objective function and the coefficient in the constraints are independent of the values of x_i
- All parameters a_{ij}, b_j, and c_i must to be known

Summarizing, what the GP method do, is that it basically transforms a MOOP in to a single objective problem minimizing the deviations from each objective function to its optimal value (aspiration levels) and penalizing this distance. This deviation or distance can be positive or negative considering that the GP model is defined by Equation 3

$$Minimize\ Z = \sum_{i \in m}(d_i^+ + d_i^-) \quad (3)$$

$$s.t.$$

$$\sum_{j=}^{n} a_{ij}x_j - d_i^+ + d_i^- = b_i, for\ i = 1, \dots, m$$

$$d_i^+, d_i^-, x_j \geq 0, for\ j = 1, \dots, n$$

Where:

d_i^+ = Positive deviation variable

d_i^- = Negative deviation variable

This method represents one of the most used methods to solve MOOP due to its simplicity of implementation. However, it does not optimize all the objectives at the same time and in some complex problems a feasible solution cannot be obtained.

2.1.2. Weighted sum method

This method is based in the previous one with the significant difference that each objective function has a different weight or importance for the decision maker (DM). The general model for the weighted sum method is presented in Equation 4

$$Z = \sum_{i=1}^{n} w_j(c_j x_j) \quad (4)$$

$$s.t.$$

$$\sum_{j=1}^{n} a_{ij}x_j \geq b_i, \qquad for\ i = 1, \dots, m$$

$$x_j \geq 0, \qquad for\ j = 1, \dots, n$$

Where w_i represents the weight assigned for each objective.

In this method each objective of the optimization problem is more important than others. Therefore, the decision maker assigns weight or importance for each objective

of the problem. These weights are added in the objective function to be optimized (*Z*). For instance, consider a problem with 3 objectives to optimize and the weights for these objectives are 0.80, 0.10, and 0.10, respectively. Equation 5 shows the general model for this problem.

$$Minimize\ Z = .80(d_1^+ + d_1^-) + .10d_2^- + .10d_3^- \quad (5)$$

$$s.t.$$

$$\begin{aligned} ax_1 + ax_2 - d_1^+ + d_1^- &= b_1 \\ x_1 + d_2^- &\leq b_2 \\ x_2 + d_3^- &\leq b_3 \\ x_1, x_2, d_i^+, d_i^- &\geq 0 \end{aligned}$$

As it can be inferred by equation 5, the model is almost the same for the normal GP model the differences are presented in the objective function. This methodology is very easy to implement for any problem that can be modeled as a GP model and have the plus that some weight of importance can be added for each objective. Due to weights exits in many real world optimization problems (some objectives are more important than others), this method has been used in several areas of optimization. However, the weighted sum method has the same disadvantages that GP has.

2.1.3 Lexicographic method.

This method, as the weighted sum method, assigns importance to the objectives to optimize. However, in the lexicographic ordering (Fishburn,1974) the more important objective is infinitely more important than the less important objective.

At first, the DM assigns importance to the objectives $f_1(x), f_2(x), \dots, f_n\ (x)$.The most important objective is optimized without considering the other objectives (6)

$$\min f_1(x) \tag{6}$$

$$s.t.\ x \epsilon X$$

Then the second objective function is optimized subject to the original constraints and to a new one that guarantees the optimality of the first objective function. The procedure continues until the last objective function is optimized.

This method is very easy to implement, but choosing the order or importance of the objectives can be a difficult task. Other disadvantage is that the method is very rough and often the process stops before less important objective functions are optimized.

In order to make the method more versatile or more suitable to practical problems some modifications have been proposed to the lexicographic method. δ-lexicographic method tries to overcome some of the drawbacks of the lexicographic method allowing small increments of the first objective to be traded off with the decrement of the second objective. Even though the lexicographic method has several disadvantages and do not optimize all the adjectives simultaneously, any lexicographic solution is efficient and can be proven to be Pareto optimal.

2.1.4 Utility theory.

Utility function, also called value function is a mathematical expression that assigns a value to all possible choices. The utility function is a very good method to solve multi criteria problems when an explicit mathematical formulation for the value function is known. The problem can be defined by Equation 7

$$maximize\ v\big(f(x)\big) \tag{7}$$

$$subject\ to\ x \epsilon X$$

Since the value function provides a complete ordering in the search space, a best Pareto optimal solution is found with this method. Unfortunately, the real problem is to obtain this value function. Getting this value function can be very difficult, if not impossible, in several multi-objective optimization problems such as network reliability optimization problems. And even if the function is known, it can be very difficult to optimize because of its complicated nature.

However, if the value function can be defined, this method becomes very effective and useful and can be applied to any kind of multiple criteria optimization problems [5].

All the methods mentioned before are mathematical methods that can be applied to multiple objective optimization problems. However, they present important drawbacks that make them impractical to be applied in complex problems such as in network optimization problems. The second general approach involves the use of metaheuristic methods. Instead of obtaining a global optimal solution such as with the use mathematical methods, meta-heuristic methods obtain an estimated solution. These methods have proved to perform well in a variety of complex combinatorial problems getting good approximations to the true Pareto front with relatively low computational effort. These methods are explained in next section.

2.2 Meta-heuristic methods

2.2.1 Tabu-search (TS)

Tabu Search is a meta-heuristic search method presented first by Glover in 1986. It searches for the optimal solution using memory systems which exploits its past history and leads to good solutions [4].

Tabu search (TS) permits the incorporation of procedures to search the solution space economically and effectively. Its memory system is one of the most important aspects of the TS method and the main difference with other meta-heuristic methods that are memory less such as genetic and annealing approaches. While other exploration methods keep in memory the objective value $f(x)$ of the best solution visited so far, TS also keeps the information on the itinerary through the last solution visited. This information will be used to move from one solution to another. Memory can be used to identify elements that are common to good solutions or to paths that lead to such solutions. The flexibility of these memory structures allows the search to perform well in a multi-objective environment [10].

Tabu search begins in the same way as an ordinary local or neighborhood search procedure, proceeding iteratively from one solution to another until a chosen termination criterion is satisfied. For instance consider a simple optimization problem presented in Equation 8

$$\begin{gathered} minimize\ , maximize\ f(x) \\ subject\ to\ x\ \epsilon\ X \end{gathered} \tag{8}$$

Figure 2 presents a pseudo-code with the more important steps for the TS method

1. Chose $x \in X$
2. Find $x' \in N(x)$ such that $f(x') < f(x)$
3. If no such x' can be found, x is the local optimum and the method stops
4. Otherwise, designate x' to be the next new x and return to step 2

Figure 2: TS structure

TS has been implemented in many optimization problems in different fields and has proved to be a good meta-heuristic algorithm. Some of the more important features of TS are: easy implementation for almost any optimization problem and that can be easily modified to enhance the algorithm performance.

2.2.2 Simulated annealing

This algorithm was proposed by Metropolis *et al.*(1987) as an algorithm that simulates the evolution of a solid to thermal equilibrium [6]. The name come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. By analogy with this physical process, each step of the simulated annealing (SA) algorithm replaces the current solution by a random "nearby" solution.

Given a neighborhood structure, simulated annealing continuously attempts to transform the current configuration into one of its neighbors. The acceptance of this neighbor is given by a probability presented in Equation 9

$$p = exp\,(-\delta f(x) / T) \tag{9}$$

Where δ represents the increase in $f(x)$, and T is an important control parameter in simulated annealing which by analogy represents temperature. The structure of the algorithm is presented in Figure 3

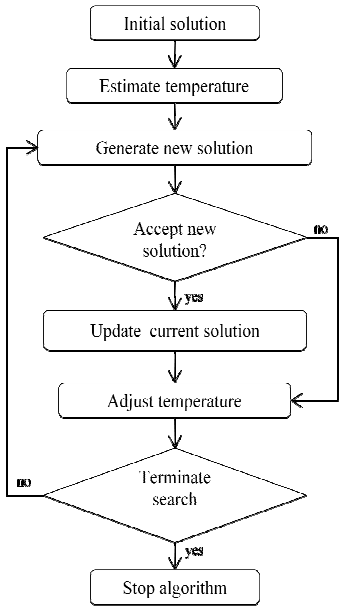

Figure 3: Simulated Annealing flow chart

Some of the advantages of the method are presented below:

- Performs well with nonlinear models, chaotic and noisy data and many constraints.
- Flexibility and ability to approach global optimal solution and escape for a local optimal.
- Easy implementation

SA is not a perfect optimization method some of its disadvantages are stated next:

- Since SA is a meta-heuristic, a lot of choices are required to turn it into an actual algorithm and an optimal global is not guarantee.
- There is a clear tradeoff between the quality of the solutions and the time required to compute them due to the seed is basically random.
- The tailoring work required to account for different classes of constraints and to fine-tune the parameters of the algorithm can be rather delicate.

2.2.3 Ant colony optimization

Ant colony optimization (ACO) is a meta-heuristic search method based in swarm intelligence that takes inspiration from the behavior of some ant species. When ants are looking for food, these ants deposit pheromone on the ground in order to mark some favorable path that should be followed by other members of the colony. ACO mimic the ants' behavior to solve optimization problems

In ACO, a number of artificial ants build solutions to the considered optimization problem. The original ant colony optimization algorithm is known as Ant System and was proposed in Dorigo, *et, al* [9]. The main underlying idea, loosely inspired by the behavior of real ants, is that of a parallel search over several constructive computational threads based on local problem data and on a dynamic memory structure containing information on the quality of a previously obtained result. The collective behavior emerging from the interaction of the different search threads has proved effective in solving optimization problems.

The behavior of artificial ants is inspired from real ants: they lay pheromone on components (edges and/or vertices) of the graph and they choose their path with respect to probabilities that depend on pheromone trails that have been previously laid by the colony. These pheromone trails progressively decrease by evaporation. Intuitively, this indirect communication gives information about the quality of path components in order to attract ants, in the following iterations, towards the corresponding areas of the search space.

Artificial ants also have some extra-features that do not have real ants. They have memory of their previous actions, and they can use that data to improve the

quality of computed paths. In many cases, pheromone is updated only after having constructed a complete path and the amount of pheromone deposited is usually a function of the quality of the complete path.

Figure 4 shows how ants choose the better path considering the pheromone trail that is the main concept for ACO.

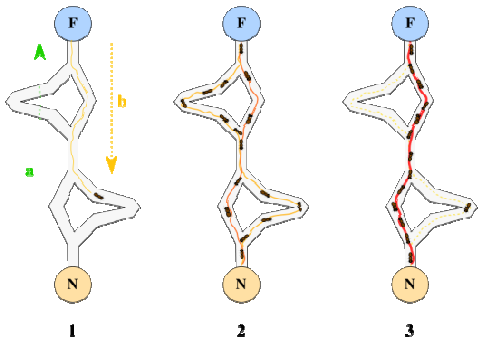

Figure 4: Ant colony concept

The process showed in figure 9 is descried next.

1. The first ant finds the food source (*F*), then returns to the nest (*N*), leaving behind a trail pheromone (*b*)

2. Ants indiscriminately follow four possible ways, but the strengthening of the runway makes it more attractive as the shortest route.

3. Ants take the shortest path, long portions of other ways lose their trail pheromones.

ACO was first applied to the traveling salesman problem searching the shortest path. Recently the method has been applied to a variety of engineering problems. And also some methods have been applied to solve MOOP.

2.2.4 Evolutionary algorithms

Evolutionary algorithms (EA) are based in evolutionary computation. The principle of these methods is to mimic how biological evolution works. It was first presented by J. H. Holland. EAs are population-based meta-heuristic optimization algorithms and there are several kinds of these methods implemented for specific situation but all of them follow the structure presented in Figure 5

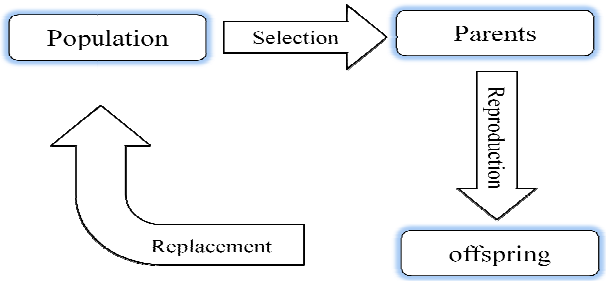

Figure 5: EA Structure

As previously mentioned, EAs are based in evolution to find Pareto optimal solutions for a MOOP. In the first stage, a first population of possible solutions is generated normally randomly. The next step is selection. In this stage, the strongest individuals are chosen to be parents of the next generation. The chosen parents undergo reproduction and a new population is generated. The algorithm continues until a stopping criterion is met. It can be observes in figure 3 that an EA is divided in four main stages: initialization, evaluation, selection, and reproduction. The success of an EA, searching for good solutions, depends in how these stages are performed in the algorithm

EAs have proved to perform well in several kind of MOOP. One of the most important advantages is the flexibility that EAs have to escape from a local optimal to a global optimal exploring more efficient the search space. There are plenty of EAs

proposed in several fields to solve different kinds of problems and not only for MOOP. In this work, the EAs used in optimization areas will be considered.

As mentioned before, all the EAs share some similarities. For example all of them have selection and reproduction. What makes an EA different to another type of algorithm is the strategies to use for each stage of the algorithm. The next section explain these evolutionary strategies used in initialization, selection and crossover.

2.2.4.1 Evolutionary strategies or operators

2.2.4.1.1 Encoded strategies.

The first step in an EA is the initialization or generation of the first population. The first population is formed for several proposed solutions. These solutions are called individuals or chromosomes. A chromosome can be defined as a string of components that have a meaning or correlation to a real problem. Each component of the chromosome is called a gene and contains important information of the generated solution. Some of the most common encoded strategies in EAs are presented below:

- Binary encoding: it is one of the most used due to its simplicity of implementation. In binary encoding, a gene can take the values of 0 or 1. These values usually represent that a component is active or disconnected. An example of this this type of encoding is presented in Figure 6

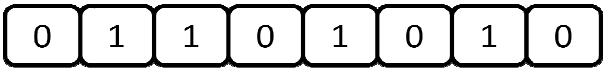

Figure 6: Binary Encoding

- Permutation encoding: this encoding strategy allows integer numbers in the chromosome. These values usually represent a position in a sequence or a number

of components in certain position. This type of encoding normally does not allow repeated values in the string or chromosome. This encoding strategy is presented in Figure 7

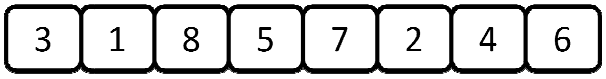

Figure 7: Integer Encoding

- Value encoding: this type of encoding is more difficult to implement in a EA. Value coding allows any type of numbers: integers and real numbers and also letters. An example is in Figure 8

Figure 8: Value Encoding

2.2.4.1.2 Selection strategies.

Selection is usually done on the basis of a fitness function value. The strongest solutions are this with the highest fitness value. A discussion of how a fitness function is obtained is presented further on as a part of this work. Some of the well-known selection strategies used in EAs are:

- Roulette wheel selection: this method works as a roulette does. All the solutions are grouped corresponding to its fitness value. A solution that has a better fitness value has more probability to be selected.
- Tournament selection: two solutions are chosen randomly and the solution with the higher fitness is selected.
- Rank selection: all the solutions are ranked from higher to lower fitness and the solutions that are better ranked are selected.

Selection is a very important step in EAs. However how recombination or offspring are generated is equally important. Some strategies used are presented in the next section

2.2.4.1.3 Crossover strategies.

Crossover represents how children will be generated from the selected parents.

- Single point crossover: this is the most common crossover operator. Its behavior is presented in Figure 9. The point that states where the parent is divided is chosen randomly

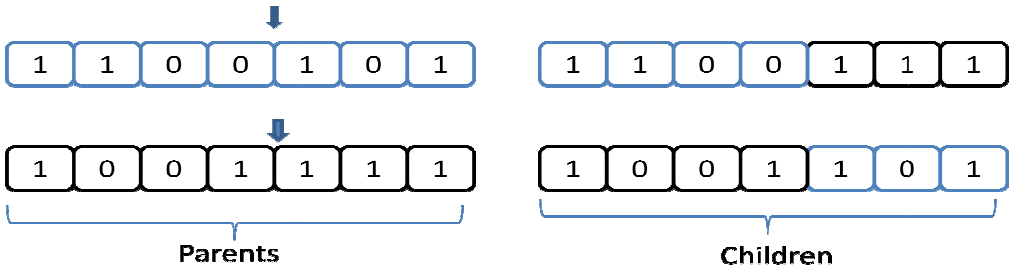

Figure 9: Single Point Crossover

- Double point crossover: this strategy works in similar way than the previous one, but instead of selecting one point, two points are selected randomly. Figure 10 presents this strategy

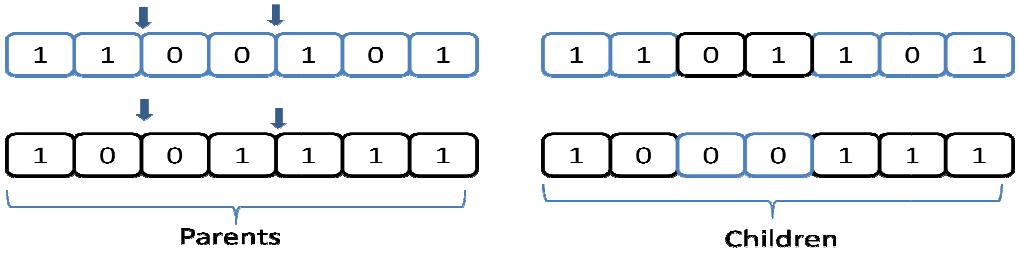

Figure 10: Double Point Crossover

These strategies are the most common used in EAs but it doesn't mean that are the more efficient or the only ones.

The main purpose of a crossover is to generate better individuals getting the best part of each parent. To achieve that, several crossover operators have been proposed recently. Some of these researches can be seen in [7, 8]. The main concept behind the improvement of a crossover operator is trying to distinguish between bad genes and good genes. Gao *et, al* [8] used hill-climbing search to find good blocks in a chromosome. Other similar approach is presented in [7] using the rough set theory to identify good genes in a specific individual. Another important aspect of a crossover operator is the ability to create diversity of children in order to explore efficiently the search space. Therefore, it is important to select a crossover operator that can create good offspring and a good diversity for the network reliability problem

At this point some of the most important operator in EAs have been mentioned and explained. To solve multi-objective optimization several EAs have been proposed in literature. These algorithms are called multi-objective optimization evolutionary algorithms (MOEAs) and are explained in the next section

2.2.3.2. MOEAs

Over time several MOEAs have been proposed in the literature to solve different kind of problems. Some of the more common MOEAs reviewed in literature are:

- Strength Pareto Evolutionary Algorithm (SPEA).
- Strength Pareto Evolutionary Algorithm (SPEA 2).
- Pareto Archived Evolution Strategy (PAES).
- Non-dominated Sorting Genetic Algorithm (NSGA).
- Non-dominated Sorting Genetic Algorithm (NSGA II).
- Adaptive Pareto Algorithm (APA).

All of the previous mentioned algorithms are MOEAs and all of them use evaluation, selection, and reproduction to obtain a new generation. The differences are how they perform fitness evaluation and reproduction. Some of them used mutation as a reproduction procedure, others combine elitism and crossover to generate a new population, and others use tournament selection to select the best children for the new population. All of those algorithms have advantages and disadvantages. A brief description of those algorithms is presented in the next section.

- Strength Pareto Evolutionary Algorithm (SPEA): It was developed by Zitzler and Thiele. The algorithm maintains an external population at every generation storing all nondominated solutions obtained so far. At each generation, both populations, current and external are mixed. All no dominated solutions in the mixed population are assigned fitness based on the number of solutions they dominate. Giving better scores to solutions that dominate more solutions. At the end a deterministic clustering method is used to ensure diversity among nondominated solutions.
- Strength Pareto Evolutionary Algorithm (SPEA 2): Zitzler, Laumanns and Thiele proposed a variant of SPEA. In SPEA2 after fitness evaluation, all nondominated solutions from current population and from external population are passed to the next population. If the number of these solutions is less than the population size, then the next population is filled with dominated individuals from both populations. The main difference between SPEA and SPEA2 is how fitness functions are calculated. SPEA2 uses a fine-grained fitness assignment that incorporates density information that identifies individuals that have the same fitness value.
- Pareto Archived Evolution Strategy (PAES): It was developed by Knowles and Corne. In this algorithm the crossover is operator is different. A parent generates

one offspring by mutation. If the offspring dominates the parent, the offspring is accepted as the next parent and the iteration continues. If the parent dominates the offspring, the offspring is discarded and the new offspring is generated. If the offspring and the parent do not dominate each other, a comparison set of previously nondominated individuals is used. For maintaining diversity along Pareto front, an archive of nondominated solutions is archived. A new offspring is compared with the archive to verify if it dominates any member of the archive. If yes, then the offspring enters the archive and is accepted as a new parent. The dominated solutions are eliminated from the archive. If the offspring does not dominate any member of the archive, both parent and offspring are checked for their nearness with the solution of the archive. If the offspring resides in the least crowded region in the parameter space among the members of the archive, it is accepted as a parent and a copy is added to the archive.

- Non-dominated Sorting Genetic Algorithm (NSGA): It was developed by K. Deb and his students. Initially a random population, which is sorted based on the no domination concept, is created. Each solution is assigned fitness equal to its no domination level. Binary tournament selection, recombination and mutation are used to create a children population. A combined population is formed from the parent and offspring population using elitism criteria. The population is sorted according to the nondomination relation. The new parent population is formed by adding the solutions from the first front and the followings until exceeding the population size. Crowding comparison procedure is used during the population reduction phase and in the tournament selection for deciding the winner.

- Adaptive Pareto Algorithm (APA): This algorithm uses a new technique called Adaptive Representation Evolutionary Algorithm. The main idea of this technique is to allow each solution be encoded over a different alphabet. Moreover, the representation of a particular solution is not fixed. Representation is adaptive and may be changed during the search process as effect of the mutation operator. The algorithm uses a single population of individuals. The initial population is randomly generated. Each individual is selected for mutation, which is the unique variation operator. The offspring and parent are compared. Dominance relation guides the survival.

CHAPTER 3: Network optimization

3.1 Networks and graph theory

In the previous chapter a review about multiple objective optimization and the most common methods to solve multi-criteria problems was covered. However, networks that are the main subject of this study have not been covered so far. The objective of this chapter is to give a detailed description about what networks are, how they can be used to model engineering problems, and how they can be implemented to solve the reliability network optimization problem applied in telecommunication networks.

Graph theory was introduced as early as 1736 when Euler published a paper about the seven bridges of Konigsberg. The problem was to find a path to cross all the seven bridges of the city of Konigsberg without having to cross any bridge more than once. Euler proved using graph theory that such journey was impossible. He developed a model that can be applied not only for Konigsberg bridges, but to any network of bridges from anywhere. Network theory is a part of graph theory that study networks as a representation of symmetric relations or asymmetric relation between discrete objects covering a large range of fields. Some examples of these networks are: gene regulatory networks, metabolic networks, social networks, Transportation networks, distribution networks, World Wide Web to mention a few [11].

A network can be defined as a series of points or nodes interconnected by communication paths. In Figure 11 several network examples are presented. Nodes are represented by circles and the arcs or lines represent the links of the network. Links in the network can be bidirectional or unidirectional, and each link normally has associated values, functions, related to specific parameters of the network.

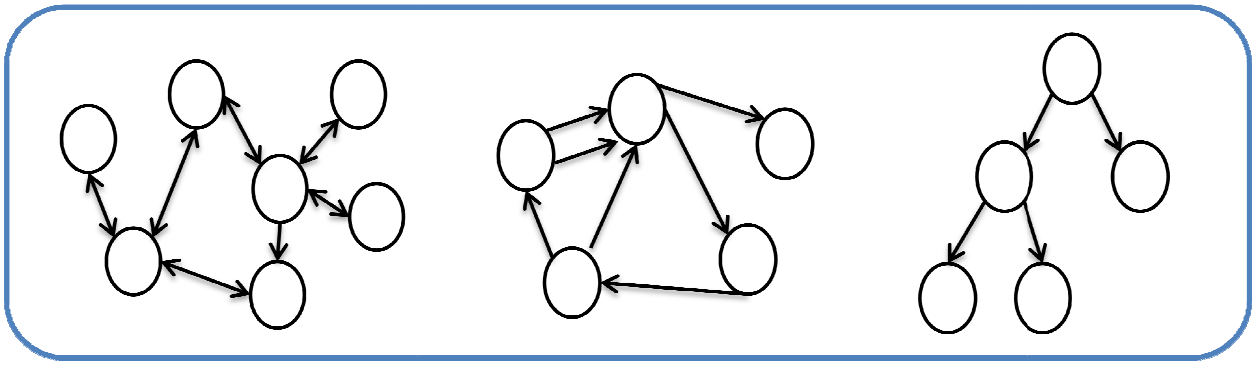

Figure 11: Network Examples

Mathematically a network can be defined as a graph *G*= (*N*, *L*). Where *N*= # of nodes and *L*= # of links and $N \neq 0$ [12]. If the set *L* is empty and the network consists only of nodes the network is called *null network.* On the other hand a network that has all the possible links connected without redundancies is called *full connected network.* These two configurations are presented in Figure 12.

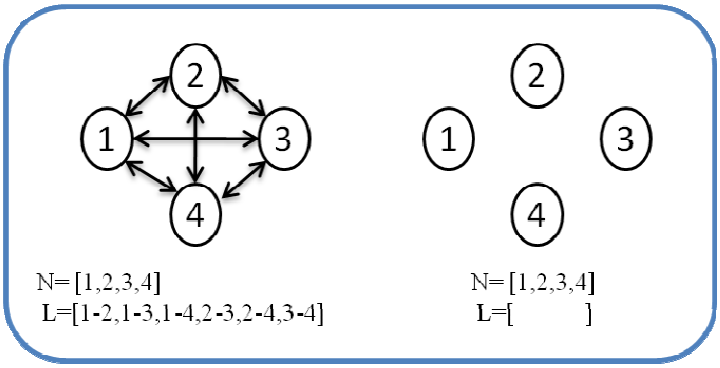

Figure 12: Fully Connected & Null Network

In Figure 11 several network examples are presented. It can be observed from that picture that links can have one direction or can be bidirectional an also that can be more the one link to connect two nodes. This redundancy is not common in network notation and it is normally avoided. A network that its links has direction is called directed network or dinetwork. In Figure 12 a network with bidirectional links is presented.

Finding a path through a network under different conditions is the fundamental objective to use networks to model practical problems. These problems can be applied in many

areas such as transportation, electric distribution, communication networks, etc. The following examples can help to understand how networks can be applied to several fields.

Mechanical networks: The nodes in these networks are joints between arcs representing linear mechanical elements such as rods, beams, and springs. These networks normally generalize the concept of particle systems and extend their application

Electrical Networks: In electrical systems, electrical junctions are represented by the nodes of the network, whereas the arcs represent wires, resistors, batteries, generations, and other electrical components.

Transportation networks: The elements of *N* represent certain places (cities, warehouses, factories, etc. Elements of *L* are transportation links (roads shipping routes, etc). In this case parallel arcs or links are allowed since there may be more than one means of transportation between two locations. The links are commonly bi-directional, unless there is only one direction of transport of interest. Another important characteristic in a transportation network is that each link has a quantity of material passing through one node to another. Such quantities may be subject to various constraints and costs.

Telecommunication networks: In this, case the nodes are places (transmission facilities, satellites, etc) and links are cables, microwaves, etc, the material being transported consists of call, messages, data, etc. links are commonly bi-directional and redundancies between nodes are not commonly allowed. Telecommunication networks are used to analyze several aspects for the network. For instance, estimate how much capacity can be expected to be available between two locations. Another problem is to

analyze the reliability of the network or its vulnerability to disaster. This work focuses in determining how to evaluate the reliability of a telecommunication network.

3.2 Networks representation

The examples presented before represent a network as a picture or figure. This notation is useful to understand connections and to observe how the network is built. However, in order to be able to work with networks using a computational system another representation is needed. There are several forms to represent a network or graph [13]:

- Incidence list
- Adjacency list
- Incidence matrix
- Adjacency matrix

Incidence list: This representation uses an array, also known as a vector or list, each element in the array corresponds with a single link. Each link is formed of two elements; the first is a list number or the number of a link and the second element is the end point of the link. Therefore, an incidence list for a graph *G* is given by:

(a) The number of nodes or vertices *n*

(b) *n* list $L_{1,...,} L_n$, where L_i contain the links beginning in node *i*. Here a link *l*= *ij* is recorded by listing its name and its end point j, that is (*l*,j). For instance a network presented in Figure 13 can be represented as follows:

(a) *n*=4

(b) L_1:(1,2),(6,3) ; L_2:(2,4),(5,3); L_3:(4,4); L_4:(3,1)

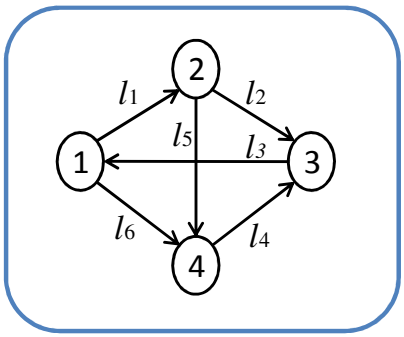

Figure 13: Digraph Example

Adjacency list: this representation is very similar to the incidence list, but in this case each link is formed by one element that represents the end point of a specific link. In this representation, each node N_i has a list of which nodes it is adjacent to. This representation makes easier to find all the nodes which are connected to a single node, since these are explicitly listed. Therefore, an adjacency list for a graph is given by:

(a) The number of nodes or vertices *n*

(b) *n* list $L_{1,...,} L_n$, where L_i contains the nodes *j* or end points for each link *l* (*i,j*). For instance a network presented in Figure 13 can be represented using adjacency list as follows:

(c) n=4

(d) L_1:2,3 ; L_2:3,4; L_3:4; L_4:1

As can be observed both representations have advantages and disadvantages. Incidence list are used when information about links is more desirable than the information about the nodes.

Incidence Matrix: This representation uses a matrix (*N*x*M*) of *M* links by *N* nodes. If the node is an endpoint to the link, a value of 1 is assigned to their crossing, otherwise, a value of 0 is assigned. The Incidence matrix that represents the network from Figure 13 is presented in table 1. This representation represents a terrible waste of space in

memory as every column or row represented by the link can only have two values of 1 while the rest are labeled 0. This situation becomes worse if the network is larger and has just few links inside.

Table 1: Incidence Matrix

	L_1	L_2	L_3	L_4	L_5	L_6
Node 1	0	0	1	0	0	0
Node 2	1	1	0	0	0	0
Node 3	0	0	0	0	1	1
Node 4	0	1	0	1	0	0

Adjacency matrix: This representation uses a matrix *N* x *N*, where *N* is the total number of vertices in the graph. If there is a link from some node x to some node y, then the element would be 1, otherwise it would be 0. The adjacency matrix that represents the network from Figure 13 is presented in Table 2.

Table 2: Adjacency Matrix

	Node 1	Node 2	Node 3	Node 4
Node 1	0	1	1	1
Node 2	0	0	1	0
Node 3	0	0	0	1
Node 4	1	0	0	0

In this case, the graph represented is a directed graph, where the direction is given from node from row *i* to node from column *j*. if the graph from Figure 12 is considered as a bidirectional graph that means that each link is bidirectional the adjacency matrix is given by Table 3. For adjacency an adjacency matrix is easy to observe that the

diagonal of the matrix is always 0 and that for a bidirectional network the matrix is always symmetric. The space needed in memory is n^2 places that represent a lot of memory for a larger network. This representation should be only used to represent a digraph that has many links.

Table 3: Adjacency matrix for a bidirectional graph

	Node 1	Node 2	Node 3	Node 4
Node 1	0	1	1	1
Node 2	1	0	1	1
Node 3	1	1	0	1
Node 4	1	1	1	0

Due to the amount of memory needed to store a network using a matrix representation, list representation is preferred. However, some algorithms used in networks to evaluate the shortest path or the max network flow uses the matrix representation.

3.3 Networks and trees

Networks or graphs that are encountered in most of the applications are connected graphs that mean that each node is connected to at least other node in the network and all nodes work. Among connected graphs trees have the simplest structure and are perhaps the most important ones [14]. The word tree suggests branching out from a root and never completing a cycle. Trees as graphs have many applications in network models, especially in data storage and communication such as telecommunication networks [15].

A tree is a connected, acyclic (if it has no circuits), undirected (links have no direction) graph *G*. If an undirected graph is acyclic but possibly disconnected, it is

called a forest. Many algorithms use in graph theory that work for trees also work for forests. Figure 13 shows three examples: Figure 14(a) shows a tree that satisfies all the conditions to be a tree, and Figure 14(b) shows a forest. The forest in Figure 14(b) is not a tree because it is not connected graph. The example in Figure 14(c) is neither a tree nor a forest, because it contains a cycle in it [15].

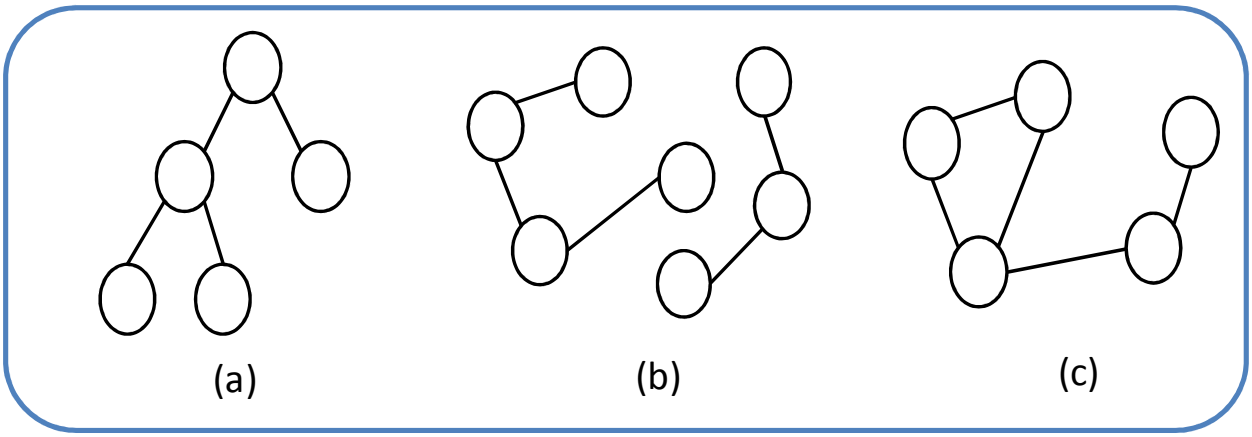

Figure 14: (a) A Tree (b) A Forest (c) A Graph That Contains a Cycle

Summarizing, a graph is a tree if it satisfies the following conditions [13]:

- *G* does not contain any cycles, but adding any further link yields a cycle
- Any two nodes of *G* are connected by a unique path
- *G* is connected and $l=n-1$
- Any link of *G* is a bridge

3.3.1 Spanning trees and cut sets

A spanning tree *T* of a connected, undirected graph *G* is a tree composed of all the nodes or vertices and some (or perhaps all) of the links of G. Informally, a spanning tree of *G* is a selection of links of *G* that form a tree spanning every node. That is, every node lies in the tree, but no loops are formed. On the other hand, every bridge of *G*

must belong to *T*. In this section, some interesting results which relate cut sets and circuits to spanning trees and co-spanning trees are discussed. It is obvious that removal of a cut set *S* from a connected graph *G* destroys all the spanning trees of *G. A* little thought will suggest that a cut set is a set of edges whose removal from *G* destroys all spanning trees of *G*.

3.3.2 Breadth-First Search and Depth-First search

Many graph algorithms require a systematic method of visiting the vertices of a graph. There are two methods that can be used to produce efficient algorithms. The breadth-first search and the depth-first are such methods. To discuss these two algorithms, it is important to introduce two basic data structures: stacks and queues. A stack is a list in which insertions and deletions are always made at one end, called the *top*. The *top* item in the list is the most recently inserted. Stacks are sometimes referred to as LIFO lists (Last in, first out). A queue is a list in which all insertions are made at one end, called the tail, and deletions are made at the head. Queues are referred to as FIFO lists (first in, first out) [17].

Breadth-First Search (BFS): BFS is a search method that aims to expand and examine all nodes of a graph or combination of sequences by systematically searching through every solution. In other words, it exhaustively searches the entire graph or sequence without considering the goal until it finds it. From the standpoint of the algorithm, all child nodes obtained by expanding a node are added to a FIFO queue. The pseudo code is presented next.

Input: An unweighted graph or digraph and a start node *u*

Idea: Maintain a set *R* of nodes that have been reached but not searched and a set *S* of nodes that have been searched. The set *R* is maintained as a FIFO list (queue) so that the first vertices found are the first vertices explored.

Initialization: $R=\{u\}$, $S= 0$, $d(u,v)=0$.

Iteration: as long as R ≠0, we search the head node *v* of *R*. The neighbors of *v* not in *R* or *S* are added to the tail of *R* and assigned distance $d(u,v) + 1$, and then *v* is removed from the head of *R* and placed in *S*. Figure 15 shows the behavior of the BFS method.

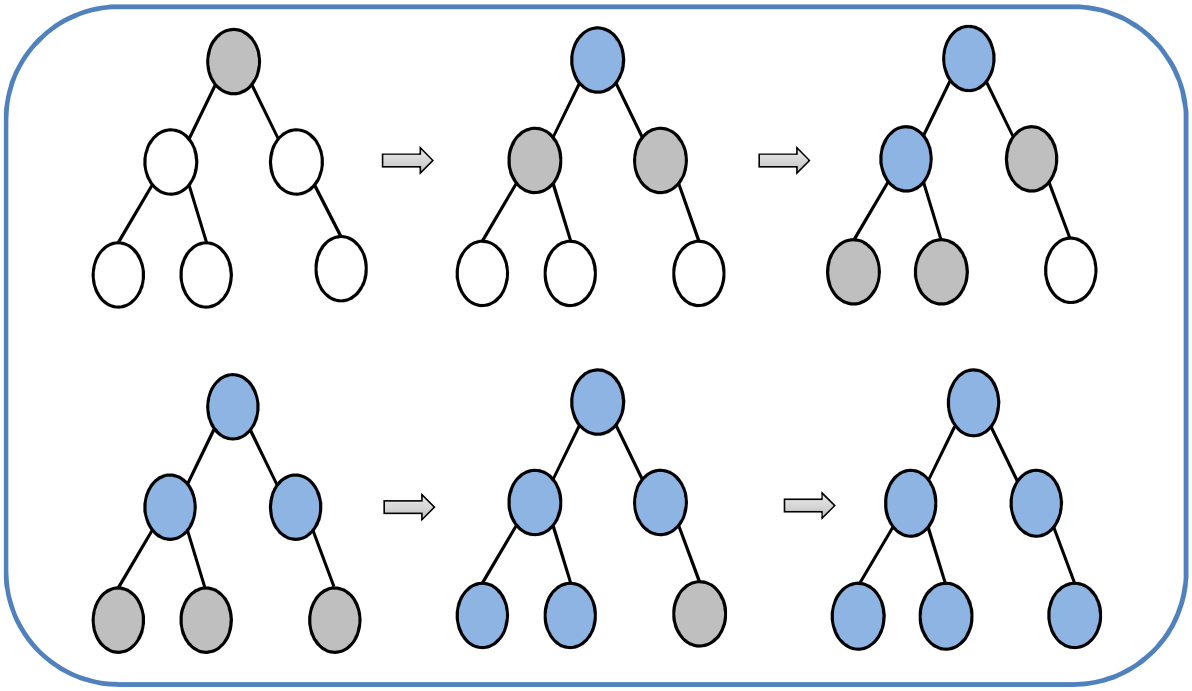

Figure 15: BFS

Space Complexity: Since all of the nodes of a level must be saved until their child nodes in the next level have been generated, the space complexity is proportional to the number of nodes at the deepest level. Given a branching factor *b* and graph depth *d,* the asymptotic space complexity is the number of nodes at the deepest level, $O(b^d)$. When the number of nodes and links in the graph are known ahead of time, the space complexity can also be expressed as O (| L | + | N |) where | L | is the cardinality of the

set of links (the number of links), and | N | is the cardinality of the set of nodes. In the worst case, the graph has a depth of 1 and all nodes must be stored

Time complexity: In the worst case, breadth-first search has to consider all paths to all possible nodes the time complexity of breadth-first search asymptotically approaches $O(b^d)$. The time complexity can also be expressed as O(| L | + | N |) since every node and every link will be explored in the worst case.

Depth-First search (DFS): It is a search that progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children. Then the search backtracks, returning to the most recent node it hasn't finished exploring. In a non-recursive implementation, all freshly expanded nodes are added to a stack for exploration [16]. The pseudo code of DFS is presented next[17].

Input: An unweighted graph or digraph and a start node *u*

Idea: Maintain a set *R* of nodes that have been reached but not searched and a set S of nodes the have been searched. The set *R* is maintained as a LIFO list (stack) so that the first nodes found are the last vertices explored.

Initialization: $R=\{u\}$, $S= 0$.

Iteration: as long as $R \neq 0$, we remove the top element *v* of *R* and search *v*. The neighbors of *v* not in *R* or *S* are added to the top of *R*.

Figure 16 shows how BFS Works

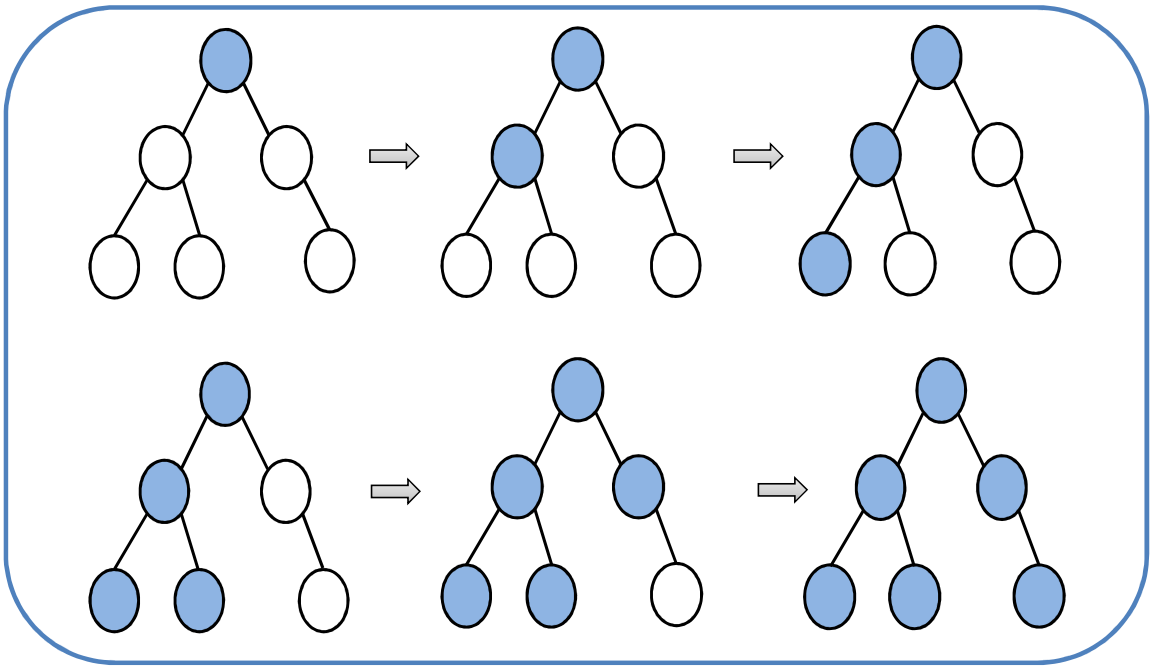

Figure 16: DFS

The time and space analysis of DFS differs according to its application area. In theoretical computer science, DFS is typically used to traverse an entire graph, and takes time O(|N| + |L|), linear in the size of the graph. In these applications it also uses space O(|N| + |L|) in the worst case to store the stack of vertices on the current search path as well as the set of already-visited vertices. Thus, in this setting, the time and space bounds are the same as for breadth first search and the choice of which of these two algorithms to use depends less on their complexity and more on the different properties of the nodes orderings the two algorithms produce.

3.4 Telecommunication & computer networks.

This chapter has covered so far network and graph theory in a general way. However, this thesis work focuses in the reliability optimization of a telecommunication network. At the beginning of this chapter, a brief definition about telecommunication

networks was given. However, more detailed information is needed to understand the problem that this work wants to address.

A telecommunications network is a network with links and nodes arranged so that messages may be passed from one part of the network to another over multiple links and through various nodes. The information passed over the network can be either voice or data. Examples of telecommunications networks are:

- Computer network
- Internet Network
- Public switched telephone network
- Global Telex network
- Aeronautical ACARS network

All telecommunication networks are made up of five basic components that are present in each network environment regardless of type or use. These components include terminals, processors, telecommunications channels, computers, and software. Terminals are the nodes that represent the starting and stopping points in any telecommunication network environment. Telecommunications processors are support data transmission and reception between terminals and computers by providing a variety of control and support functions. Telecommunications channels are the way by which data is transmitted and received. Telecommunication channels are created through a variety of media of which the most popular include copper wires and coaxial cables. Telecommunications control software is present on all networked computers and is responsible for controlling network activities and functionality [18-19].

Most networks can be classified into several different types. These include wide area networks (WAN), local area networks, (LAN), virtual private networks (VPN), client/server networks, network computing, and peer-to-peer networks [20-21].

Wide Area Network (WAN): Any network that encompasses a large geographic area is referred to as a WAN or Wide Area Network. Many large businesses and government agencies use WANs to keep their employees and citizens connected as well as provide a quick and effective way to send and receive information.

Local Area Network (LAN): Similar in many ways to WANs; Local Area Networks or LANs are responsible for connecting computers in a much smaller limited physical area. A good example of a LAN would be a hotel's wireless Internet offering which is self-contained within their own facility. There are multiple standards for Local Area Networks. Examples include IEEE 802.3 (Ethernet), IEEE 802.11 (Wi-Fi) or ITU-T G.hn

Personal Area Network (PAN): A Personal Area Network (PAN) is a network that is restricted to the area of a person's body. It is much smaller than a Local Area Network. It typically incorporates ad hoc connections to other PANs or directly to Bluetooth devices.

Virtual Private Network (VPN): Virtual Private Networks or VPNs are a type of network that builds off of the concept of a WAN, however relies upon the internet and an encrypted connection mechanism to establish a secure environment for internal or external employees or customers.

Client/Server Network: The Client-Server network architecture continues to be the main architectural choice for most enterprise network computing. In a client/server environment the client (*i.e.* PC) relies on a LAN to connect with a back office network server that is responsible for the connection, retrieval, and storage of data and other critical company or personal information.

Network Computing: Network computing is a network architecture that has grown with the Internet and resulting connection speeds. In a network computing architecture a

computer uses its web browser to connect to another network computer that actually is running the application. A good example of this architecture in use is Google Docs, or Microsoft Office online. Both services allow users the ability to login to Google or Microsoft servers respectively and work similarly to how it would be performed on their own computing environment.

Peer-to-Peer Network: Peer to peer networks are now beginning to be realized for the positive benefits they provide and not as only used for the sharing files. Peer-to-peer networks can be separated into two major types: Central Server and Pure. In a central server environment one host server maintains all active connections and shared information. When information is requested, the central server informs the user where they can receive the file and allows the connection directly to the other PC to download. A pure peer-to-peer network has no central server to maintain active users. It relies instead on the individual computers to seek out all other computers offering the same information being requested [19].

3.5 Network Reliability

In a Telecommunication network, regarding the type of network, the communication over the network assumes the availability of a reliable network. At this level, data are expected to traverse the network and to arrive intact at their destination. The physical systems that compose a network, on the other hand, are subject to a wide range of problems, ranging from signal distortion to component failures [22].

The concept of all- terminal network reliability, proposed by Kel'mans, is defined as follows: Let *G* be a graph that represents the topology of a network. Assume that the nodes are perfectly reliable, but the links operate independently with a probability *p*. The

network is operational if the underlying probabilistic graph is connected. In the reliability network problem there are three main cases or scenarios:

All- terminal: The all-terminal reliability can be defined as the probability that each node in a network can communicate with every other node [23]. It means that the network has to form at least one spanning tree. In order to find out if there is a spanning tree several algorithms have been proposed by previous researches. The efficiency and computational time of some algorithms are discussed in Bazlamacci [24]. In the present research, the method selected to investigate if there is a spanning tree is the Breadth-First Search method, and to evaluate the network reliability, a Monte Carlo simulation is implemented.

k-terminal: The *k*-terminal reliability can be defined as the probability that considering *k* specified nodes of a network there exits paths between each pairs of the k nodes [23]. Hardy *et al*. [25] proposed a binary decision procedure to calculate the exact network reliability. A similar approach is also used in Ghasemzadeh *et al*. [26]. In the present research, the Breadth-First Search method used to evaluate the all-terminal reliability was modified to be implemented in the *k*-terminal case.

Two-terminal: The two-terminal case represents a sub case of the *k*-terminal case. Therefore, the methodology used to evaluate this case is the same as the one used in the previous case but the only difference is that the number of nodes required to be connected are always two. Jane e*t al*. [27] solved the two-terminal network reliability problem using a practical bounding algorithm. Cook *et al*. [28] analyzed the two-terminal problem in a mobile ad-hoc wireless network.

A solution for the network reliability problem is a network configuration that consists of a list of the links that have to be active in order to maximize the reliability. Due to the large amount of possible solutions and the combinatorial nature of the problem, the network reliability optimization problem is considered as an NP-Hard allocation problem. In literature researchers had addressed the network reliability design problem either considering the maximization of the system reliability for all cases (all-terminal, two-terminal, *k*-terminal) or the minimization of total system cost subject to diverse constraints [29]. Just a few approaches have considered the problem as a multiple objective optimization problem.

The exact calculation of network reliability is an NP-hard problem by itself, with computational time and effort increasing exponentially as more nodes and links are added to the network [30]. Due to the complexity in the reliability calculation of these types of networks, minimal cut sets have been previously used in [31]. Also, a similar approach is presented in [32] using a Tree Cut and Merges algorithms. Although this method has proved to be effective, it becomes impractical to use it in relatively large networks. For those kinds of networks meta-heuristic methods have proven to obtain a very good approximation for this metric. For instance Smith *et al.*[30] developed a methodology using an artificial neural network (ANN). Ramirez Marquez *et al* [33] and Smith *et al.* [34] used Monte Carlo (MC) simulation to estimate the network reliability. In [35] a combination between ANN and MC simulation is implemented to estimate the all-terminal reliability of a network. Other interesting approaches to evaluate network reliability have been presented in literature. For instance, in [36] a binary-decision-diagram and sensitivity analysis was implemented for the two-terminal case. Whereas, in [32] the tree cut and Merge algorithm that is a Hybrid modification for a MC was used

reducing the high computational cost of MC simulation that is one of the most important disadvantages of this method. In this paper, a MC simulation was the chosen method for reliability estimation. Since MC simulation is a meta-heuristic method to iteratively evaluate a deterministic model using sets of random numbers as inputs, MC simulation is suitable to be used when the model is complex, nonlinear, or involve several parameters. Even though there are many different methods to evaluate the reliability of the network, MC simulation is the method used in this work.

3.5 Monte Carlo Simulation

A Monte Carlo (MC) method is a technique that involves using random numbers and probability to solve problems. The term Monte Carlo Method was coined by S. Ulam and Nicholas Metropolis in reference to games of chance, a popular attraction in Monte Carlo, Monaco.

There is no single Monte Carlo method; instead, the term describes a large and widely-used class of approaches. However, these approaches tend to follow a particular pattern:

- Define a domain of possible inputs.
- Generate inputs randomly from the domain using a certain specified probability distribution.
- Perform a deterministic computation using the inputs.
- Aggregate the results of the individual computations into the final result.

In the next chapter, a more detailed explanation of how MC simulation was used to evaluate the network reliability will be presented.

CHAPTER 4: The proposed algorithm

In previous chapters, several important concepts were epxlained. In chapter one, the network reliability problem was introduced as a multiple oblective optimization problem. In the second chapter, the notion of multiple criteria optimization was introduced and several methods used in optimimization problems that involve more than one objectives were presented. Chapter 3 introduced network and graph theory. In that section the concept of telecomunication network was presented and the different type of telecomunications were briefly expained.. In Chapter 3 the complexity of obatining the relibility of a network and some methodologies and aproaches were presented. In the present chapter, the problem and the methodoly proposed to solve the reliability network optimization problem as a multiple objective optimization are presented.

The telecomunication network design problem cosists of dermining the optimal collection of links that have to be active in order to optimize several important objectives such as reliability, cost, weight among others. A solution for the problem is a network configuration or network topology that consists in a list of the links that simultaneously optimize the desired objectives. Due to the large amount of possible solutions and the nature of the problem this problem is considered as an NP-Hard combinatorial allocation problem. A simple method to solve these type of problems is to evaluate all the considered objectives for all the possible solutions and choose the one which has the better objective values. Usually, because of the nature of the problem a unique solution does not exist and the selection of one solution over others is also a problem that has been fully studied in literature. However, the evaluation of all the possible solutions become impractical in real life problems such as in the network reliability

problem where the number of solutions increases exponentially as the number of nodes is added to the network.

Due to the importance of designing reliable networks, it is important to develop a method that can solve this type of problems fast and efficiently. It is easy to observe that the reliability of the network can be improved increasing the number of spanning trees or redundancies in the network, but each link in the network is associated with a specific cost. Even though reliability is important, the system cost is very important as well. Reliability and cost of a network are two conflictive objectives that have to be optimized simultaneously. Figure 17 shows a wireless network with 10 nodes and two possible configurations. It can be inferred that with more links better reliability and higher cost, and with less links the opposite scenario is obtained. In Figure 17, the circles or nodes represent terminals (routers switches computers) and lines represent an established link. The present work adds an additional objective to the equation that is called weight and it is to be minimized. The combinatorial allocation problem presented consists in selecting which links have to be activate in order to obtain the highest reliability, the lowest cost, and the lowest weight.

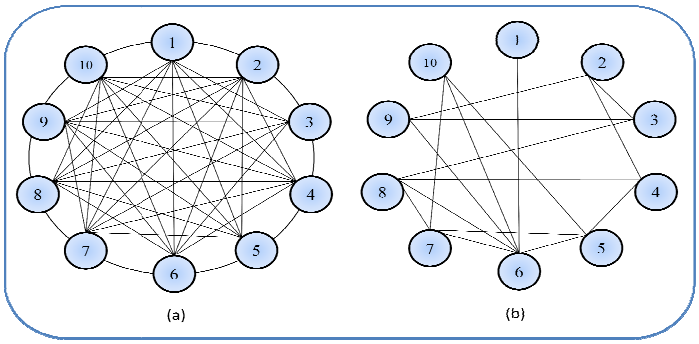

Figure 17:(a) Fully Connected Network: Higher Reliability, Higher Cost

4.1 Notation & Assumptions

In previous section the telecommunication design problem has presented. In order to model the problem as a multiple objective optimization problem some assumptions are considered:

- Nodes are perfectly reliable.
- Link costs, weight and reliability are fixed and known.
- Each link is bi-directional and there are no redundant links in the network
- Links are either operational or failed
- The failures of links are independent.
- No repair is considered

The notation implemented to model the problem and used in the algorithm is as follows:

G	Probabilistic graph
$\mathbf{x}$	Network design vector $\mathbf{x}= (x_{12}, .., x_{1n}, x_{23}, .., x_{2n}, .., x_{ij}, .., x_{nn-1})$ $i<j$
N	Set of nodes
L	Set of links
$R(\mathbf{x})$	All terminal reliability of $\mathbf{x}$
$\mathbf{p}$	Reliability vector of (i,j)
$\mathbf{y}$	Subsystem state design vector
$C(\mathbf{x})$	Cost of network of vector $\mathbf{x}$
$\mathbf{c}$	Cost vector of link (i,j)
$W(\mathbf{x})$	Weight of network $\mathbf{x}$
$\mathbf{w}$	Weight vector of link (i,j)
$f_1(\mathbf{x})$	Fitness metric 1: dominance count-based
$f_2(\mathbf{x})$	Fitness metric 2: distance-based
$F(\mathbf{x})$	Aggregated fitness
n	population size
g	number of generation

E Elitism parameter

Cross Crossover parameter

The mathematical model to be optimized in this work is presented in Equation 10. The objectives are to minimize the cost and weight while maximizing the reliability of the network configuration. It is assumed that the number of nodes, cost, weight, and reliability vectors are known.

$$\text{Max } R(\mathbf{x}), \text{ Min } C(\mathbf{x}), \text{ Min } W(\mathbf{x}) \quad (10)$$

s.t.

$x_{ij} \in \text{Bin}(0,1)$

Where:

x = Network design vector $\mathbf{x}= (x_{12}, .., x_{1n}, x_{23}, .., x_{2n}, .., x_{ij}, .., x_{nn-1})$ $i<j$

$R(\mathbf{x})$ =Network reliability of **x**

C(**x**) =Total network cost

W(**x**) =Total weight cost

Vector **x** represents our decision variable. In other words, **x** represents a network configuration or a possible solution for our optimization problem. $R(\mathbf{x})$, C(**x**), W(**x**) are the three objectives to be optimized. In order to model our decision variable **x,** consider a network $G= (N, L)$ in which N represents the number of nodes and L represents the set of bidirectional links. $|L|=l$. For a fully connected network $l = (n(n-1))/2$. The network state vector $\mathbf{x}=(x_{12},..,x_{1n},x_{23},..,x_{2n},..,x_{ij},..,x_{nn-1})$ represents the state of all the links of the network (network configuration). The value $x_{ij}=0$ denotes that the link is broken and $x_{ij}=1$ denotes that the link between nodes i and j are established. Each link in the network has a specific reliability. This reliability is represented using a vector

p=(p_{12},..,p_{1n},p_{23},..,p_{2n},..,p_{ij},..,p_{nn-1}). Costs and weight for each link are represented by vectors **c** and **w** [33]. For illustration purposes, consider Figure 18 with a small network showing that links between nodes {1-2}, {2-3}, and {1-4} are active.

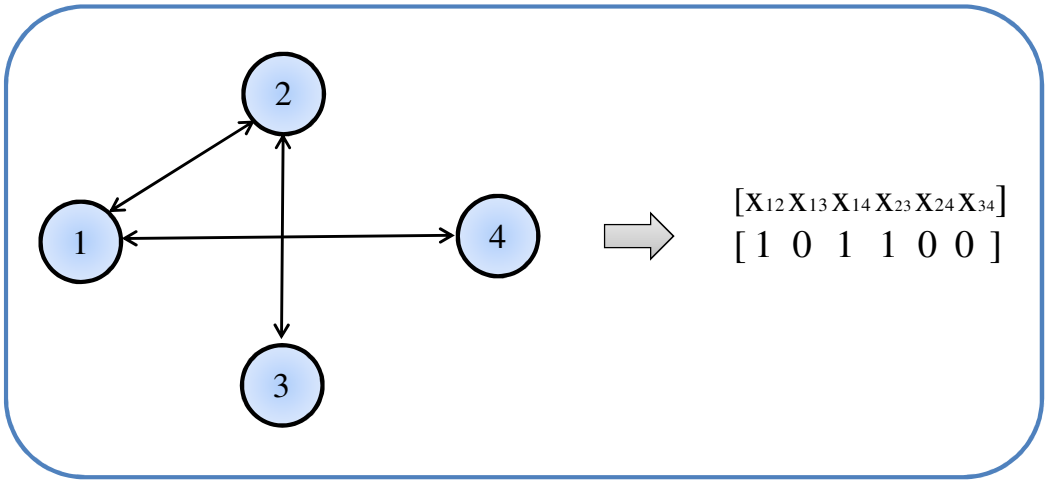

Figure 18: Example of a Network Configuration

The problem presented is a combinatorial optimization problem that considers three objectives to be optimized simultaneously. In Chapter 2 the most common methods to solve multi-criteria decision making problems were explained. An option is to choose a mathematical method such as goal programming and lexicographic method. However these methods do not really optimize all the objectives simultaneously. Utility theory is another mathematical method used to solve multiple objective optimization problems even though this method obtains a very good solution its implementation is very difficult and sometimes impossible.

Meta-heuristic methods are another alternative to solve this type of problems. Meta-heuristic methods normally mimic a natural behavior for a specific phenomena such as evolution, how animal find their food, how the human brain works among other. The behaviors are adapted into an algorithm to solve a variety of problems. These methods have proved to obtain very good solutions in very complex problems. However one

important drawback is that the solutions obtained are just approximations and they cannot guarantee to obtain a global optimal solution.

Among meta-heuristic methods, evolutionary algorithms are commonly used to solve optimization problems due to the flexibility of the approach and because it is relatively easy to implement. There are several multiple objective evolutionary algorithms that have been proposed in literature (see chapter 2). However, they represent general algorithms that need to be adapted to a specific problem. This work proposes a new MOEA applied to solve the three cases of network reliability problems considering the three objectives to be optimized simultaneously.

A mentioned in previous chapters, a MOEA is an evolutionary algorithm. Its behavior is based in Darwin's evolutionary theory. The main idea behind this concept is that the best individuals in a population will survive and generate a better individual. If we consider individuals as a solution and we consider that better solutions have better objectives, then it is logical to think that those solutions can generate also good solutions.

The proposed MOEA is basically divided in four stages: initialization, evaluation, selection, and reproduction. A general flow chart is presented in Figure 19, each step will be explained in detail in further sections.

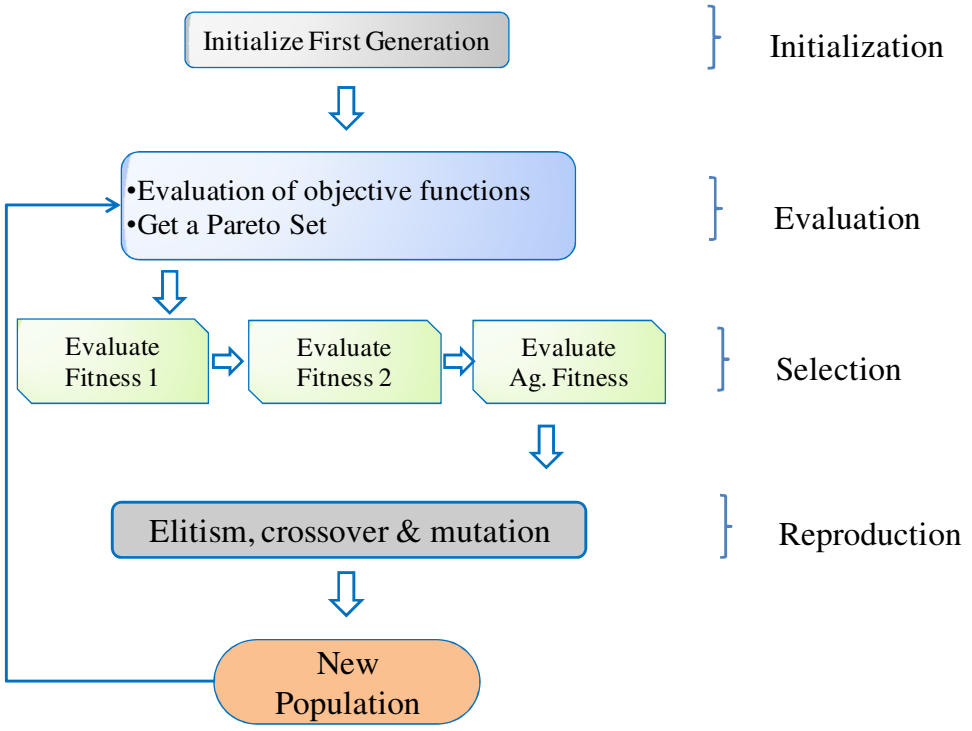

Figure 19: Algorithm's Flow Chart

4.2 Algorithm's steps.

4.2.1 Initialization.

The first step of the algorithm is to generate the initial generation of possible solutions. This step is vital because all the next generations will be created from the first one.

The way the many proposed evolutionary algorithms deal with this issue is generating the first generation randomly in order to obtain diversity of solutions in the first generation. How that random generation does affects the performance in terms of the number of iterations to find the approximate solution of the algorithm has been the topic of many previous works. For instance Maaranen [37] presented an study of how Quasi-random initial population affects GAs. On the other hand Karci [38] proposed a

method of generating the initial population by using upper and lower bounds of variables instead of of pseudo-random sequence. Park *et, al* [39] presented that generating a good initial population is very important and they used G&T algorithm to generate the initial population for a scheduling problem.

In the presented method, *n* number of possible network design configurations (**x** vectors) are randomly generated. Each generated solution has to represent a possible solution for example a null network or network with no links is considered as no feasible solution. Depending of the case that is considered (all-terminal, *k*-terminal, and two-terminal) each gerated solution is checked for feasibility. Figure 20 shows an example of an initial population of 8 chromosomes for a network with 4 nodes. Each vector represents a possible solution for the problem. How to evaluate each objective (R(**x**),C(**x**), and W(**x**)) for each solution is explained in the next section

x_{12}	x_{13}	x_{14}	x_{23}	x_{24}	x_{34}	$R(\mathbf{x})$	$C(\mathbf{x})$	$W(\mathbf{x})$
0	1	0	0	0	1	0.98	300	0.78
1	1	0	0	1	1	0.97	325	0.75
1	0	0	1	1	1	0.90	200	0.251
1	1	0	0	0	0	0.85	500	0.97
0	1	0	0	0	1	0.78	450	0.88
0	1	1	0	0	1	0.99	500	0.69
1	1	1	0	0	1	0.85	198	0.82
0	1	1	1	0	1	0.95	330	0.78

Figure 20: Initial Population

4.2.2 Evaluation.

In this stage, the three objectives considered will be evaluated for each generated network. The evaluation of the cost and weight does not represent any problem and can be calculated straight forward. However, reliability evaluation is more complicated.

The all-terminal network reliability evaluation is an NP-hard problem (Chapter 3). There are two main approaches to obtain the all-terminal reliability. One option is to obtain an exact calculation using analytical methods [40] and, another one is using estimation by Monte Carlo simulation [41]. In order to evaluate the all terminal network reliability, several approaches have been proposed. Rocco [23] proposed a method using a cellular automata and Monte Carlo simulation. Other approaches using MC simulation are presented in [34]. Due to the flexibility an accuracy of the method, MC simulation was used to obtain the all-terminal network reliability in the present work

4.2.2.1 Reliability evaluation.

MC simulation is a probabilistic simulation method that can be used to solve different types of problems. The reliability of the network is evaluated generating randomly many possible networks configurations and determining how many of the networks generated are connected. The connectivity is evaluated for all-terminal, *k*-terminal, and two-terminal. The method works as follows: Given a proposed network design (vector **x**) and a given vector **p**, set Q=0 (# of successes). Then, generate a subsystem state vector **y** (dependent on vector **x**). To obtain vector **y,** a random number between 0 and 1 is generated for each link in the network and compared to vector **p**. Then, evaluate if the new network design defined by vector **y** has connectivity depending of the case evaluated (all-terminal, *k*-terminal, and two-terminal.) and update the counter of

connected systems (Q=Q+1). The procedure is repeated for a specific number of runs (10,000, or more). The reliability of the **x** vector or chromosome is obtained dividing the # of successes by the # of runs. The described method can be observed in the pseudo code presented in Figure 21.

```
Initialize RUNS=0, Q=0, y=0

For RUNS = 1:10000

For i=1,..,n and j=1,..,n i< j

If x_ij=1 then

If rand() < p_ij then y_ij=1 else y_ij= 0 end if

End if

End For

if a subsystem vector y form a path

( depend of the case evaluated)
```

Figure 21: Pseudo Code for Network Reliability Evaluation

The pseudo code presented in figure 21 is explained in more detail in figure 22.

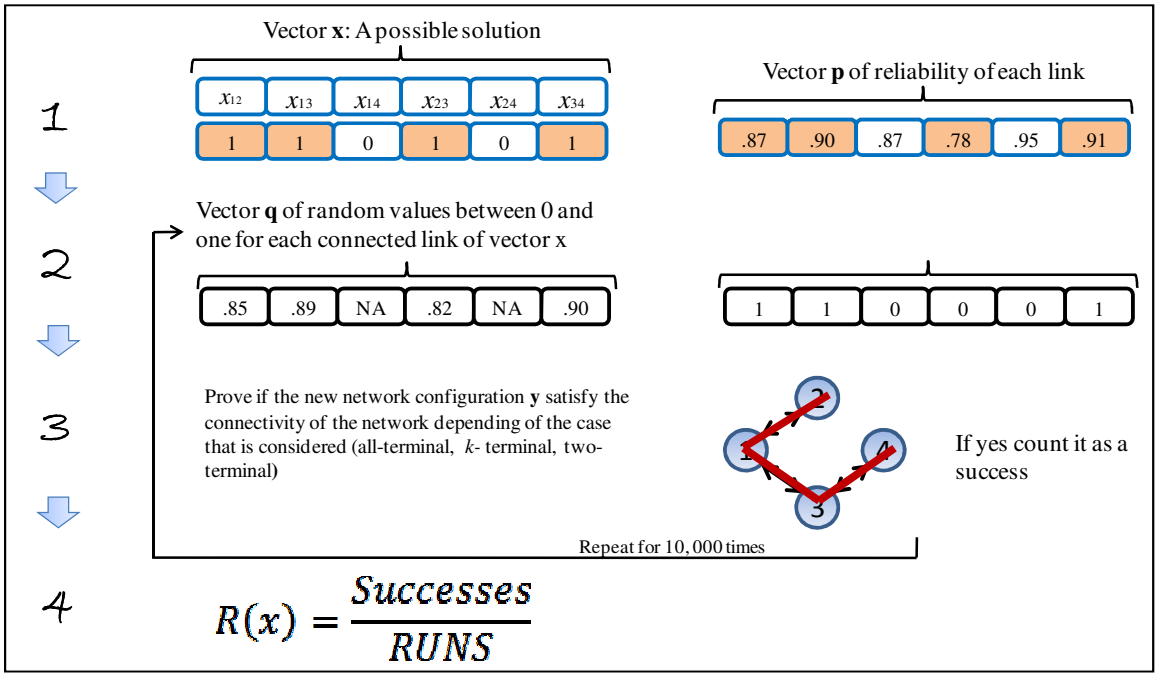

Figure 22: MC Simulation Method

4.2.2.2 Cost & Weight Evaluation:

Cost and weight evaluation is easier than the evaluation of reliability. Detailed explanation of the procedure used is not necessary. Figure 23 shows in an easy way how cost is obtained.

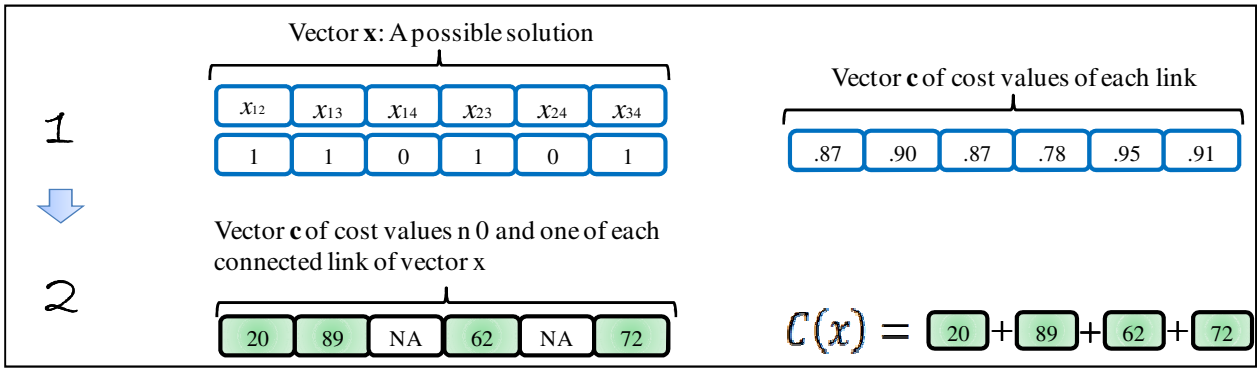

Figure 23:Cost Evaluation

Once the three objectives are evaluated for each possible solution or chromosome in the population the next step is selection. Better solutions are needed to generate new solutions. Based in the Pareto dominance concept, just nondominated solutions are considered for selection. Therefore, the previous step before selection is to eliminate all the dominated solutions from the population. Some works have proposed that combing good solutions with bad solutions generate better solutions than solutions generated just for good solutions. This work bases its behavior in the Pareto dominance concept working just with nondominated solutions.

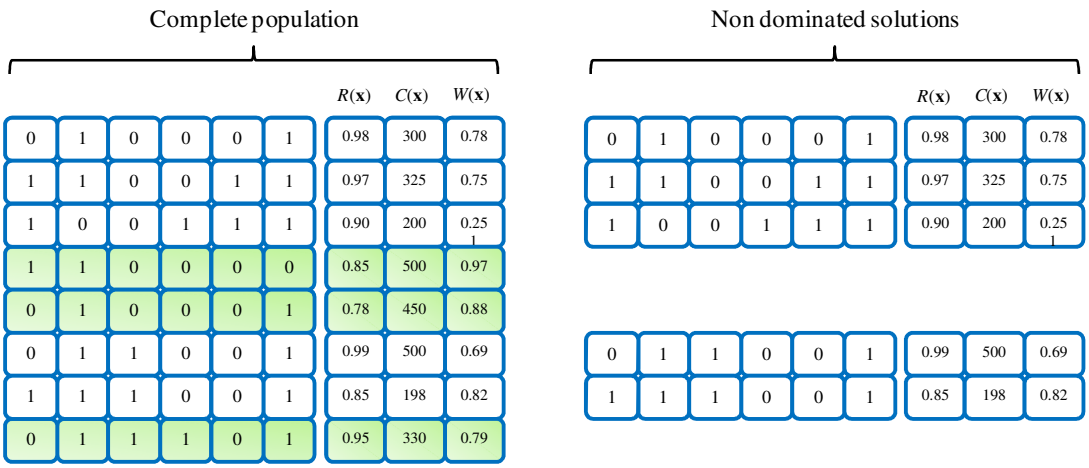

Figure 24: Eliminate dominated solutions

4.2.4 Selection.

This stage of the algorithm has special importance because in this step the solutions that will be part of the new solution for next generation are selected. In order to select the best solutions among the set of no dominated solutions, two fitness functions are considered [42]. The main idea for a fitness function in evolutionary algorithms is to measure the *quality* of the represented solution. This topic has been considered in many previous works because selection is a vital step in any evolutionary algorithm. For instance, Sano & Kita [43] proposed a GA for optimization of continuous fitness functions with observation noise utilizing history of search so as to reduce number of fitness evaluation. Kuncheva [44] proposed an editing technique for the *k*-nearest neighbor (*k*-NN). Other approach treats the problem using a Vectorizing Fitness function. These works take in to consideration the two fitness functions presented in [42]

The first fitness metric, $f_1(x)$, is a dominance count-based metric. It aims to select individuals which are more dominating (intended to achieve proximity). While the second fitness metric, $f_2(x)$, is distance-based. This metric is intended to maintain

population diversity. Solutions that are farther away respect to other solutions (Euclidian distance) have better fitness values.

4.2.3.1 Fitness metric 1.

This fitness metric intends to obtain solutions that are close to the true Pareto front of solutions. By true Pareto Front, it is understood the set of nondominated solutions that is obtained when all the possible solutions in the problem were obtained. In other words the true Pareto is the set of global optimal solutions. However, the true Pareto front is never known. Hence, selecting solutions that are close to the true Pareto front is not straight forward.

This fitness metric is dominance count based. The main idea behind this metric is that solutions that are close to the true Pareto front tend to dominate more solutions than solutions that are farther away from the true Pareto front. This idea is presented in Figure 25. Closer solutions to the true Pareto front dominate more solutions than other solutions.

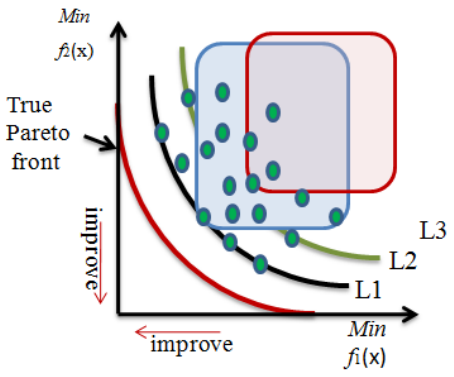

Figure 25: Dominance Count Based Fitness Concept

This metric basically gives a better fitness value to solutions that dominates more solutions and a lower value to solutions that dominate fewer solutions. Consider the

population presented in Figure 24. Figure 26 shows the dominance count for each solution and the given fitness value for each solution.

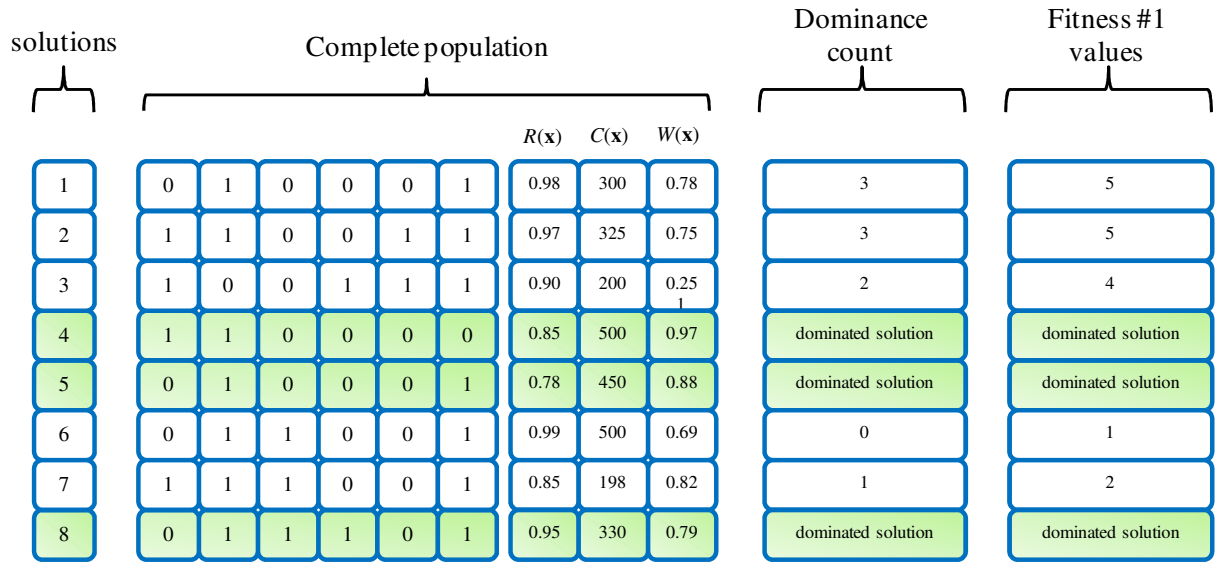

Figure 26: Fitness 1 Calculation

The fitness values are given as follows: Let's consider the example in Figure 25, the maximum dominance count is 3, then we consider 5 intervals and divide the maximum dominance count by the number of intervals, then we give a fitness value depending in the interval that each solution belongs. This procedure is presented in Table 4. The number of intervals can be defined by the designer.

Table 4: Fitness Values for Intervals

Fitness value	***Intervals***	***solutions***
1	0 ≤ Dominance count < 0.6	6
2	0.6 ≤ Dominance count <1 .2	7
3	1.2 ≤ Dominance count < 1.8	0
4	1.8 ≤ Dominance count < 2.4	3
5	2.4 ≤ Dominance count ≤ 3.0	1,2

4.2.3.2 Fitness metric 2

This fitness metric intends to achieve diversity of solutions. In other words, it intends to cover all the search space. Sometimes, generated solutions cover just one part of the search space and these solutions normally generate solutions near to them. This situation is not desired because there is a risk to stay in a local optimal and not find a global optimal solution. In order to prevent this situation fitness 2 (distance-based metric) give a better value to solutions that are far away from other solutions. By doing this, it is assumed to obtain a better spread of the solutions. Figure 27 shows a scenario where some clusters are generated. And the objective is to select solutions that farther away from others and not only solutions from the same cluster.

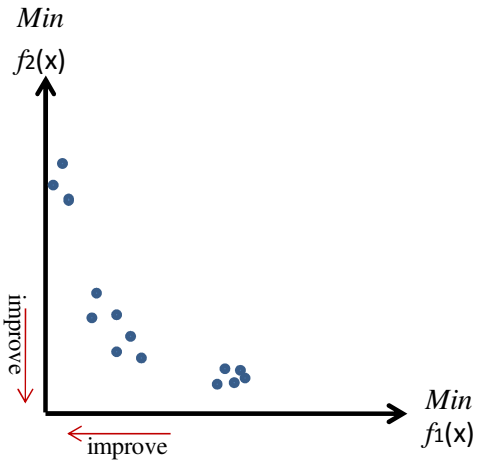

Figure 27: Solutions in Clusters

Fitness metric 2 is distance-based and is evaluated as follows: Consider again the nondominated solutions from Figure 23. The first step is to normalize all the objectives in order to have all the objectives in the same terms. Table 5 shows the objectives normalized. Since reliability is wanted to be maximized and the other two objectives minimized, negative reliability is considered to have all objectives in minimization terms.

This will make calculation easier. However, this will not impact the reliability evaluation at all.

Table 5: Normalized objectives

Non dominated Solution	***Negative Reliability***	***Cost***	***Weight***
1	0.071429	0.337748	0.929825
2	0.142857	0.42053	0.877193
3	0.642857	0.006623	0
6	0	1	0.77193
7	1	0	1

The next step is to compute the Euclidean distance from each solution to the others. Next, the sum of the distances from each solution to the rest of the solutions is obtained, and the maximum and minimum value of all the sums is calculated. Table 6 shows these values.

Table 6: Euclidean Distance

Individual	***1***	***2***	***3***	***6***	***7***
1	0	0.121346	1.140504	0.684551	0.990577
2	0.121346	0	1.091232	0.606032	0.962611
3	1.140504	1.091232	0	1.412777	1.061883
6	0.684551	0.606032	1.412777	0	1.432486
7	0.990577	0.962611	1.061883	1.432486	0
Sum	2.936978	2.781221	4.706395	4.135846	4.447557
Min	2.781221				
Max	4.706395				

The ranges of values are considered in the same manner for the fitness metric 1. The values for each solution for 5 ranges are presented in table 7.

Table 7: Intervals for Fitness Metric 2

Fitness value	***Intervals***	***solutions***
1	2.781221 ≤ Dominance count < 3.166256	1,2
2	3.166256 ≤ Dominance count <3.551291	0
3	3.551291≤ Dominance count < 3.936326	0
4	3.936326≤ Dominance count < 4.321361	6
5	4.321361≤ Dominance count ≤ 4.706395	3,7

Now fitness metric 2 is evaluated. Figure 28 shows the nondominated solutions with their respective fitness values

solutions	Complete population						$R(\mathbf{x})$	$C(\mathbf{x})$	$W(\mathbf{x})$	Sum of distance	Fitness # 2 values
1	0	1	0	0	0	1	0.98	300	0.78	2.9369	1
2	1	1	0	0	1	1	0.97	325	0.75	2.78122	1
3	1	0	0	1	1	1	0.90	200	0.25	4.7063	5
4	1	1	0	0	0	0	0.85	500	0.97	dominated solution	dominated solution
5	0	1	0	0	0	1	0.78	450	0.88	dominated solution	dominated solution
6	0	1	1	0	0	1	0.99	500	0.69	4.1358	4
7	1	1	1	0	0	1	0.85	198	0.82	4.4475	5
8	0	1	1	1	0	1	0.95	330	0.79	dominated solution	dominated solution

Figure 28: Fitness 2 Calculation

4.2.3.3 Aggregated fitness metric

Finally, the third fitness metric used is the aggregated fitness metric, $f_a(x)$. The aggregated fitness metric is the result of the sum of fitness metric 1 plus fitness metric 2; $f_a(x) = f_1(x) + f_2(x)$. It aims to weigh both metrics equally. Then, the nondominated solutions are ranked based on this aggregated fitness metric. At this point it is assumed that solutions with better aggregated fitness value are solutions that are closest to the true Pareto front and also farther away from other solutions. Figure 29 shows nondominated solutions ranked from their respective aggregated fitness values

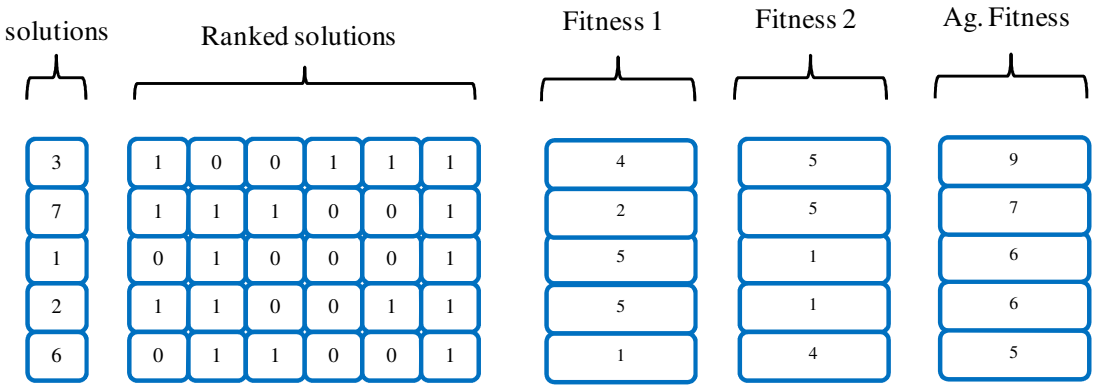

Figure 29: Aggregated Fitness Metric

4.2.4 Reproduction:

In this section the current ranked solutions for the previous step will generate the new individuals that will be part of the next generation. Three parameters considered for reproduction are: elitism, crossover, and mutation.

4.2.4.1 Elitism

Elitism is used to prevent losing the best solutions from each generation. At this step, a percentage of the nondominated solutions with the best aggregated fitness are selected to be part of the next generation. This percentage can be defined by the

experimenter, in this work is considered the 25% of elitism. Figure 30 shows the set of nondominated solutions and the part selected by elitism and crossover.

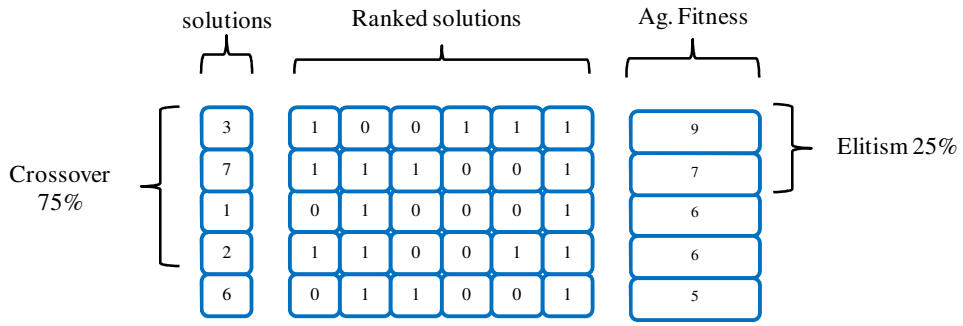

Figure 30: Solutions Selected for Elitism and Crossover

4.2.4.2 Crossover

This operator represents one of the most important parts of any evolutionary algorithm. Some of the most common operators used in EAs such as one-point or multiple points are likely to destroy the information obtained previously because of their random choices of crossover points [7-8]. Another important aspect of a crossover operator is the ability to create diversity of children in order to explore efficiently the search space. Therefore is important to select a crossover operator that can create good offspring and a good diversity for the network reliability problem. The crossover method used in this study due to the characteristics of the problem is the subsystem rotation crossover (SURC) presented in [42]. This method produces a larger number of children in the mating pool, providing a large number of diverse solutions to choose from.

What subsystem rotation crossover does is divide the chromosome in several subsystems. Then rotate the first subsystem in order to generate new solutions and

repeat the procedure until return to the initial accommodation and continue to the next subsystem and so on. Figure 31 shows the procedure.

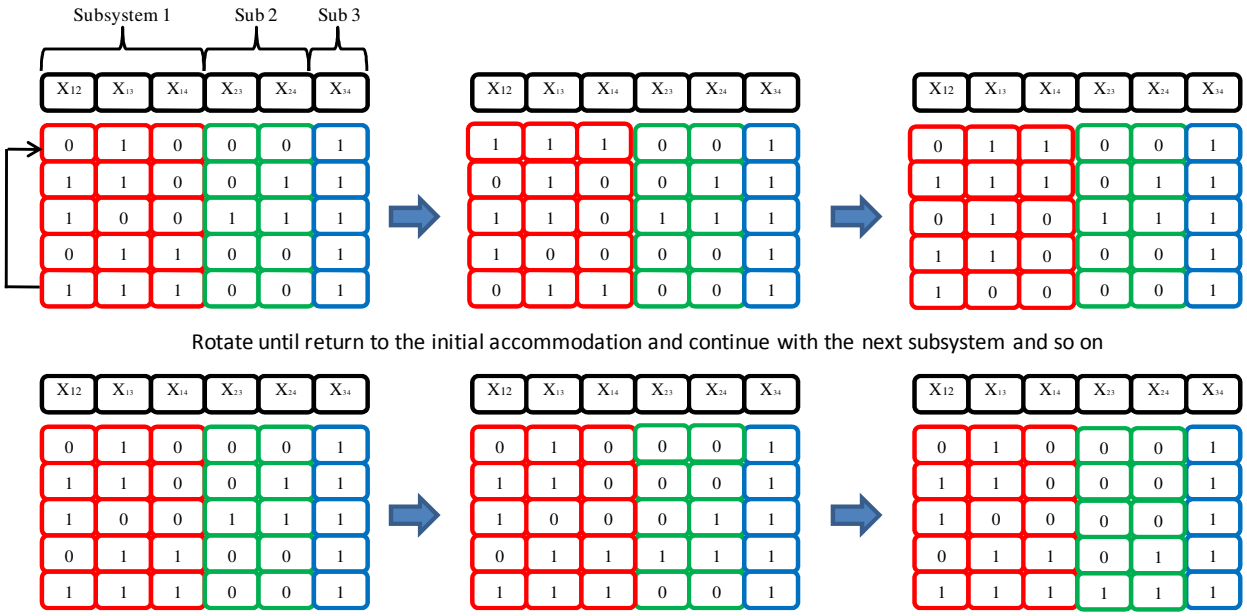

Figure 31: Subsystem Rotation Crossover

4.2.4.3 Mutation

The mutation selected for this algorithm is a two point mutation. The mutation probability selected was p_{mut}= 0.01. For each child, a random number is generated between 0 and 1, if this number is smaller than 0.01 then the child is selected to undergo mutation. Once the child is selected for mutation two random point in the chromosome are generated and the values are switched this can be observed in Figure 32.

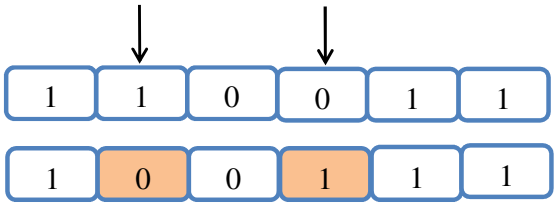

Figure 32: Two point mutation

4.2.4.4 New Population

The new generation is formed by elite parents and new children. 25% of the new population is form by the elite parents and 75% of children are selected randomly from the mating pool. Once the next generation is completed, the algorithm returns to the evaluation stage and repeat the procedure until the specified number of generations is reached. Figure 33 presents how the new generation is formed.

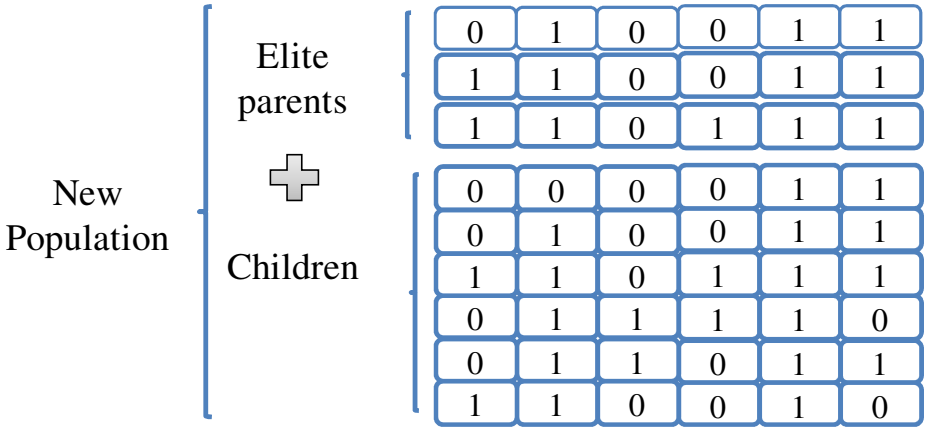

Figure 33: New population

After the specified number of generation is reached the algorithm stops and the nondominated solutions for the last generations represents the solution of the problem.

CHAPTER 5: Numerical Example

In the previous chapter the behaivor of the developed algorithm was exaplined. In the present chapter the algorithm was tested in an fiticius network in order to show the performance of the algorithm.

It is important at this point to recall what is the objective of this new multiple objective evolutionary algorithm . The reseach objective is finding the best network configuration or in other words detrmine which links need to be connected in the network in orther to optimize the following objectives:

- Maximize reliability $R(\mathbf{x})$
- Minimize total network cost $C(\mathbf{x})$
- Minimize weight $W(\mathbf{x})$

The algorithm was fully coded using MATLAB 2007b. The algorthm solved the network reliability problem for the three well known cases mentioned before (all-terminal, *k*-terminal, and two terminal) . The result for each case was a set of nondomitaed solutions.

The example consits in a network with 10 nodes. Those are all the parameters used in the algorithm

- Network with 10 nodes
- 25% elitism
- 75% crossover
- 1% mutation
- 100 of population size
- 100 generations
- Computer

- Core 2 duo E6550
- 2GB RAM

Table 8 presents da data for each link in the network

Table 8: Links Information

x	$\mathbf{x_{1,2}}$	$\mathbf{x_{1,3}}$	$\mathbf{x_{1,4}}$	$\mathbf{x_{1,5}}$	$\mathbf{x_{1,6}}$	$\mathbf{x_{1,7}}$	$\mathbf{x_{1,8}}$	$\mathbf{x_{1,9}}$	$\mathbf{x_{1,10}}$	$\mathbf{x_{2,3}}$	$\mathbf{x_{2,4}}$	$\mathbf{x_{2,5}}$
R(**x**)	0.90	0.85	0.95	0.80	0.99	0.79	0.80	0.92	0.83	0.87	0.99	0.97
C(**x**)	18	17	17	16	29	5	13	19	17	9	21	19
W(**x**)	0.30	0.43	0.20	0.39	0.10	0.60	0.70	0.60	0.40	0.53	0.30	0.40
x	$\mathbf{x_{2,6}}$	$\mathbf{x_{2,7}}$	$\mathbf{x_{2,8}}$	$\mathbf{x_{2,9}}$	$\mathbf{x_{2,10}}$	$\mathbf{x_{3,4}}$	$\mathbf{x_{3,5}}$	$\mathbf{x_{3,6}}$	$\mathbf{x_{3,7}}$	$\mathbf{x_{3,8}}$	$\mathbf{x_{3,9}}$	$\mathbf{x_{3,10}}$
R(**x**)	0.94	0.99	0.97	0.85	0.89	0.87	0.82	0.79	0.78	0.77	.95	0.82
C(**x**)	18	23	17	15	17	18	15	10	9	8	20	17
W(**x**)	0.52	0.30	0.60	0.10	0.45	0.39	0.42	0.51	0.53	0.60	0.27	0.52
x	$\mathbf{x_{4,5}}$	$\mathbf{x_{4,6}}$	$\mathbf{x_{4,7}}$	$\mathbf{x_{4,8}}$	$\mathbf{x_{4,9}}$	$\mathbf{x_{4,10}}$	$\mathbf{x_{5,6}}$	$\mathbf{x_{5,7}}$	$\mathbf{x_{5,8}}$	$\mathbf{x_{5,9}}$	$\mathbf{x_{5,10}}$	$\mathbf{x_{6,7}}$
R(**x**)	0.83	0.97	0.92	0.83	0.84	0.75	0.91	0.83	0.87	0.84	0.88	0.89
C(**x**)	13	25	22	14	13	11	21	17	13	14	19	17
W(**x**)	0.31	0.1	0.40	0.30	0.41	0.31	0.20	0.38	0.42	0.40	0.20	0.25
x	$\mathbf{x_{6,8}}$	$\mathbf{x_{6,9}}$	$\mathbf{x_{6,10}}$	$\mathbf{x_{7,8}}$	$\mathbf{x_{7,9}}$	$\mathbf{x_{7,10}}$	$\mathbf{x_{8,9}}$	$\mathbf{x_{8,10}}$	$\mathbf{x_{9,10}}$			
R(**x**)	0.93	0.97	0.74	0.99	0.97	0.91	0.80	0.81	0.87			
C(**x**)	20	25	13	29	25	20	17	12	13			
W(**x**)	0.19	0.30	0.30	.11	30	0.38	0.41	0.60	0.50			

The maximun number of links that this network can have is 45. Figure 34 shows a picture with a network with 10 nodes and all the links connected

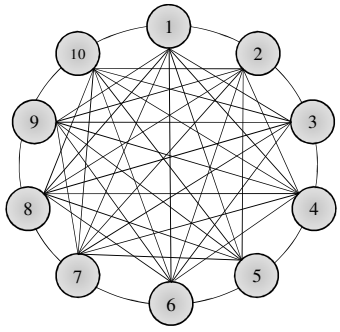

Figure 34: Fully Connected Network

5. 1 Two-Terminal network reliability

The two-terminal reliability is represented by the probability that two specific nodes in the network have connectivity or communication. In this specific case the nodes considered were node 1 and 2. After running the algorithm for 100 generation a set of no dominated solution was obtained. Figure 35 shows the results for the two-terminal case.

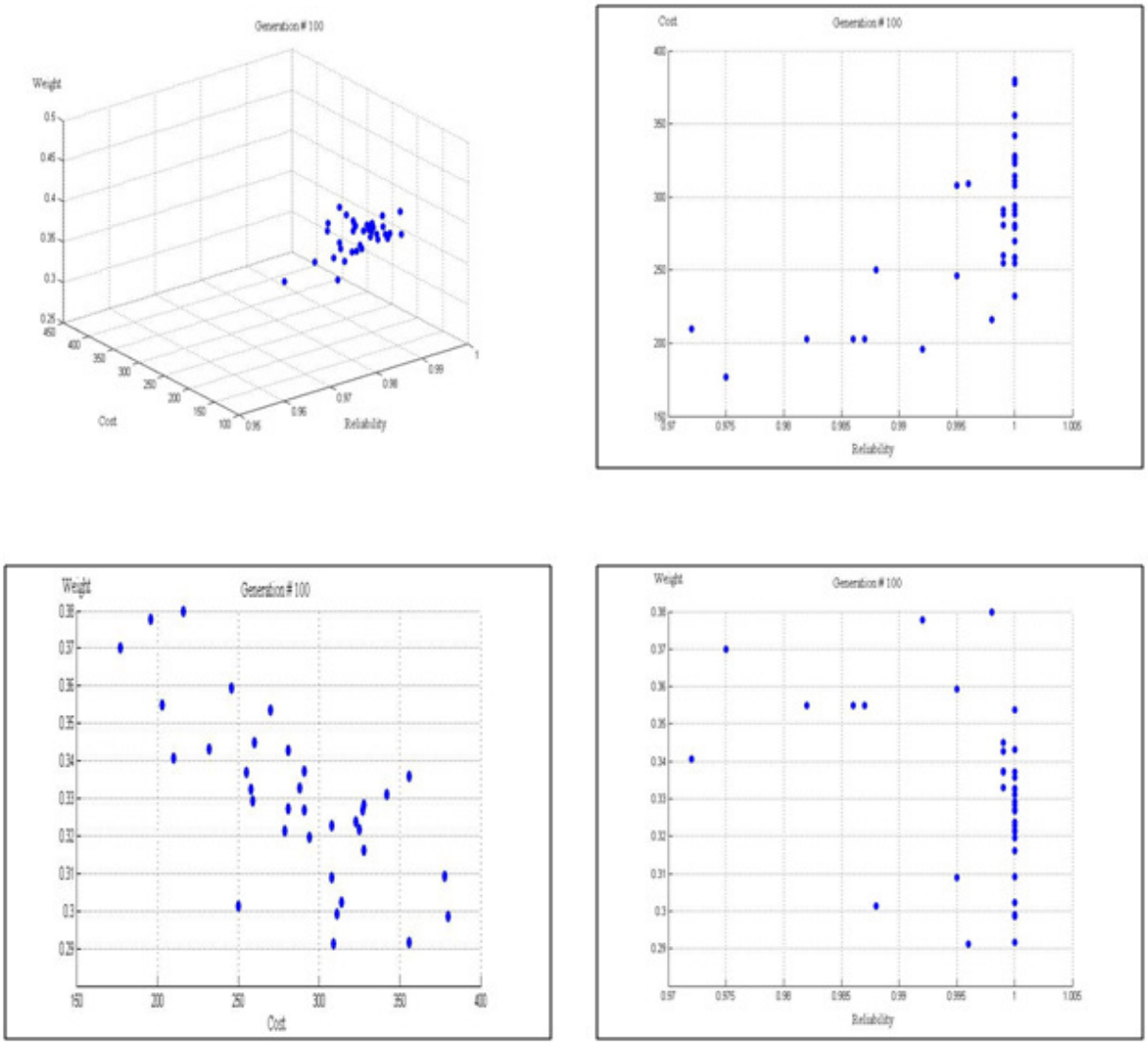

Figure 35: Two-Terminal Results

5.2 *k*-Terminal network reliability

The *k*-terminal reliability is represented by the probability of a specific set of nodes in the network have connectivity or communication. In this specific case the nodes considered were node 1, 3, 5, 7, and 9. After running the algorithm for 100 generation a set of no dominated solution for the *k*-terminal case was obtained. Figure 36 shows the results of this case.

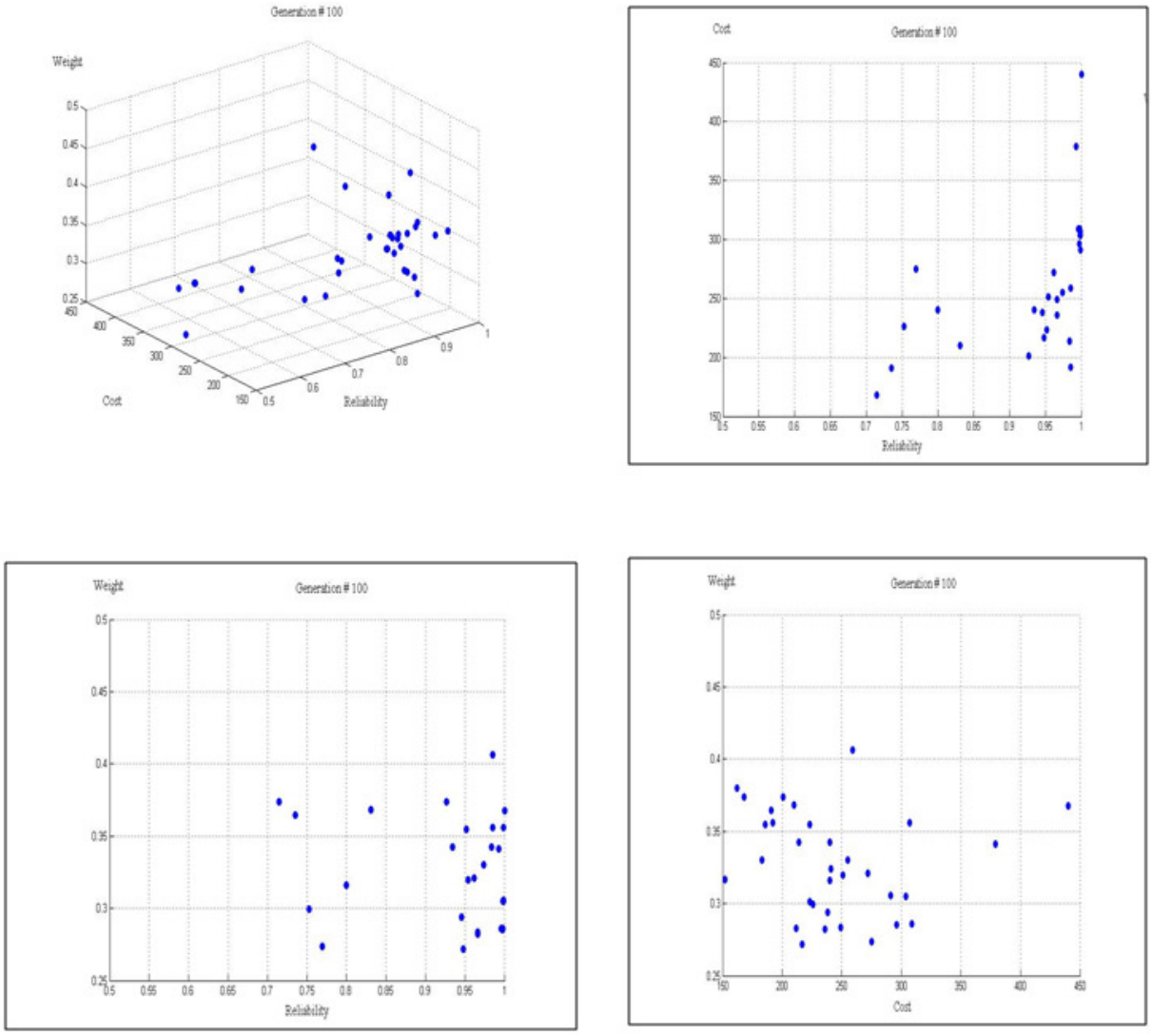

Figure 36: k-Terminal Results

5.3 All-Terminal network reliability

The all-terminal reliability is represented by the probability that each node in the network is connected to any other node in the network. In other words at least one spanning tree is formed. After running the algorithm for 100 generation a set of no dominated solution for the all-terminal case was obtained. Figure 37 shows the results of this case.

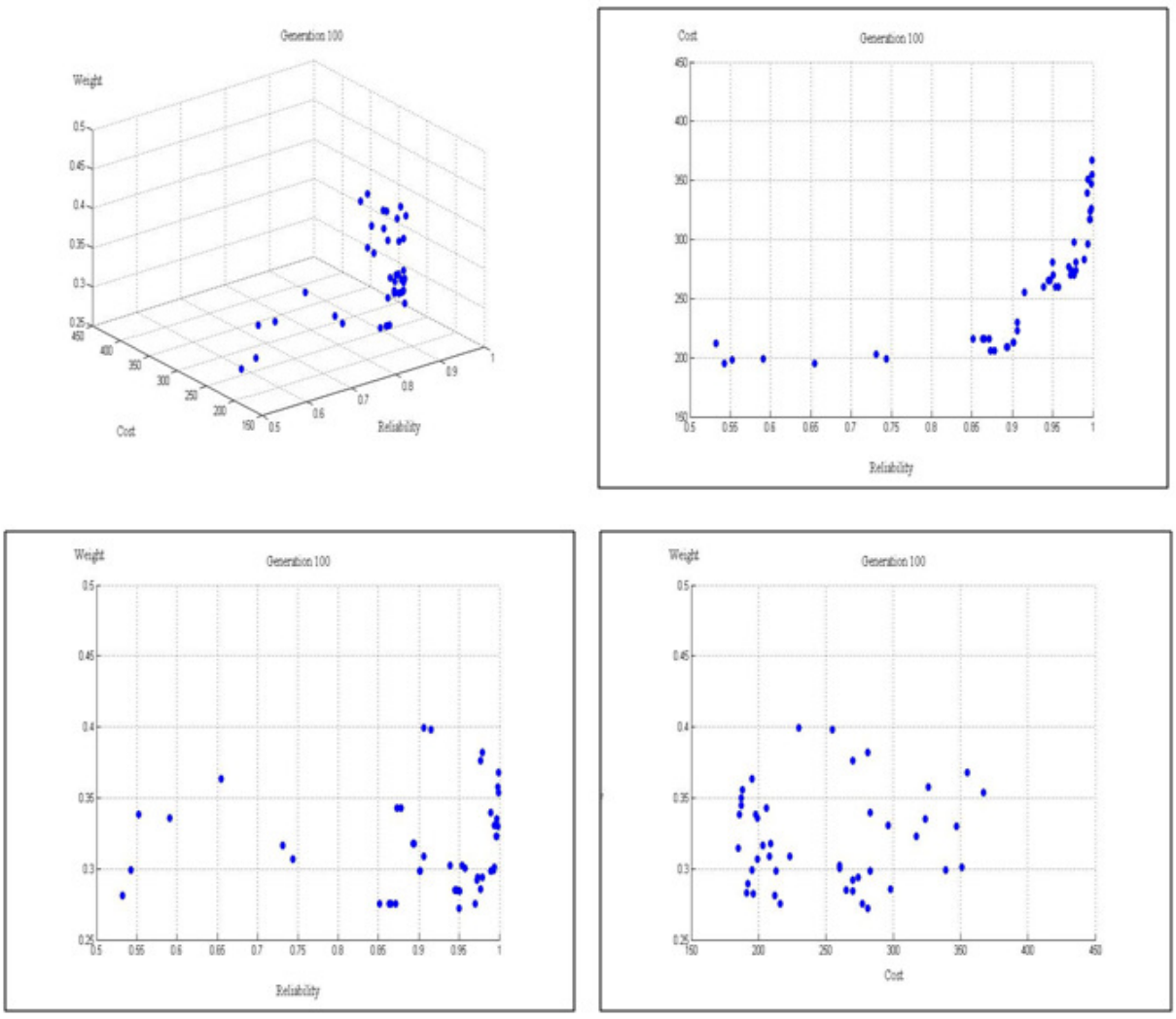

Figure 37:All-Terminal Results

The next step after obtain the Pareto set of nondominated solutions is to select one solution among the set obtained. This methodology is called post Pareto optimality analysis. There are many approaches proposed in literature. However, post Pareto optimality is not a part of the present work.

In order to select one solution let consider the all terminal case and select the solution that is closest to the ideal point. The ideal point in this case is [1,0,0] that represents 1 of reliability 0 of cost and 0 of weight. In order to select the solution closest to this point all the objectives were normalized and the Euclidian distance was used to select the closest solution to [1,0,0] Considering the all terminal case, table 9 shows the selected network configuration and figure 38 shows the representation of the same configuration. The objectives values for the selected solution are:

- **R(x)=.849**
- **C(x)=228**
- **W(x)=.283**

Table 9: Selected Network Configuration

$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$	$x_{1,6}$	$x_{1,7}$	$x_{1,8}$	$x_{1,9}$	$x_{1,10}$	$x_{2,3}$	$x_{2,4}$	$x_{2,5}$	$x_{2,6}$
0	0	0	0	1	0	0	0	0	1	1	0	0
$x_{2,7}$	$x_{2,8}$	$x_{2,9}$	$x_{2,10}$	$x_{3,4}$	$x_{3,5}$	$x_{3,6}$	$x_{3,7}$	$x_{3,8}$	$x_{3,9}$	$x_{3,10}$	$x_{4,5}$	$x_{4,6}$
0	0	1	0	0	0	0	0	1	1	0	1	0
$x_{4,7}$	$x_{4,8}$	$x_{4,9}$	$x_{4,10}$	$x_{5,6}$	$x_{5,7}$	$x_{5,8}$	$x_{5,9}$	$x_{5,10}$	$x_{6,7}$	$x_{6,8}$	$x_{6,9}$	$x_{6,10}$
0	1	0	0	1	1	0	0	1	1	1	1	1
$x_{7,8}$	$x_{7,9}$	$x_{7,10}$	$x_{8,9}$	$x_{8,10}$	$x_{9,10}$							
1	0	1	0	0	0							

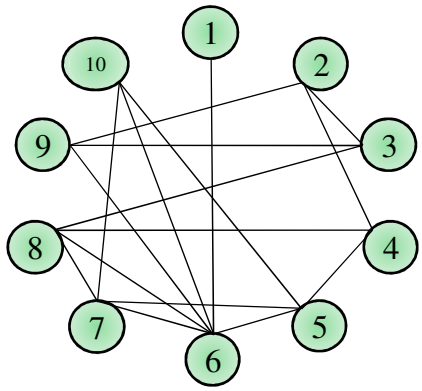

Figure 38: Network Representation of Selected Solution

CHAPTER 6: Conclusions & Future Research

After presented the problem to be solved in Chapter 1, reviewing multiple objective optimization and network theory in Chapter 2 and 3. A new method was proposed to solve the network reliability problem in chapter 4. In order to show how the algorithm works an example was presented for the three cases of the network reliability problem in chapter 5. Now is time to present some conclusions and comments about the work presented.

This work introduced a new multiple objective evolutionary algorithm to solve the network reliability problem considering the maximization of the network reliability, minimization of the network cost, and the minimization of the weight as objectives to be optimized simulateusly. Optimizing is understood as to be as good as possible. However, the presented multiple objective optimization problem involves several conflictive objectives that make optimizing them simultaneously hard and complicated.

The developed algorithm is based in evolutionary computation that mimics how biological evolution selects the better adapted individuals to generate new individuals to be part of future generation. Then the algorithm continues to generate new populations of solutions until the stopping criteria is reached that can be a specific number of generation or until the algorithm converges.

The new MOEA developed is flexible enough to be applied to the three cases of the network reliability problem obtaining good solutions of each case. As any heuristic search method it is impossible to guarantee a global optimal solution. Instead heuristic methods obtain very good approximation. The proximity of the solutions obtained to the true Pareto front can be evaluated obtaining all the possible solution and obtaining the

true Pareto front of solutions. This method is impractical because evaluate all the possible solutions of the problem is very complex and almost impossible. Other alternative is running the proposed algorithm many times and obtain a pseudo Pareto front of solutions.

The presented algorithm obtain as a solution a set of non dominated solutions, all the solutions of the set are considered optimal. Then the decision maker has to select one solution among those solutions. However, the Pareto-optimal set is often large and cumbersome, making the post-Pareto analysis phase complex. This problem has been studied in literature and several approaches have been proposed. Taboada [57] propose two different methods to intelligently filter or reduce the size of the Pareto-optimal set. One is pruning using non-numerical objective function ranking preferences. The second method involves pruning by using data clustering. Zhang [58] also work in this problem he proposed a min–max method with adaptive weightings. He based his work in methods based in linear programming and weighted sum methods proposed by Koski and Lin. Das & Dennis proposed an approach to problem in the same way that Zhang introducing the normal-boundary intersection method. Other approaches have been presented by Grierson [60] and Utyuzhnikov [61]. Post Pareto Optimality represents a very important part of any multiple objective optimization problem because is in this part where the real solution is obtained. However, Post Paero optimally analysis is not a part of this thesis and due to the importance of this analysis will be considered as a future research.

The presented work considers several assumptions in order to simplify the problem. One of these assumptions considers that the links of the network are operational or failed. This situation does not happen in reality. In real life link has a level of

degradation. This issue increases the complexity of network reliability evaluation. Yi-Kuei Lin [62-63] proposed a method using minimal paths to evaluate the reliability of the network considering multistate links and nodes, focusing in the two-terminal case.

Summarizing, A new multiple objective evolutionary algorithm was developed to solve the telecommunication design network problem considering three different objectives to be optimized simultaneously. The results obtained for: the all terminal, *k*-terminal, and two-terminal cases shows the algorithm performs well in the three cases. In order to expand this work multi state links and post Pareto analysis will considered as a future research.

REFERENCES

1. Kaisa, M. ,"Nonlinear Multiobjective Optimization"
2. Branke, J.; Deb, K.; Miettinen, K.; Slowinski, R. (Eds.) "Multiobjective Optimization Interactive and Evolutionary Approaches", Spinger
3. Lee, S., Goal Programing for Decision Analysis, Auerbach Publishers Inc.
4. Hajji, O., Brisset, S. and Brochet, P., "A New Tabu Search Method for Optimization With Continuous Parameters" IEEE Transactions on Magnetics, 40,2,1184-1187
5. Maged, E., Chang, L-M. and Zhang, L., 2005, "Utility-Function Model for Engineering Performance Assessment," J. Constr. Engrg. And Mgmt, (131), 558-568.
6. Laarhoven, P. & Aarts, E., 1987,"Simulated Annealing: Theory and Applications" ,Kluwer Academic Publishers.
7. Li,F.,Liu, Q.,Min, F., and Yang, G, 2005, "A New Crossover Operator Based on the Rough Set Theory for Genetic Algorithms", Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, 18-21.
8. Zhang, M., Gao, X. and Lou, W., 2007, "A New Crossover Operator in Genetic Programing for Object Clasification," IEEE Transactions on systems, Man, and Cybernetics, 37,5,1332-1343
9. M. Dorigo, V. Maniezzo, and A. Colorni., 1996 "Ant System: Optimization by a colony of cooperating agents," IEEE Transactions on Systems, Man, and Cybernetics, 26, 1, 29-41.
10. Hertz, A., Taillard, E., and Werra, D., "A Tutorial on Tabu Search," Department of mathematics University of Montreal.

11. Mandl, C., 1979, “Applied Network Optimization”, Academic Press Inc.,(2),7-47.
12. Price,W,1971, “Graphs and Networks An Introduction,” Butterworth & Co.,9-44.
13. Jungnickel, D.,2005,”Graphs, Networks and Algorithms,”Springer,(2),36-39
14. Swamy, M. & Thulasiraman, K.,1981,”Graphs, Networks, and Algorithms,” John Wiley & Son, (2),32-51.
15. West, D.,1996, “Introduction to Graph Theory,” Prentice Hall,(2),51-85.
16. Cormen,T., Leiserson, C., Rivest, R. and Clifford, 2001,” Introduction to Algorithms, Second Edition” The MIT Press,VII , 1000-1020.
17. Hsu, L-H. & Lin C-K.,2009, “Graph Theory and Interconnected Networks,” Taylor & Francis Group,LLC,(4), 61-78.
18. Morreale, P. & Terplan Kornel, 2001, “The handbook of modern telecommunications,” CRC Press, (1) 1-30,(3), 2-26.
19. Freeman R.,2004, “Telecommunication System Engineering,” Wiley-Intercience,(1), 1-34.
20. Sharma, R., Sousa, P., and Ingle, A.,1982, “Network Systems Modeling, Analysis and Design,” Van Nostrand Reinhold Data Processing Series. 3-50.
21. Anttalainen, T.,2003, “Introduction to Telecomunications Network Engineering,” Artech House, 1-75.
22. Sevtap, A.,Semih, M., 1999, “Reliability of lifeline networks under seismic hazard,” Reliability Engineering & System Safety, 65(3), 213-227
23. Rocco,C. and Zio, E., 2005, “Solving advanced network reliability problems by means of cellular automata and Monte Carlo sampling,” Reliability Engineering & System Safety, 86(2), 219-226.

24. Bazlamacci, C. and Hindi, K., 2001, “Minimum-weight spanning tree algorithms A survey and empirical study,” Computer & Operations Research, 28, 767-785.
25. Hardy, G., Lucet, C. and Limnios, N., 2007, “K-Terminal Network Reliability Measures With Binary Decision Diagrams,” Transactions on reliability, 56(3), 506-515.
26. Ghasemzadeh, M., Meinel,C. and Khanji, S., 2008, “K-terminal Network Reliability Evaluation Using Binary Decision Diagram,” Information and Communication Technologies: From Theory to Applications, 7(11),1-5.
27. Jane, C., Shen, W. and Laih, Y.,2008, “Practical sequential bounds for approximating two-terminal reliability,” European Journal of Operation Research, 195,427-441.
28. Cook, J. and Ramirez-Marque, J., 2006, “ Two-terminal reliability analyses for a mobile ad hoc wireless netwok,” Reliability Engineering and System Safety, 92, 821-829.
29. Kavian Y. & Naderi, M., 2006, “An Evolutionary Approach to Design WDM Telecommunication Survivable Networks,” IEEE International Conference on Engineering of Intelligent Systems, 1-6.
30. Srivaree-ratana, C., Konak,H. and Smith, A.E., 2002, “Estimation of all-terminal network reliability using an artificial neural network,” Computer & Operation Research, 29(7), 849-868.
31. Lin,H-Y, Kuo,S-Y and Yen, F-Y., 2003,”Minimal Cut set Enumeration and Network Reliability Evaluation by Recursive Merge and BDD,” Proceedings of the Eight IEEE International Symposium on Computers and Communication, 2, 1341-1346

32. Hui, K.-P., Bean, N.G., Kraetzl, M. and Kroese, D., 2003, “Network Reliability Estimation Using The Tree Cut and Merge Algorithm with Importance Sampling,” Design of Reliable Communication Networks, Banff, Alberta, Canada, October 2003, 254-262.
33. Ramirez-Marquez, J. and Rocco, C., 2008, “All terminal network reliability optimization via probabilistic solution discovery,” Reliability Engineering & System Safety, 93 (11), 1689-1697.
34. Dengiz B., Altiparmak F. and Smith A., 1997, “Local search genetic algorithm for optimal design of reliable networks,” IEEE Transactions on Evolutionary Computation, 1,179-88.
35. He, Fangguo & Qi, Huan, 2008, “A method of estimating network reliability using an artificial neural network,” IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 2, 57-60
36. Xing, L, 2008, “An Efficient Binary-Decision-Diagram-Based approach for Network Reliability and Sensitivity Analysis” IEEE Transactions on Systems, MAN, and Cybernetics, 38(1), 105-115.
37. Maaranen, H., Miettinen, K., and Makela, M., 2004, :Quasi-random initial population for genetic algorithms,” Department of Mathematical Information Technology, University of Jyväskylä
38. Karci, A.,2004, “Novelty in the Generation of Initial Population for Genetic Algorithms,” Knowledge-Based Intelligent Information and Engineering Systems,3214
39. Park, B., Choi, H., Kim, H.,2003,” A hybrid genetic algorithm for the job shop scheduling problems,” Computers & Industrial Engineering, 45(4), 597-613.

40. Atiqullah, M. and Rao, S., 1993, “Reliability optimization of communication networks using simulated annealing,” Microelectronics and Reliability, 33, 1303–1319.
41. Satish-Kamat, J. and Riley, M., 1975,” Determination of reliability using event based Monte Carlo simulation,” IEEE Transactions on Reliability, 4, 73–75
42. Taboada, H. and Coit, D., 2008, “MOEA-DAP: a New Multiple objective Evolutionary Algorithm for solving Design Allocation problems.”
43. Sano, Y. & Kita, H., 2000, “Optimization of Noisy Fitness Functions by Means of Genetic Algorithms Using History of search,” Parallel Problem Solving from Nature PPSN VI, 571-580.
44. Kuncheva, L.,1997, ” Fitness functions in editing k-NN reference set by genetic algorithms,” Pattern Recognition,30(6), 1041-1049
45. Rockafellar, R. T., 1984 “Network Flows and Monotonic Optimization”, John Wiley & Sons.
46. Georgans, N.,1980, “Modeling and analysis of message switched computer-communication networks with multilevel flow control,” Computer Networks, 4, 285-294
47. Wada, H. & Fukushima, H., 1996, “Mobile computing on wireless telecommunication networks,” Universal Personal Communications, 1996. Record., 1996 5th IEEE International Conference on, 2, 778-782
48. Hayashi, M.; Abe, T., 2008, “Evaluating Reliability of Telecommunications Networks Using Traffic Path Information,” IEEE Transactions on Reliability, 57(2),283-294.

49. Malheiros, D. & Silva, J., 1998, “Modeling a telecommunication network for fault management applications” Network Operations and Management Symposium, 1998. NOMS 98., IEEE, 723-732.
50. Dengiz, B & Altiparmak, F.& Belgin, O. (2007) A hybrid ant colony optimization approach for the design of reliable networks. IEEE Congress on Evolutionary Computation 25-28(1118 – 1125)
51. Azaron,A & Perkgoz,C & Sakawa, M. (2005) A genetic algorithm approach for the time-cost trade-off in PERT networks. Applied Mathematics and Computation.168(1317-1339)
52. Ho,W. & Emrouznejad, A. (2009) Multi-criteria logistics distribution network design using SAS/OR. Expert Systems with Applications. 36(7288-7298)
53. Kaisa Miettinen ,”Nonlinear Multi-objective Optimization”, Kluwer’s INTERNATIO-AL SERIES
54. Zitzler, E. and Thiele, L., 1999, “Multi-objective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach,” IEEE Transactions on Evolutionary Computation, 3(4) 257-271
55. Knowles, J. D. & Corne, D. W., 2000, “Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy, “Evolutionary Computation, 8(2) ,149-172.
56. Deb, K., Agrawal, S., Pratap, A. and Meyarivan, T., 2002, “A fast and elitist multi-objective genetic algorithm: NSGA-II,” IEEE Transactions on Evolutionary Computation, 6(2), 182-197
57. Taboada, H. A. and Coit, D. W. 2005, “Post-Pareto optimality analysis to efficiently identify promising solutions for multi-objective problems “

58. Zhang, W. & Gao, T., 2006, “A min–max method with adaptive weightings for uniformly spaced Pareto optimum points,” Computers and structures, 84,1760-1769

59. Das, I. &Dennis, J., 2006 “Normal-boundary intersection: a new method for generating the Pareto surface in non linear multicriteria optimization problems, Sino-French Laboratory of Concurrent Engineering, 8(3),631–657.

60. Grierson, D., 2008, “Pareto multi-criteria decision making” Advanced Engineering Informatics, 22, 371-384.

61. Utyuzhnikov, S.,Fatini, P., and Guenov, M., 2009, “A method for generating a well-distributed Pareto set in nonlinear multi-objective optimization,” Journal of Computational and Applied Mathematics, 223, 820-841.

62. Lin, Y-K, 2001, “ A simple algorithm for reliability evaluation of a stochastic-flow network with node failure,” Computers & Operatons Research, 28, 1277-1285

63. Lin, Y-K, 2001,” Two-commodity reliability evaluation of a stochastic-flow network with varying capacity weight in terms of minimal paths

CURRICULUM VITA

Oswaldo was born in Chihuahua, Chih., Mexico. The first of three offspring of Ruben Aguirre and Teresa Ortega. He received his bachelor degree in the Chihuahua Institute of Technology in fall 2005. Then he started his professional career as a engineer in the industrial Engineering department in a wheel manufacturing company called Superior in Chihuahua City. Later, he entered The University of Texas at El Paso in fall 2007. While pursuing a master's degree in industrial engineering, he worked with Dr. Heidi Taboada as his Research Assistant. He had the opportunity to present his research about data survivability in the 2009 UTEP SACNAS Expo Research, in El Paso Texas, in the 2009 INFORMS Western Regional Conference in Tempe, Arizona, and also in the 2009 INFORMS National Conference in San Diego, California. He also had the opportunity to present his research in network optimization in the 2009 IIE Annual Conference and Expo in Miami, Fl. and in 2009 15th ISSAT International Conference in San Francisco CA.