A COMPARATIVE STUDY OF EVOLUTIONARY NETWORK DESIGN

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

SİNAN KALKAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF COMPUTER ENGINEERING

DECEMBER 2003

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Ayşe Kiper
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Göktürk Üçoluk
Co-Supervisor

Dr. Onur T. Şehitoğlu
Supervisor

Examining Committee Members

Assoc. Prof. Dr. Hakkı Toroslu

Assoc. Prof. Dr. Göktürk Üçoluk

Asst. Prof. Dr. Y. Murat Erten

Dr. Cevat Şener

Dr. Onur T. Şehitoğlu

ABSTRACT

A COMPARATIVE STUDY OF EVOLUTIONARY NETWORK DESIGN

KALKAN, SİNAN

M.Sc., Department of Computer Engineering

Supervisor: Dr. Onur T. Şehitoğlu

Co-Supervisor: Assoc. Prof. Dr. Göktürk Üçoluk

DECEMBER 2003, 41 pages

In network design, a communication network is optimized for a given set of parameters like cost, reliability and delay. This study analyzes network design problem using Genetic Algorithms in detail and makes comparison of different approaches and representations.

Encoding of a problem is one of the most crucial design choices in Genetic Algorithms. For network design problem, this study compares adjacency matrix representation with list of edges representation. Also, another problem is defining a fair fitness function that will not favor one optimization parameter to the other. Multi-objective optimization is a recommended solution for such problems. This study describes and compares some of those approaches for different combinations in network design problem.

Keywords: genetic algorithms, multiobjective optimization, network design, network representations, adjacency representation, edge representation

ÖZ

EVRİMSEL AĞ TASARIMI PROBLEMİNİN KARŞ ILAŞ TIRMALI Ç ALIŞ MASI

KALKAN, SİNAN

Y¨uksek Lisans, Bilgisayar M¨uhendisliği B¨ol¨um¨u

Tez Y¨oneticisi: Dr. Onur T. Ş ehito˜glu

Ortak Tez Y¨oneticisi: Assoc. Prof. Dr. G¨okt¨urkÜç oluk

ARALIK 2003, 41 sayfa

Bir ağ tasarımı probleminde iletiş im ağı, fiyat, sağlamlık ve gecikme gibi verilen parametrelere g¨ore optimize edilir. Bu ç alış ma, ağ tasarımı problemini Genetik Algoritmalar kullanarak detaylı bir ş ekilde inceler ve farklı yaklaş ımları ve g¨osterimleri karş ılaş tırır.

Bir problemin g¨osterimi, Genetik Algoritmalar iç in en ¨onemli tasarım seç eneklerinden biridir. Bu ç alış ma, ağ tasarımı problemi iç in, bitiş iklik matrisi g¨osterimini bağlantı listesi g¨osterimi ile karş ılaş tırmaktır. Ayrıca, diğer bir sorun, bir optimizasyon parametresini bir diğerine g¨ore kayırmayacak adil bir değer fonksiyonu tanımlamaktır. Ç ok amaç lı optimizasyon, bu problemler iç in ¨onerilen bir ç ¨oz¨umd¨ur. Bu ç alış ma, ağ tasarım probleminde, farklı kombinasyonlar iç in, bu yaklaş ımlardan bazılarını tarif eder ve karş ılaş tırır.

Anahtar Kelimeler: genetik algoritmalar, ç ok amaç lı optimizasyon, ağ tasarımı, ağ g¨osterimi, bitiş iklik matrisi g¨osterimi, bağlantı g¨osterimi

To my life..

ACKNOWLEDGMENTS

I would like to thank Onur Tolga ŞEHİTOĞLU for his precious supervision and other members of the GA group of the Department (namely, Göktürk ÜÇOLUK and Erkan KORKMAZ) for their fruitful discussions. You were very important for the very early steps of my academic life.

That was the support of my friends that made me not to give up at the difficult times.

Without love and patience of my family, I wouldn't do MSc, and this study would not exist at all.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Design of communication networks has been very important for users due to high performance demanding characteristics of inter and intra net traffics. The design phase involves the optimization of several parameters under some pre-specified constraints, and this process is known to be an NP-complete problem [13]. If the network under consideration has n different computers (or nodes) and p different possible bandwidths, then the size of the space of potential topologies would be $(p+1)^{(n(n-1)/2)}$ [1]. For the values $n = 10$ and $p = 3$, the size of the search space would be 1.2×10^{27}. Even for this small problem, search space is very huge, which makes the problem intractable for search strategies that use basic enumeration techniques.

Both non-evolutionary and evolutionary methods have been attempted in the literature. Non-evolutionary studies include [16, 30, 17, 32, 10, 21]. These are mainly variations of the branch-and-bound algorithm, which tries to get the optimal solution by adding and/or deleting some edges. Naive search techniquies such as simulated annealing and tabu search have also been adapted in network optimization [3, 28, 18, 4]. Hybrids of evolutionary and non-evolutionary approaches are also studied. Such methods mainly use non-evolutionary approaches as a sub-step of genetic algorithms [26, 29].

The encoding of a problem in a chromosome is critical for the performance of an evolutionary algorithm. Due to the huge search space of the network design problem, this importance even increases and becomes one of the key aspects of the overall performance of the ge-

[1] The term $(p+1)$ includes an exra bandwidth because two nodes in the network may not have a link in between

netic algorithm. Nevertheless, existing genetic network design studies have not investigated alternative representations much, and mainly variations of adjacency matrix representation have been exercised [12, 31, 24, 22, 1, 25, 11, 13, 15, 14, 2, 27]. The variations include appending routing tables to the chromosome [31], storing link types for existing links [12], etc. An alternative representation, list of edges representation (also called edge representation), is used in a few works [15, 5]. [5] is known to be the first study to use edge representation in a network design problem.

In [15], a GA that used edge representation with repair of unreliable networks has been implemented and it is compared with adjacency matrix representation that didn't have repair mechanisms for unreliable networks. [5] implemented edge representation as tuples and compared it with non-evolutionary approaches. They also applied repair mechanisms for unreliable networks.

Network design by definition is a multi-objective optimization problem because it involves optimization of several objectives. However, evolutionary approaches in the literature have dealt with the problem mostly in the form of aggregate optimization [11, 13, 31, 27, 1]. The multi-objective nature of the problem is not adequately investigated although some initial attempts exist in the area [24, 25, 26, 22].

In summary, this study makes the following contributions:

- Analysis of an alternative representation for reliable network design, i.e. list of edges representation.
- Systematic comparison of two representations for reliable network design; namely, adjacency representation and list of edges representation.
- A multi-objective implementation of network design problem.
- Analysis of multi-objective network design with respect to its aggregate counter-part.

This thesis is organized as follows. In chapter 2, background, namely, basics of Genetic Algorithms, multiobjective optimization and evolutionary network design are presented. Chapter 3 analyzes two different network representations. Chapter 4 presents and analyzes a multi-objective algorithm for evolutionary network design. Finally, chapter 5 concludes the thesis with discussions and possible future directions.

CHAPTER 2

BACKGROUND

2.1 GENETIC ALGORITHMS

Genetic Algorithms (GAs) are search algorithms that mimic natural evolutionary principles to constitute search and optimization procedures. They use the mechanisms of the nature such as mating, mutation and selection of the fittest individuals for finding solutions of optimization problems. They were first developed for continuous non-linear optimization and later extended for combinatorial problems. Now, they are popular search mechanisms that perform better than their non-evolutionary counterparts due to their balanced combination of exploitation and exploration.

GAs were pioneered by Holland [20] and Goldberg [19]. Since then, there have been many attempts to understand and outline the principles behind GAs to be able to use them more effectively in the optimization and search problems.

A GA system holds a set of individuals (also called *chromosomes*), called the *population*. At each step of the evolution, some subset of individuals are mated (also called *crossover*), then some subset of individuals are mutated, and after these steps, some of them are selected to survive and give their genes to the next generation. In order to guide the search in the direction that leads to optimal solutions, the selection process favors fitter individuals, mimicing survival of the fittest phenomenon of the nature.

As the search space is represented by the individuals, effective encoding of the problem to be optimized is critical for the success of GAs. A good encoding mechanism should include all aspects of the domain that are relevant to the optimization problem.

A GA starts with an initial population, which is usually randomly constructed in practice. The size and the distribution of the individuals of the initial population are very important for the rest of the search.

The next step of GA assigns each individual a value which indicates its quality as a solution to the optimization problem at hand. This value is also called the *fitness* of the individual.

Then, some subset of the population is selected for crossover. The individuals are selected with a probability distribution that favors individuals with higher fitness values. So, higher the fitness values, higher the chances of giving the genes to the next generation in the crossover step.

Crossover mates selected individuals from the previous step. The mating determines which parent contributes which bits of the child. There are several approaches to this process. Some of the mostly used crossover methods include:

- One-point crossover. Parents are divided into two at some random point, and the first part of one parent is postfixed by the second part of the other parent.
- Multi-point crossover. Multiple random points are used instead of one. Corresponding parts of parents are swapped to construct the child.
- Uniform crossover. Uniform crossover combines bits sampled uniformly from the two parents. The crossover mask is generated as a random bit string with each bit chosen at random from either parent and independent of other bits.

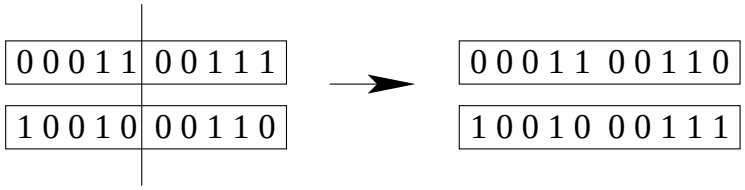


Figure 2.1: One-point crossover.

As the last step, random minor modifications on individuals are perfomed. This step is called *mutation* and simulates unexpected change phenomenon of the nature.

With the last step, a new population with new solutions is constructed.

The above steps are repeated either for a fixed number of iterations or till a pre-defined fitness value is reached.

In summary, a typical GA runs the following pseudo-code:

Figure 2.2: Mutation.

```
function GA
{
          t = 0
          initialize P(t) /* initial population */
          evaluate P(t) /* fitness values assigned */
          while ( not termination-condition )
          {
                    t = t + 1
                    select P(t) from P(t-1)
                    alter P(t) /* crossover and mutation */
                    evaluate P(t)
          }
}
```

2.2 MULTI-OBJECTIVE OPTIMIZATION

Multi-objective optimization (MOO) is an optimization problem which involves optimization of multiple objectives, or functions. In most cases, these functions are conflicting with each other.

A MOO problem is formally defined as:

minimize/maximize	$f_m(x)$,	$m = 1, 2, ..., M;$
subject to	$g_j(x) \geq 0,$	$j = 1, 2, ..., J;$
	$h_k(x) = 0,$	$k = 1, 2, ..., K;$
	$x_i^{(L)} \leq x_i \leq x_i^{(U)},$	$i = 1, 2, ..., n;$

where x is a solution representing a vector of n decision variables: $x = (x_1, x_2, ..., x_n)^T$; $f_m(x)$ are the objective functions; $g_j(x) \geq 0$ and $h_k(x) = 0$ are the inequality and equality constraints, respectively; and, the last set of constraints gives the bounds of the decision variables.

A MOO problem does not have a global optimum in the case of conflicting objectives. Rather, there are a set of solutions, called pareto optimal solutions. A set of solutions are

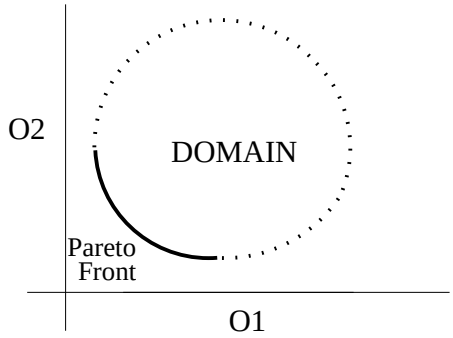


Figure 2.3: Pareto front for two objectives O1 and O2 which are to be minimized.

pareto optimal if they are not dominated by any other solution in the search space. A solution y dominates x if for all i, $y_i \geq x_i$, and there exists some i such that $y_i > x_i$, where $i = 1, 2, ...n$.

The set of non-dominated solutions forms a line (or a surface in the case of 3 or more objectives) called Pareto Front in the solutions domain of the problem. The aim of a multi-objective algorithm is basically to move the population towards to the Pareto Front. Figure 2.3 gives an example of Pareto front for an imaginary problem whose two objectives have to be minimized (If both objectives were to be maximized, Pareto front would be on the upper right corner of the domain. If O1 were to be minimized and O2 were to be maximized, Pareto front would be upper left corner of the domain. If O1 were to be maximized and O2 were to be minimized, Pareto front would be lower right corner of the domain.).

The simplest approach to a MOO problem is to construct a composite function out of given objective functions (i.e. $F(x) = w_1 * f_1(x) + w_2 * f_2(x) + ... + w_M * f_M(x)$). The construction process requires assignment of weights to each objective function. If the weights are carefully chosen, this approach may reach to a solution. However, it is highly sensitive to the weights chosen, and it converges to only one local optimum. To be able to find multiple solutions and eliminate the bias of the weights, multiple iterations may be run with different weight vectors. However, it is very difficult to find a weight vector in most of the problems. So, a non-subjective method should be preferred.

2.2.1 MULTI-OBJECTIVE EVOLUTIONARY OPTIMIZATION

Evolutionary MOO approaches could be divided into three categories [6]:

- **Approaches that use aggregating functions.** A composite objective function is formed using summation, multiplication, division etc. They are very simple to implement; however, they are unable to yield multiple solutions, and it is very difficult to find the right combination in most of the cases.

- **Approaches that use pareto optimality.** The individuals of the population are sorted according to the number of individuals that dominate them, and the fitness values are assigned using this ranking (proposed by [19]). Some of the algorithms under this category are Multi-Objective Genetic Algorithm (MOGA), Non-dominated Sorting Genetic Algorithm and Niched Pareto Genetic Algorithm.

- **Other approaches.** They are mainly based on population policies or special handling of objective functions. The following can be listed under this category: Vector Evaluated Genetic Algorithm, Lexiographic Ordering, Approaches that Use Game Theory, Using Gender to Identify Objectives, Weighted Min-max Approach.

2.3 EVOLUTIONARY NETWORK DESIGN

Network design is a combinatorial optimization problem involving optimization of several objectives such as cost, average delay and reliability of the network. The search space of the problem is huge even for small number of computers and is intractable for non-evolutionary approaches.

A network is basically a graph G of a set of nodes N and a set of links L. The links denote the connections between the nodes and are assumed to be bi-directional. Networks could be of two kinds: Backbone Networks and Local Area Networks (LAN). LANs involve the end-users, and backbone networks are responsible for connecting LANs to each other. Network

design problem implicitly assumes backbone networks; however, the ideas are applicable to both types.

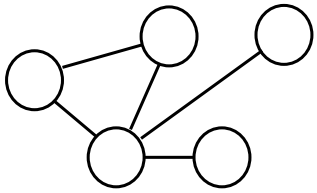


Figure 2.4: A typical network.

Formal statement of the problem as generally adopted in the literature [13] is as follows: Notation:

- N : Set of nodes (also called terminals, computers).
- L : Set of links (also called edges, connections, arcs).
- (i, j) : Link between nodes i and j.
- p : Link reliability of the network.
- x_{ij} : 1 if there is a link between nodes i and j; 0, otherwise.
- x : A topology of $x_{11}, x_{12}, ...x_{ij}, ..x_{NN-1}$.
- c_{ij} : Cost of link (i, j).

In order to simplify the problem, the following assumptions are usually made:

- The location of each node is given.
- The nodes are perfectly reliable.
- Capacity and reliability of each link are fixed and known.
- Each link is bi-directional.
- Only one link is possible between nodes i and j.
- The links are either operational or failed.
- The failure of links are independent.

- No repair for failures is considered.

With these assumptions, the network design problem may involve the optimization of the following objectives:

- Network cost. Total cost of the network is:

$$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} x_{ij}$$

- Average delay. Calculation of average delay is generally adapted from Queueing Theory:

$$\frac{1}{\gamma} \sum_{i,j=1}^{N} \frac{f_{ij}}{cap_{ij} - f_{ij}}$$

  where γ denotes the total traffic, f_{ij} the flow of link (i, j) and cap_{ij} is the capacity of link (i, j). An approximate average delay can also be used. This approximation uses average number of nodes (hops) on the path between nodes i and j.

- Routing. A route between two nodes is a path that connects these two nodes. Routing optimization involves minimization of routes between every node pair in the network. It tries to establish both effective use of bandwidths and a low average delay.

- Reliability. Reliability is the ability of the network to survive failures of components. Basically three approaches exist for enforcing reliability:

  - K-connectivity. K (link and/or node) disjoint paths between each pair of nodes should exist in the network for it to be k-connected. Generally, 2 and 3 link-disjoint-connected reliability are exercised [27, 1, 5, 22].

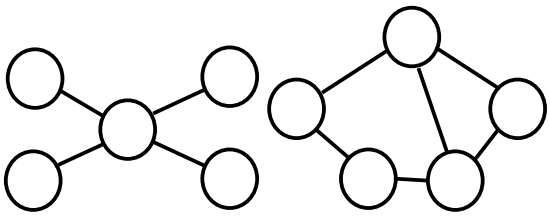


Figure 2.5: 1-connected and 2-connected networks.

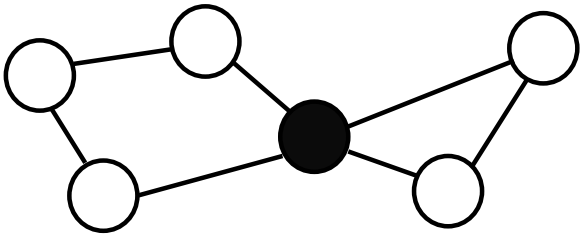


Figure 2.6: Articulation points. The black node is an articulation point. Because it divides the network into two when removed.

- Overall (All-terminal) Reliability. Overall reliability of a network is defined as the probability that every node can communicate with every other node. This probability is a measurable quantity [8]. However, calculation of this exact quantity is itself an NP-complete problem.
- Articulation Points. An articulation point is any node whose removal results in a disconnected network. It guarantees 2 node and link disjoint connectivity.

2.3.1 APPROACHES

2.3.1.1 NON-EVOLUTIONARY APPROACHES

Non-evolutionary algorithms can be summarized as follows:

- Branch X-Change (BXC) [16, 30]. The algorithm starts from an arbitrary configuration and tries to get to a local minimum by means of local transformations, which include adding new links, deleting some old links and correcting the resultant network (in the sense of connectivity). This algorithm exhaustively searches the search space, and it is not preferred.
- Concave Branch Elimination [17, 32]. The algorithm starts from fully connected network and eliminates uneconomical links until a local minimum is reached.
- Cut-Saturation [10]. The algorithm extends BXC by choosing economical transformations rather than trying all transformations.
- Simulated Annealing [3, 28].
- Tabu Search [18, 4].

Non-evolutionary approaches perform bad for large N, return only one solution, and are sensitive to local optima [15].

2.3.1.2 EVOLUTIONARY APPROACHES

There are two important points for evolutionary network design: Enforcement of network reliability and representation of a network in a chromosome.

As previously mentioned, network reliability can be imposed using three different ways (k-connectivity, exact calculation of reliability and articulation points). K-connectivity is used in [27, 1, 5, 22]. Studies using k-connectivity generally adopt a k-corrector process as an operator of GA. The job of this operator is to maintain k-connectivity of non-k-connected individuals by adding appropriate links to the network. Calculation of exact reliability for each individual in each generation is not practicle due to its computational complexity. There are appoaches that avoid this complexity by using upper and lower bounds of this exact quantity [9], estimations through Monte Carlo Simulation [33, 11] and neural networks [7]. [25] used articulation points to impose network reliability; however, he also assumes that computers of the network are reliable due to the requirements of the other parts of his algorithm.

- **Adjacency Matrix Representation and Variations.** For encoding a network in a chromosome, the simplest approach is to use the upper triangular of the adjacency matrix of the nodes of the network.

  For example, the representation of the network in figure 2.7 is as follows:

x_{12}	x_{13}	x_{14}	x_{15}	x_{23}	x_{24}	x_{25}	x_{34}	x_{35}	x_{45}
1	1	1	1	1	1	0	0	1	0

  In a chromosome that uses this representation, first bits are occupied by the initial rows of the adjacency matrix. These initial rows reserve more number of bits in the chromosome than other rows of the adjacency matrix, which brings out a bias (*initial-rows bias*) in which the bits of the initial rows have more chances of getting splitted by a crossover operation. The ideal case should give equal chances to all rows of the adjacency matrix.

  One variation of adjacency representation is to embed the types of the links in the representation. This approach stores an integer k which denotes the type of the link if

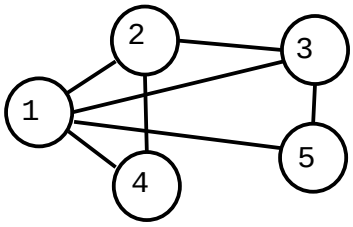


Figure 2.7: A network.

the link exists in the network; zero otherwise. [13] used bandwidth types of the links in the representation. Another variation stores routing table, link bandwidths and the adjacency matrix as three different segments of a chromosome [31].

Adjacency matrix representation seems to be weak because it is not likely to grow or preserve building blocks in the encoding. Besides, it introduces the initial-rows bias which causes the labelling and ordering of the nodes to have significance in the performance of the genetic algorithm.

- **Edge Representation.** An alternative representation, called edge representation, stores just the links that exist in the network, rather than enumerating all possible edges and marking those that exist in the network. Hence, it is a variable-length encoding. For example, the network in figure 2.7 can be represented using the following list (tuple (a, b) denotes an edge between node a and b):

  [(1,5), (2,3), (1,4), (1,2), (1,3), (3,5), (2,4)]

  Note that the numbers in each tuple are ordered but the edges themselves have no specific ordering.

  [15] assigned each edge a unique integer and stored these integers rather than node tuples. The study compared a GA that used adjacency matrix representation with another GA that used edge representation. Their study showed that the GA with edge representation is better than the one with adjacency matrix representation in terms of both speed and solution quality. However, as the aim was not to compare the performances of the representations, they used different parameters (the GA that used edge representation employed repair mechanisms while the GA that used adjacency matrix representation did not).

[5] compared a non-evolutionary algorithm and a GA which used edge representation with tuples.

One of the advantages of edge representation is that it does not have initial-rows bias; so, it gives equal chances to all rows of the adjacency matrix. Another advantage is that it allows building blocks as group of edges to grow.

Crossover and mutation operators of edge representation can produce duplicate edges in an individual. One of the important design issues in edge representation is what to do with these edges. There are three alternatives: To delete duplicate edges whenever a duplicate edge is created from any operation of GA; to keep duplicate edges as they are, but to give a penalty to the individual for each duplicate edge introduced; to keep duplicate edges as they are without giving any penalties.

2.3.1.3 CRITICS OF THE LITERATURE

Most of the studies make the assumption that nodes are perfectly reliable. This assumption is unrealistic because computers are vulnerable to failures as much as links are.

Network design, by definition, involves optimization of several objectives. However, the literature has approached the problem mostly in the form of aggregate optimization except for [24, 25, 26, 22, 23].

Another weakness of the existing studies comes from the representations used in GA. Except for two studies mentioned in previous section, they have used adjacency matrix representation which is vulnerable to initial-rows bias, and alternative representations have not been analyzed.

CHAPTER 3

NETWORK REPRESENTATIONS

As mentioned in section 2.3.1.2, there are several alternatives for representing a network in a chromosome. In this chapter, two of these representations are analyzed; namely, adjacency matrix representation (will be called adjacency representation, for simplicity) and edge representation. Adjacency representation suffers from initial-rows bias. Edge representation does not have this bias, and as possibly it can store useful edges in sequence, it may better represent a network structure.

This chapter compares these two representations whose implementational details are outlined in the next section.

3.1 METHODOLOGY

In previous studies, it has been tried to optimize several aspects of network design problem: reliability, delay, cost, routing, etc [1, 22, 27, 15]. In this chapter, we adopted the most usual setting which involves optimization of reliability and cost.

3.1.1 NETWORK DESIGN PARAMETERS

Reliability:

In this work, 2-connectivity is used as the reliability criteria of the network with giving penalties to non-2-connected individuals. However, in our experiments, no repair is considered in order to analyze the performances of the representations isolated from the effects of

repairing.

Cost:

The cost of a network is simply taken to be the sum of the costs of existing links in the network.

3.1.2 GENETIC ALGORITHM PARAMETERS

Initial Population:

The initial population in both representations are created randomly. To make the representations have the same initial distribution, edge representation is made to use the initial population creation scheme of adjacency representation.

Crossover:

Adjacency representation uses one-point crossover. The nature of edge representation offers several alternatives for crossover operation. Some of the alternatives that we have considered are:

- Choose two points in each individual and swap the post-segments.
- Choose one point along the smaller individual and swap post-segments around this point.
- Choose one point along the longer individual. If the point lies inside the smaller individual, swap post-segments around this point. Otherwise, append the post-segment of the longer individual to the end of the smaller individual.

The alternative chosen is very important for the performance as the lengths of both individuals are severely affected by this operation. We have used the third alternative because it turned out to give the best performance.

[5] used one-link crossover, which swaps two randomly selected edges from mating individuals. This alternative is weak because it only makes one change at a time, slowing down the search process in the huge search space of network design problem. [15] uses a form of uniform crossover. Both studies repair non-2-connected individuals.

Mutation:

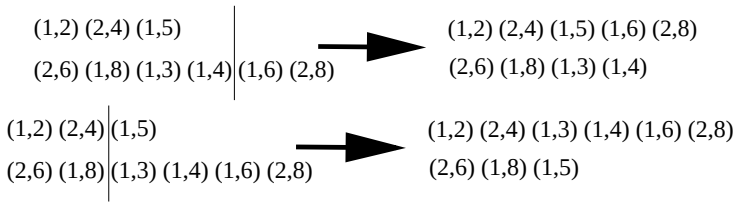


Figure 3.1: Crossover for edge representation.

Adjacency representation uses random bit mutation; a randomly selected bit is set to a random binary value. In the case of a 1 chosen for mutation, random bit mutation either keeps the bit (and the corresponding link) as it is, or makes it 0 (deleting the corresponding link). In the case of a 0 chosen for mutation, random bit mutation either sets it to 1 (inserting a new link), or keeps it the same. For edge representation, we adopted a mutation which corresponds to this outline: For each edge chosen for mutation, the edge is deleted with 50% probability; for each empty position (which corresponds to the 0s in adjacency representation) chosen for mutation, a new edge is inserted with 50% probability. Insertion of a new edge should hold the relation $a < b$ for each new edge (a, b).

[5] removes links while maintaining the reliability of the individual. [15] adds a new edge or deletes an old edge based on the degrees of the nodes.

Fitness:

The fitness of an individual, which will be minimized, is defined as:

$$Fitness(I) = [\sum_{e \in edges} Cost(e)] + P \cdot N(I)$$

where $N(I)$ is zero for 2-connected individuals and one, otherwise; and, P is a constant penalty value.

3.2 EXPERIMENTS

The experiments aim to compare the representations and analyze the effect of duplicate edges in edge representation. In experiments that compare the representations, the expectation is to see adjacency representation to perform worse due to the bias discussed in section 2.3.1.2.

We have used two different data sets of different sizes; one having 10 and the other having 15 computers.

Table 3.1: Results for both representations using rank-based selection. As the cost is minimized, small values are better.

Select.P.	Data Set	Adjacency Rep. (Best solution)	Edge Rep. (Best solution)
1-2	1	919.15	921.03
1-2	2	102.53	257.19
1-5	1	1019.98	1033.72
1-5	2	102.28	237.26
1-10	1	1052.24	1016.98
1-10	2	241.03	236.48

The crossover and mutation rates are taken to be %60 and %0.2, respectively. The size of the population is 100. The experiments are repeated 100 runs, 3000 generations each.

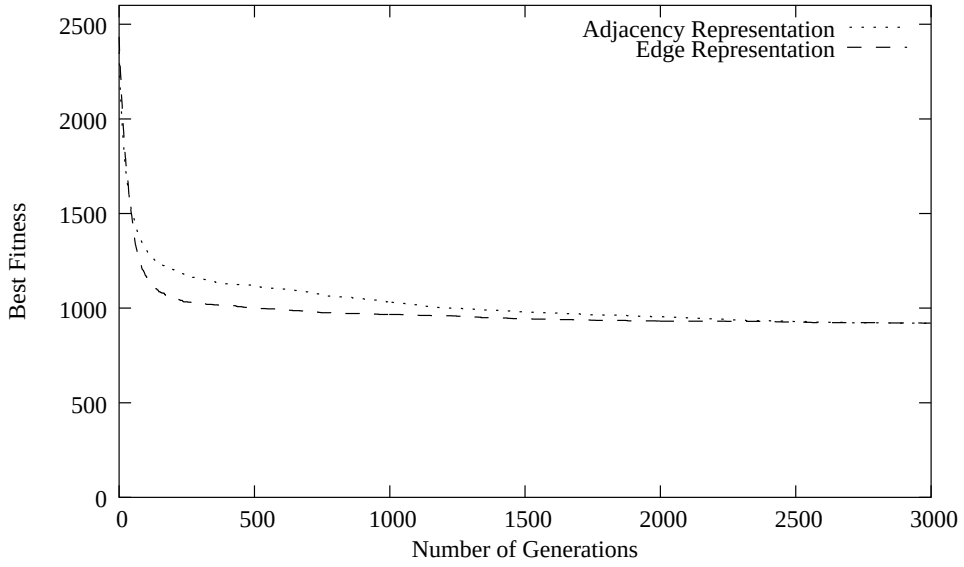
3.2.1 RANK-BASED SELECTION

In rank-based selection, individuals in the population are assigned selection probabilities based on their fitness values. The highest rank individual is assigned probability P_{high}; the lowest rank individual is assigned probability P_{low}; and, the individuals in between are assigned probabilities depending on a linear (or sometimes quadratic) function. In the experiments, we always assigned 1 to the lowest rank individual, and a numerical term like 1-P_{high} describes the selection pressures of the lowest and highest rank individuals. In other words, the best individual will have P_{high} times more chance to survive than the worst individual. A high P_{high} value means giving high chances of survival to higher rank individuals.

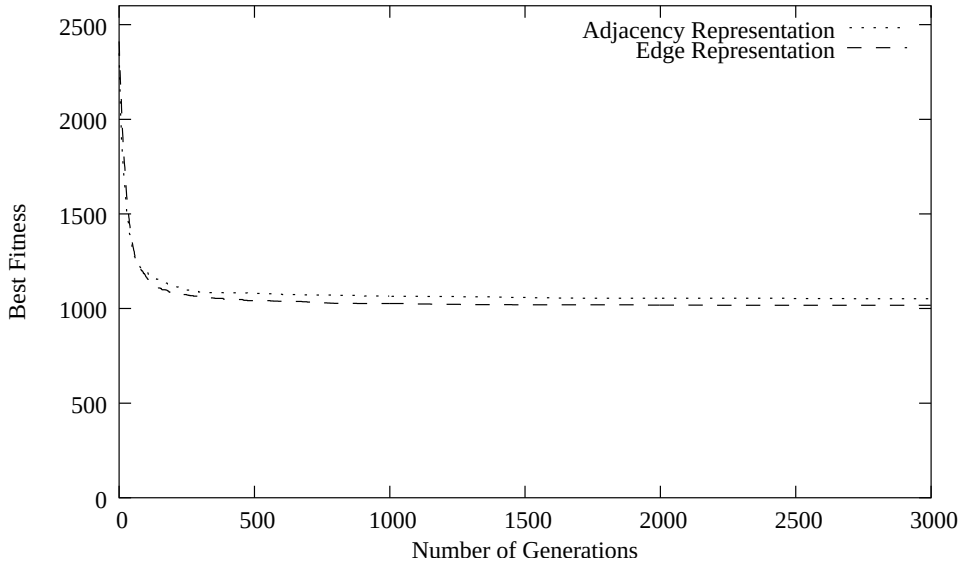
Our aim in this experiment is to control selection pressure (also called fitness pressure) and see its effects. We have employed linear rank-based selection with 3 different selection pressures; namely, 1-2, 1-5 and 1-10.

Figure 3.2 shows the performance of the representations on the first data set with selection pressures 1-2 and 1-10. For convenience, only two graphs are shown, and all results are summarized in table 3.1.

Figure 3.2 and table 3.1 show that edge representation performs better than (or at least as well as) adjacency representation in cases where small data set is used and a high fitness pressure is imposed. Table 3.1 shows that, when the selection strategy is made to have very high selection pressure (selection pressure being 1-10), edge representation performs better than adjacency representation even in the case of the large data set.



(a) Selection pressure 1-2. Average of 100 runs.



(b) Selection pressure 1-10. Average of 100 runs.

Figure 3.2: Rank-based selection performance graphs on data set 1 using selection pressures 1-2 and 1-10.

Table 3.2: Results for both representations using tournament selection. As the cost is minimized, small values are better.

Tour	Data Set	Adjacency Rep. (Best solution)	Edge Rep. (Best solution)
1	1	1117.00	1208.73
1	2	344.32	290.42
3	1	1209.03	1120.57
3	2	219.71	289.40

3.2.2 TOURNAMENT SELECTION

In tournament selection, one tour of the algorithm randomly selects n number of individuals from the population and the fittest of these individuals is selected for survival to the next tour of the selection algorithm (or, to the next generation of the GA if number of tours are exhausted). In the experiments, we used n as 2.

Our aim in this experiment is to control the number of tours and see its effects on the representations. The number of tours determines the fitness pressure of the algorithm. More number of tours means higher fitness pressure. We have employed tournament selection with 2 different number of tours; namely, 1 and 3.

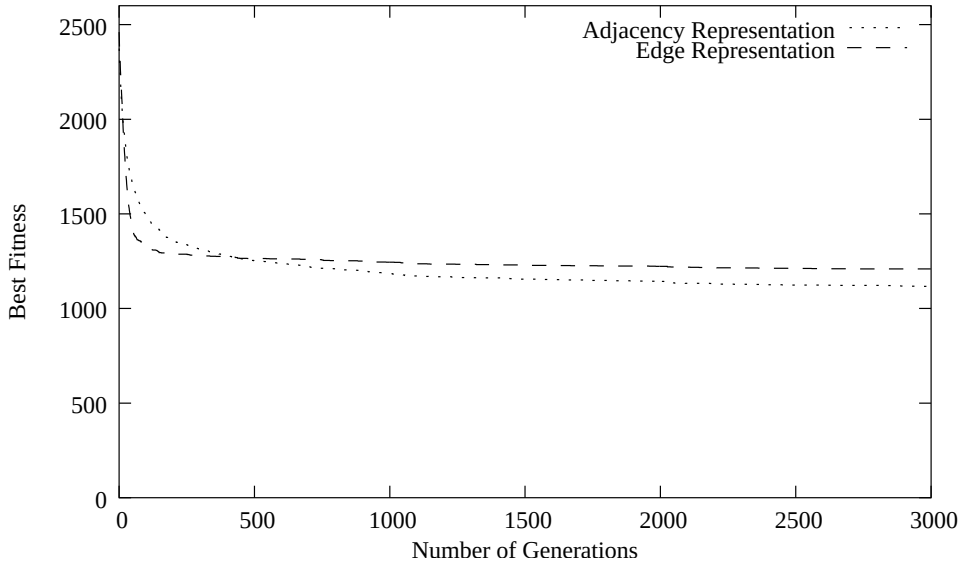
Figure 3.3 shows the performance of representations on data set one with 1 and 3 number of tours. For convenience, only two graphs are shown, and all results are summarized in table 3.2.

Figure 3.3 and table 3.2 show that edge representation is better in cases where small data set is used and a high elitism (i.e., with more number of tours) is imposed.

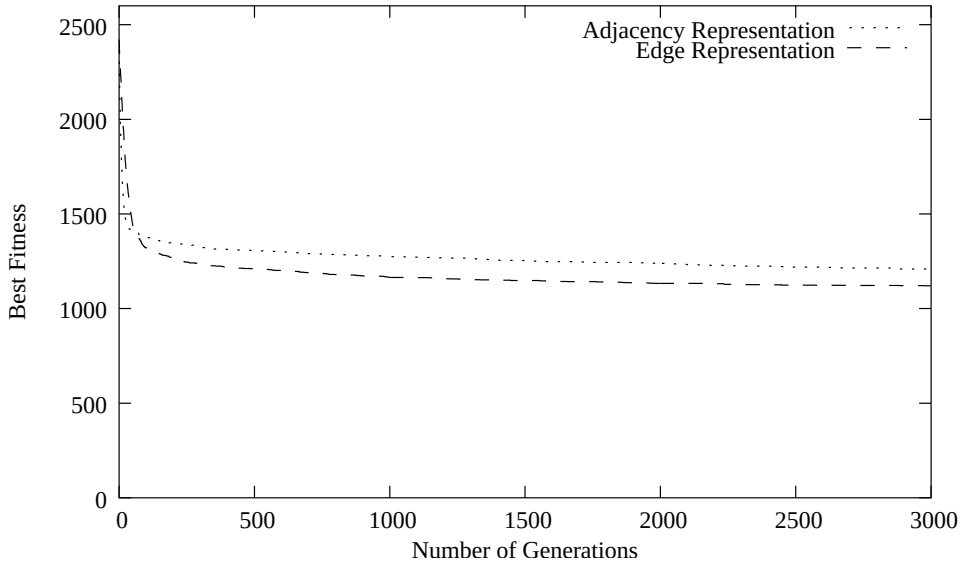
3.2.3 DUPLICATE EDGES

In the third experiment, we have analyzed the effect of duplicate edges in edge representation. As outlined in section 2.3.1.2, there are three different ways to deal with duplicate edges in a chromosome.

The experiments are performed on data set 1 using rank based selection with selection pressures 1-2 and 1-5.

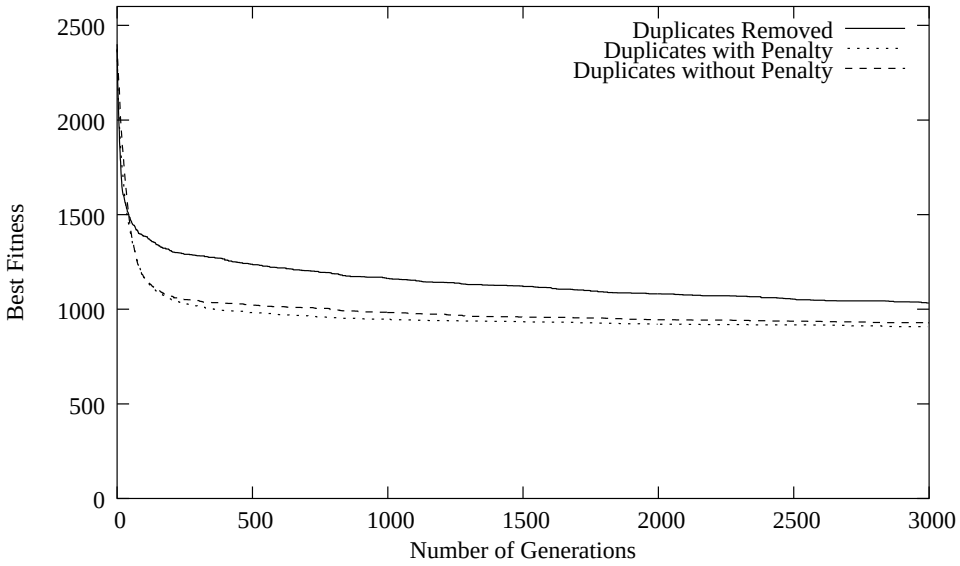


(a) 1 tour. Average of 100 runs.

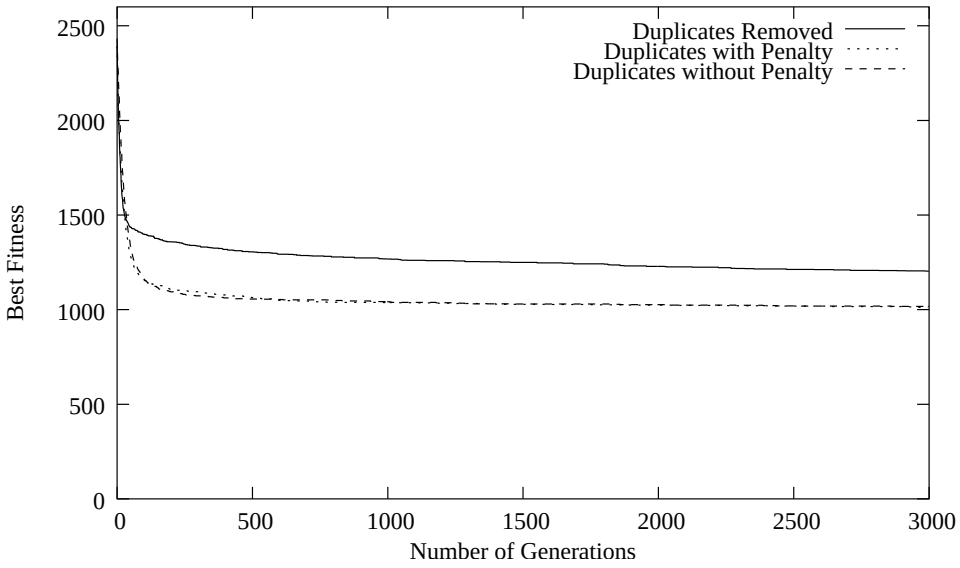


(b) 3 tours. Average of 100 runs.

Figure 3.3: Tournament selection performance graphs on data set 1 using 1 and 3 tours.



(a) Selection pressure 1-2. Average of 100 runs.



(b) Selection pressure 1-5. Average of 100 runs.

Figure 3.4: Effect of duplicate edges in edge representation on data set 1 using rank-based selection with selection pressures 1-2 and 1-5.

The results are shown in figure 3.4. The best performance is observed when duplicates are allowed with penalty. Next comes the case that uses duplicates without penalty. The worst performance is shown by the case where duplicates are removed. The difference between giving and not giving penalties in the case of keeping duplicates is small compared to the difference between keeping and removing duplicates.

The two graphs in figure 3.4 show that the performance of edge representation in the case where duplicates are removed is worse for selection pressure 1-2 than for selection pressure 1-5.

3.3 DISCUSSIONS

We have compared two main representations for network design problem. The problem that we used in our experiments included 2-connected reliability. The individuals who are not 2-connected are not repaired after crossover or mutation operators.

In this context, we have shown that neither representation is dominant in all cases we have considered. Edge representation performs better than adjacency representation in cases where small data set is used and the selection strategy is elitist. Adjacency representation performs better in cases where the data set is large and the selection strategy does not have high fitness pressure.

The overall results favor adjacency representation because it performs better in larger problems. But edge representation is very flexible (since the nature of its encoding offers several alternatives for crossover and mutation) and it has tunable parameters that adjacency representation does not have (like duplicate edges and possibble operations in mutation).

Besides, we have shown that duplicate edges in edge representation should be kept in the context we have used. Our third experiment showed that there is a significant performance difference between keeping and removing duplicate edges from individuals. Keeping duplicate edges with penalties is observed to give the best performance among the alternatives. However, the difference between giving and not giving penalties is not significant.

Duplicate edges in edge representation are beneficial for the diversity of the population. Probably, it is this diversity that makes the representation with duplicates perform better than the representation without duplicates.

The gap that this chapter fills in is the analysis of an alternative representation and a comparison with the traditional representation, in detail. The work shows the aspects of edge

representation that should be addressed and analyzes the performance for the most important aspects, namely crossover and duplicate edges.

Further analysis could involve the performance difference in the case where both representations have repair mechanisms and the case where both representations have continuous reliability without a repair mechanism.

CHAPTER 4

MULTI-OBJECTIVE NETWORK DESIGN

As mentioned in chapter 2, although network design problem is multi-objective, its multi-objective behaviour has not been analyzed in detail, yet. In this chapter, a multi-objective network design algorithm is implemented and compared with its aggregate counter-part.

As analyzed in detail in chapter 3, there are two possible representations. In this chapter, we also compared their performances in multi-objective domain.

4.1 METHODOLOGY

In this analysis, we tried to optimize cost, reliability and delay of the network. Cost and delay objectives are in conflict with the reliability objective.

4.1.1 NETWORK DESIGN PARAMETERS

Reliability:

In this work, 4 level reliability is used. The objective function is computed with the following pseudo-code:

```
function Reliability(I)
{
        if the number of articulation points in I is zero
        {
                result is TOP_RELIABLE;
        }
        else if the network I is 2-connected
        {
                result is TOP_RELIABLE / 2.0;
```

```
        }
        else if the network I is 1-connected
        {
                result is TOP_RELIABLE / 4.0;
        }
        else
        {
                result is 0;
        }
}
```

We used 4-level reliability because reliability function has to have multiple values in order to have more precision in it as an objective function. Each value in this function provides a level of reliability: the highest level is assigned to the most reliable networks that do not have articulation points; the second level is assigned to networks which are two connected and have some number of articulation points; the third level is assigned to the 1-connected networks; and the lowest level is assigned to non-connected networks.

Cost:

The cost of a network is simply taken to be the sum of the costs of existing links in the network.

Delay:

The delay of a network is approximated by the average number of hops between two pairs of computers (see section 2.3). The definition of the delay of a network is as follows:

$$Delay(I) = \frac{1}{N} \cdot \sum_{i,j \in computers} hops(i,j)$$

where N is the number of computers, and $hops(i,j)$ is the number of hops between computers i and j.

4.1.2 GENETIC ALGORITHM PARAMETERS

Initial Population, Crossover, Mutation:

Creation of initial populations, the crossover and mutation operators we have used in this chapter are the same as used in chapter 3 and outlined in section 3.1.2.

Fitness:

The definitions of the objectives for the multi-objective domain are defined in section 4.1.1.

The fitness function for the aggregate case is a summation of these objectives, as defined below:

$$Fitness(I) = Cost(I) + (MAXIMUM\_RELIABILITY - Reliability(I)) + Delay(I) * 500;$$

where the reliability value is inverted because fitness value is minimized. The delay function is too small (between 1 and 10 depending on the network) and hence, multiplied by 500 to make it comparable with other objectives. Note that 500 is not a weight but a normalization constant. The weights of the objectives in this aggregation are chosen as 1 in order not to give any priority to any objectives.

4.2 EXPERIMENTS

The experiments aim to analyze the behaviour of multi-objective network design problem.

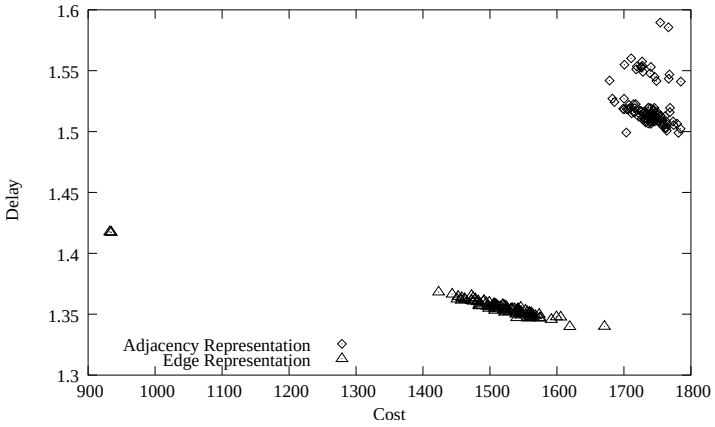
We have used the same data sets and the same parameters for the GA as mentioned in section 3.2.

Rank-based selection is used in all experiments.

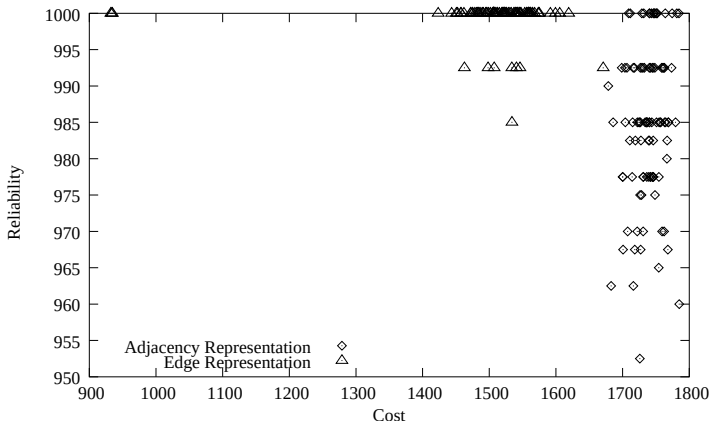
For multi-objective optimization, a Pareto Optimality based algorithm, namely, Multi-Objective Genetic Algorithm (MOGA), is used. MOGA assigns fitness values to individuals based on the number of individuals that dominate them. So, a fitness value of 0 (zero) is better and searched for.

4.2.1 PERFORMANCES OF REPRESENTATIONS

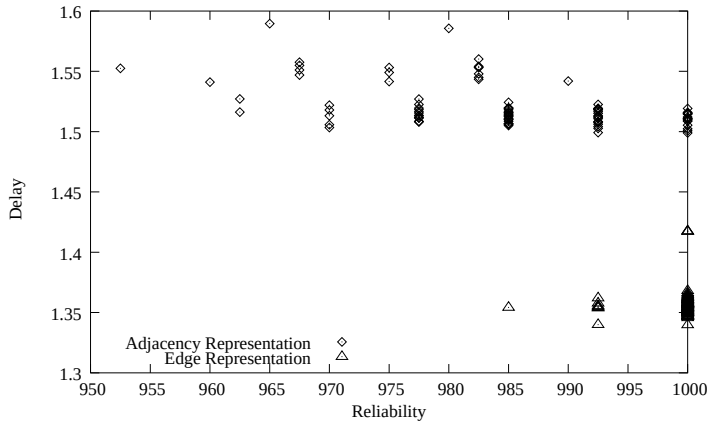
The aim of this experiment is to see which representation is better in multi-objective domain. The result of this experiment would determine which representation to use in later analysis in this chapter.



(a) Cost vs Delay.



(b) Cost vs Reliability.



(c) Reliability vs Delay.

Figure 4.1: Performances of Representations in Multi-Objective Network Domain. Rank-based selection with selection pressure being 1-10 is used on data set 2.

The final populations of edge and adjacency representation are shown in figure 4.1 for data set 15 and selection pressure 1-10. For convenience only the results for one selection pressure and data set is shown in this section, and all results for data set 1 are shown in appendix B.

The results show that edge representation significantly outperforms adjacency representation in all cases.

4.2.2 MULTI-OBJECTIVE OPTIMIZATION

As shown in previous section, edge representation is better than adjacency representation in multi-objective domain. For this reason, edge representation is used for the rest of the experiments.

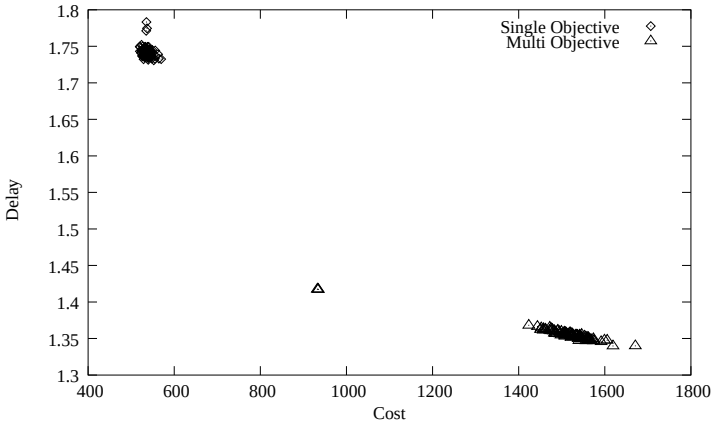
The aim of this experiment is to see the performance of multi-objective evolutionary (will be called MOO for simplicity) GA with respect to its aggregate counterpart. The aggregate counterpart (will be called AG for simplicity) uses the same objective functions of MOO to compose its fitness function. The performances of MOO and AG are shown in figure 4.2 for data set 2 and selection pressure 1-10. The figure shows that AG is better in optimizing reliability/cost; however, it is poor in optimizing delay. When the dominance of all objectives are considered, neither AG nor MOO is dominant in all cases.

In figure 4.2, MOO is observed to perform poor in optimizing cost of the network. This is due to the fact that reliability and delay of a network work against the cost of the network. Hence, MOO intuitively better optimizes reliability and delay. To analyze the performances better, we run AG and MOO for optimization of only the cost and the reliability of the network. The results are shown in figure 4.3. Although the figure shows that MOO is very poor in optimization of reliability when compared to AG, the solutions returned by AG and MOO do not dominate each other (MOO has lower cost, and AG has higher reliability).

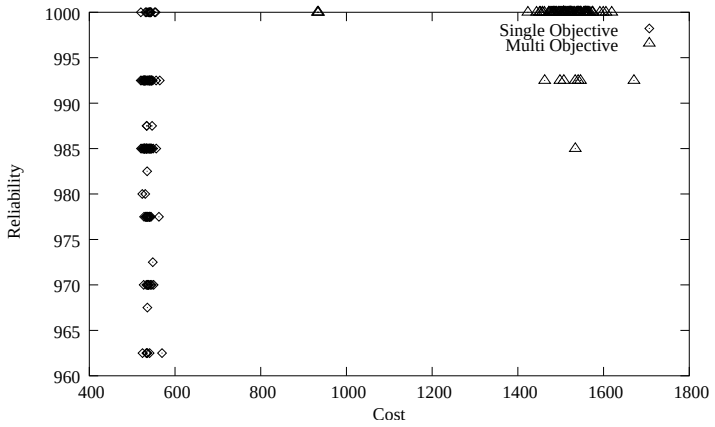
4.3 DISCUSSIONS

In this chapter, edge representation and adjacency representation are compared in multi-objective domain. As mentioned in chapter 3, edge representation and adjacency representation are not dominant over each other in all cases analyzed in section 3. Our results for multi-objective network domain show that edge representation is superior than adjacency representation in all cases.

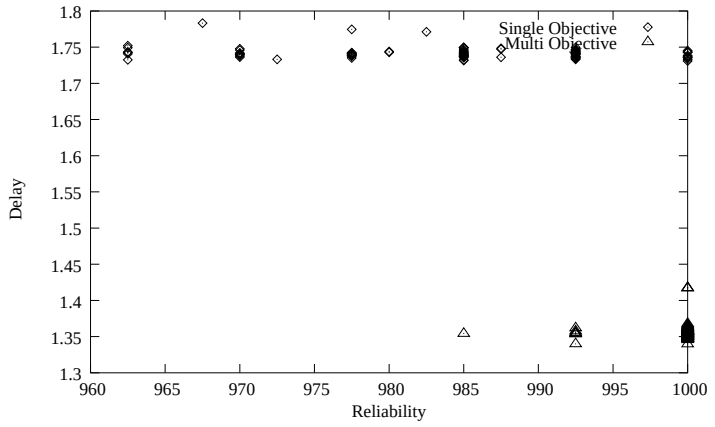
We also compared multi-objective network design algorithm with its counter-part. We



(a) Cost vs Delay.

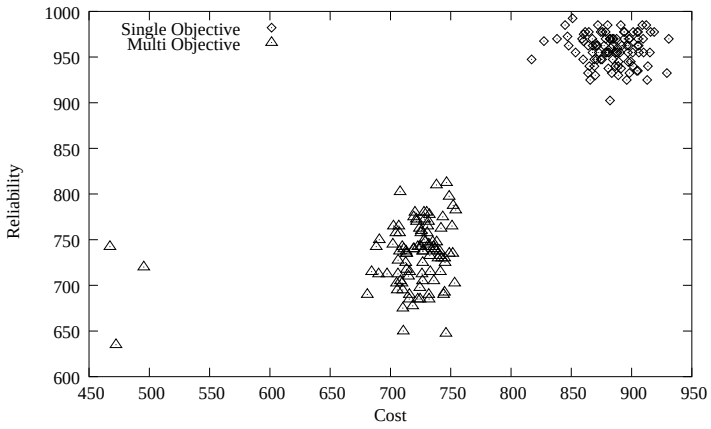


(b) Cost vs Reliability.

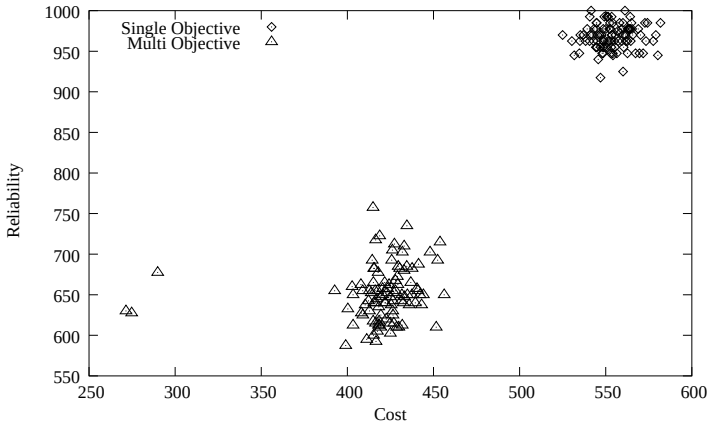


(c) Reliability vs Delay.

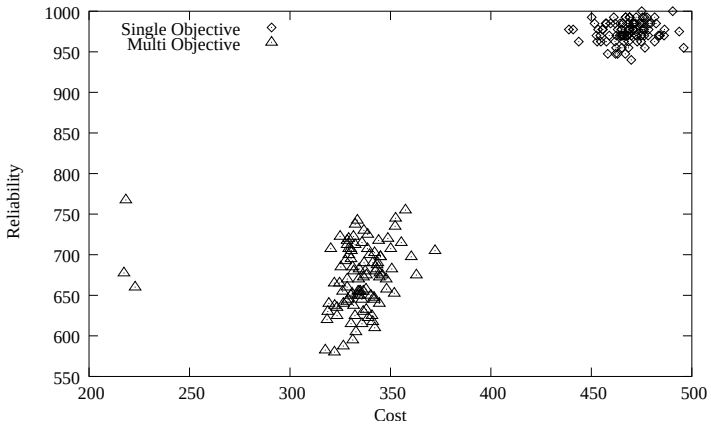
Figure 4.2: Performances of aggregate and multi-objective algorithms over cost, delay and reliability objectives. Rank-based selection with selection pressure being 1-10 is used on data set 2.



(a) Rank 2.



(b) Rank 5.



(c) Rank 10.

Figure 4.3: Performances of aggregate and multi-objective algorithms over only cost and reliability objectives. Rank-based selection with selection pressures being 1-2, 1-5 and 1-10 is used on data set 2.

showed that the solutions returned by multi-objective and aggregate algorithms are not dominant when all objectives are considered. However, aggregate case is better in optimization of reliability/cost, while multi-objective case is better in optimization of delay and reliability. Single-objective case is poor in optimization of delay, whereas multi-objective case is poor in optimization of cost.

The delay and the reliability of a network work against the cost of the network. We analyzed the performances of multi-objective and aggregate algorithms without this effect and showed that although solutions returned by both algorithms do not dominate each other, the solutions returned by aggregate algorithm are better in terms of solution quality (reliability/cost). As mentioned in section 2.2, the choice of weights for aggregation of objectives is important for the performance and in most cases it is very difficult to find the right combination (if there is any at all). In this study, we did not assign any weights.

In summary, the gap this chapter fills in is the comparison of representations for reliable network design in multi-objective domain and analysis of multi-objective optimization for reliable network design with its aggregate counterpart.

The future work for this chapter may include analysis of other multi-objective algorithm approaches like VEGA, Lexiographic Ordering and Game Theory.

CHAPTER 5

CONCLUSIONS

In this study, reliable network design problem is analyzed in detail using evolutionary algorithms. Both aggregate and multi-objective algorithms are exercised, and their performances are analyzed. An alternative representation, i.e. list of edges representation, is investigated in detail, and its performance is compared with traditional representation, i.e. adjacency matrix representation.

We showed in chapter 3 that no representation is dominant in all of the cases (i.e., data sets and selection pressures) in the aggregate context we have adopted. However, in the multi-objective domain of the problem, we showed that edge representation is significantly better than adjacency representation in the context we have adopted.

We also analyzed the design issues of a GA that may use edge representation; namely, crossover, mutation and duplicate edges. We showed that duplicate edges are mandatory for the performance of the GA that uses edge representation.

Besides, we implemented a multi-objective evolutionary GA for reliable network design problem (whose objectives to be optimized were cost, delay and reliability), and we tried to understand its performance with respect to its aggregate counterpart. We showed that aggregate case is better in optimization of reliability/cost and poor in optimization of delay. Whereas, multi-objective algorithm was poor in optimization of cost. The poorness of multi-objective algorithm in optimization of cost was supposed to be the fact that delay and reliability of a network work against the cost of the network. To analyze the performances free from this effect, we made experiments for optimization of only cost and reliability and showed that although both algorithms returned non-dominating solutions, the solutions of aggregate

optimization were better in terms of quality (reliability/cost).

In summary, the contributions of this study are:

- Analysis of an alternative representation (edge representation) for reliable network design problem.
- Comparison of two representations (edge representation and adjacency representation) for both aggregate and multi-objective optimization.
- Analysis of multi-objective behaviour of reliable network design problem with its aggregate counterpart.

5.1 FUTURE WORK

- The performance difference of representations for network design problem can be analyzed for the case where both representations have repair mechanisms and the case where both representations have continuous reliability without a repair mechanism.
- Multi-objective behaviour of network design problem can be analyzed using different kinds of multi-objective algorithms like VEGA, Lexiographic Ordering and Game Theory.

REFERENCES

[1] P. Aiyarak, A. S. Saket, M. C. Sinclair, Genetic Programming approaches for minimum cost topology optimization of optical telecommunication networks, *Int. Conf. on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'97)*, 1997.

[2] A. Kumar, R. M. Pathak, M. C. Gupta, Genetic Algorithm Based Approach for Designing Computer Network Topology, *Proceedings of the 1993 ACM Conference on Computer Science*, 1993.

[3] M. M. Atiqullah, S. S. Rao, Reliability optimization of communication networks using simulated annealing, *Microelectronics and Reliability*, vol. 33, 1993, pp. 1303-1319.

[4] H. F. Beltran, D. Skorin-Kapov, On minimum cost isolated failure immune networks, *Telecommunications Systems*, vol. 3, 1994, pp. 183-200.

[5] S. Cheng, Topological optimization of a reliable communication network, *IEEE Transactions on reliability*, vol. 47, no. 3, 1998.

[6] C. A. Coello, A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques, *Knowledge and Information Systems*, 1998.

[7] D. W. Coit, A. E. Smith, Use of a genetic algorithm to optimize a combinatorial reliability design problem, *Proc. of Third IIE Research Conference*, 1994.

[8] C. J. Colbourn, The combinatorics of networks reliability, Oxford University Press, 1987.

[9] C. J. Colbourn, T. B. Brecht, Lower bounds on two-terminal network reliability, *Discrete Applied Mathematics*, vol. 21, pp. 185-198, 1988.

[10] G. Cove, Issues on large network design, Network Analysis Corp., NY., ARPA rep., Jan. 1974.

[11] B. Dengiz, F. Altiparmak, A. E. Smith, A genetic algorithm approach to optimal topological design of all terminal networks, *Intelligent Engineering Systems Through Artificial Neural Networks*, vol. 5, pp 405-410, ASME Press, 1995.

[12] B. Dengiz, F. Altiparmak, A. E. Smith, Local search genetic algorithm for optimal design of reliable networks, *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 179-188, 1997.

[13] B. Dengiz, F. Altiparmak, A. E. Smith, Genetic algorithm design of networks considering all-terminal reliability, *In Proceedings of the 6th Industrial Engineering Research Conference*, pp. 30-35, Miami Beach, FL, 17.-18. 1997.

[14] B. Dengiz, F. Altiparmak, A. E. Smith, Efficient Optimization of All-terminal Reliable Networks Using an Evolutionary Approach, *IEEE Transactions on Reliability*, 1997.

[15] B. Dengiz, A. E. Smith, Evolutionary methods for design of reliable networks, *Telecommunications Optimization: Heuristics and Adaptive Methods*, Wiley Press, 2000.

[16] H. Frank, I. T. Frisch, W. Chou, Topological considerations in the design of the ARPA computer network, *Conf. Rec., 1970 Spring Joint Comput. Conf.*, AFIPS Conf. Proc. vol. 36, Montvale, NJ: AFIPS Press, 1970.

[17] M. Gerla, The design of store-and-forward networks for computer communications, Ph.D. dissertation, School of Eng. and Applied Sci., Univ. of California, Los Angeles, Jan. 1973.

[18] F. Glover, M. Lee, J. Ryan, Least-cost network topology design for a new service: an application of a tabu search, *Annals of Operations Research*, vol. 33, pp. 351-362, 1991.

[19] D. E. Goldberg, Genetic algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[20] J. H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan press, Ann Arbor, 1975.

[21] R. H. Jan, F. J. Hwang, S. T. Cheng, Topological optimization of a communication network subject to a reliability constraint, *IEEE Transactions on Reliability*, vol. 42, pp. 63-70, 1993.

[22] K. Ko, K. Tang, C. Chan, K. Man, S. Kwong, Using genetic algorithms to design mesh networks, *IEEE Computer*, pp. 56-61, Aug 1997.

[23] A. Konak, A. E. Smith, Multiobjective Optimization of Survivable Networks Considering Reliability, *Proc. of The 10th International Conference on Telecommunication Systems, Modeling and Analysis (ICTSM10)*, Monterey, CA, October 3-6, 2002.

[24] R. Kumar, V. P. Krishnan, K. S. Santhanakrishnan, Design of an optimal communication networks using multiobjective genetic optimization, *Industrial Technology*, 2000.

[25] R. Kumar, P. P. Parida, M. Gupta, Topological design of communication networks using multi-objective genetic optimization, *Evolutionary Computation*, 2002.

[26] C. Lo, W. Chang, A multi-objective hybrid genetic algorithm for the capacitated multipoint network design problem, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 30, no. 3, pp. 461-470, 2000.

[27] B. Ombuki, M. Nakamura, Z. Nakao, K. Onaga, Evolutionary computation for topological optimization of 3-connected computer networks, *IEEE International Conference on Systems, Man and Cybernetics*, 1999.

[28] S. Pierre, M.-A. Hyppolite, J.-M. Bourjolly, O. Dioume, Topological design of computer communication networks using simulated annealing, *Engineering Applications of Artificial Intelligence*, vol. 8, pp. 61-69, 1995.

[29] K. Sayoud, K. Takahashi, B. Vaillant, A genetic local tuning algorithm for a class of combinatorial design problems, *IEEE Communications Letters*, vol. 5, 2001.

[30] K. Steiglitz, P. Weiner, D. J. Kleitman, The design of minimum cost survivable networks, *IEEE Trans. Circuit Theory*, pp. 455-460, Nov. 1969.

[31] A. R. P. White, J. W. Mann, G. D. Smith, Genetic algorithms and network ring design, *Annals of Operational Research*, 1999.

[32] B. Yaged, Minimum cost routing for static network models, *Networks*, vol. 1, pp. 139-172, 1971.

[33] M. S. Yeh, J. S. Lin, W. C. Yeh, A new Monte Carlo method for estimating network reliability, *Proc. 16th International Conference on Computers & Industrial Engineering*, 1994.

APPENDIX A

DATA SETS

First data set:

```
0   130 90  106 100 100 200 180 200 150
130 0   980 155 180 200 300 150 200 140
90  980 0   1   150 151 152 153 154 155
106 155 100 0   1   161 162 163 164 165
100 180 1   1   0   171 172 173 174 175
100 200 151 161 171 0   181 182 183 184
200 300 152 162 172 181 0   191 192 193
180 150 153 163 173 182 191 0   1   1
200 200 154 164 174 183 192 1   0   1
150 140 155 165 175 184 193 1   1   0
```

Second data set:

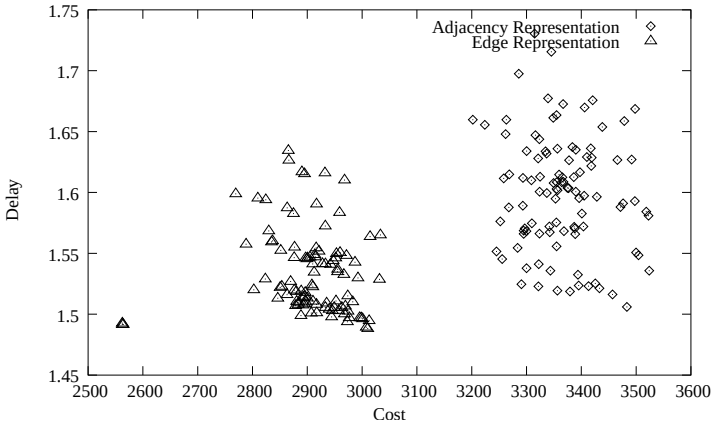
```
0 13  20 16 10 40 60 70 80 90 40 60 70 80 90
13 0  9  15 18 20 16 40 13 9  20 16 40 13 9
20 9  0  10 14 16 15 10 10 18 16 15 10 10 18
16 15 10 0  20 10 18 14 18 10 10 18 14 18 10
10 18 14 20 0  20 16 40 13 9  20 16 40 13 9
40 20 16 10 20 0  60 70 80 0  90 60 70 80 90
60 16 15 18 16 60 0  10 50 18 10 50 18 9  20
70 40 10 14 40 70 10 0  10 14 10 14 20 50 1
80 13 10 18 13 80 50 10 0  90 90 20 30 14 90
90 9  18 10 9  90 18 14 90 0  20 30 40 50 60
40 20 16 10 9  90 10 10 90 20 0  70 18 20 40
60 16 15 18 20 60 50 14 20 30 70 0  30 40 80
70 40 10 14 16 70 18 20 30 40 18 30 0  65 89
80 13 10 18 40 80 9  50 14 50 20 40 65 0  80
90 9  18 10 9  90 20 1  90 60 40 80 89 80 0
```

APPENDIX B

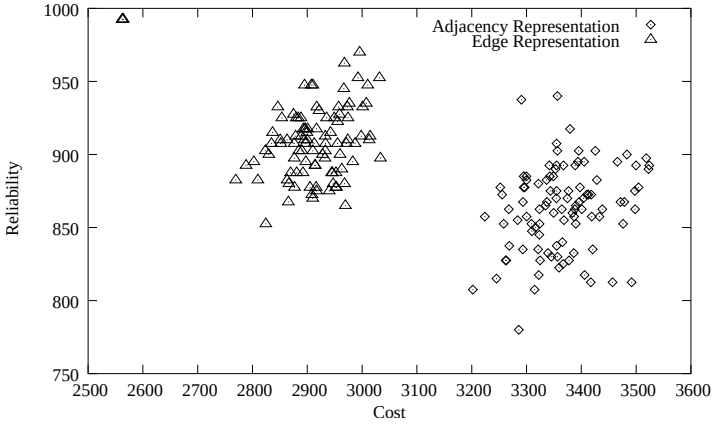
PERFORMANCE GRAPHS OF REPRESENTATIONS IN MULTI-OBJECTIVE DOMAIN

This appendix gives the graphs that show the performances of representations in Multi-objective domain.

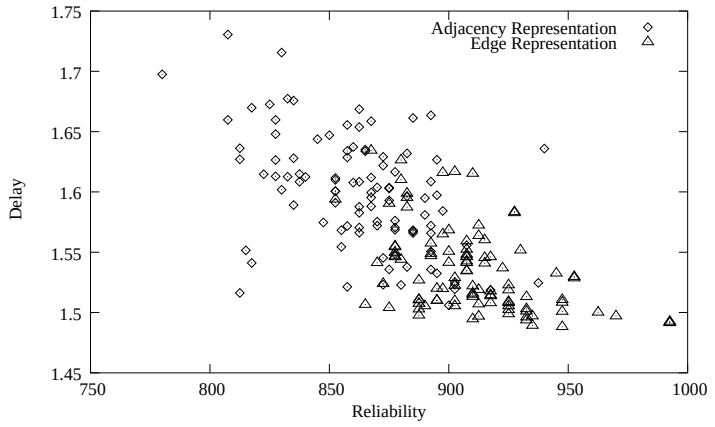
The figures are discussed in section 4.2.1.



(a) Cost vs Delay.

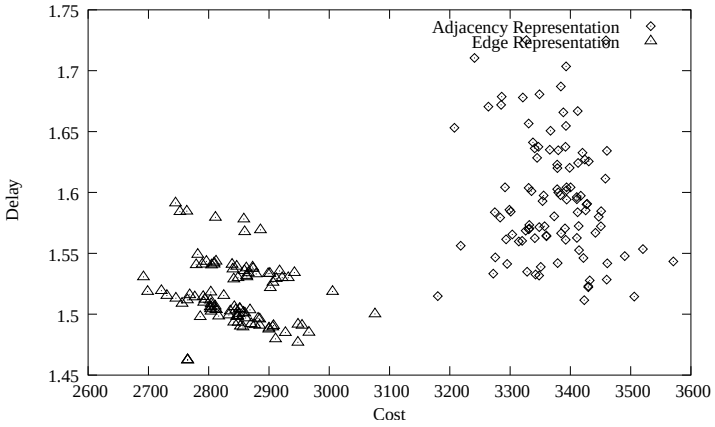


(b) Cost vs Reliability.

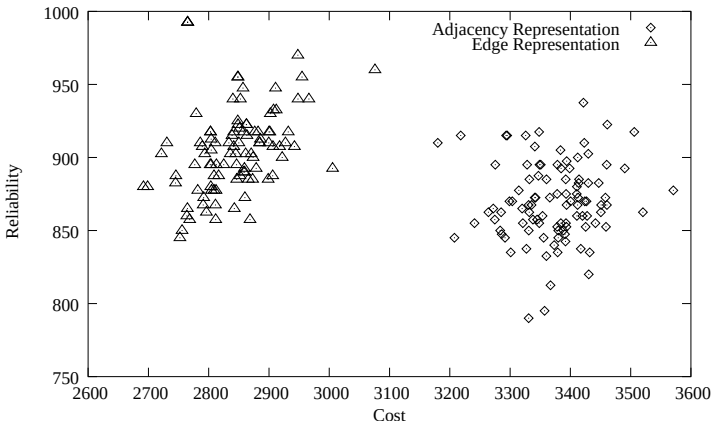


(c) Reliability vs Delay.

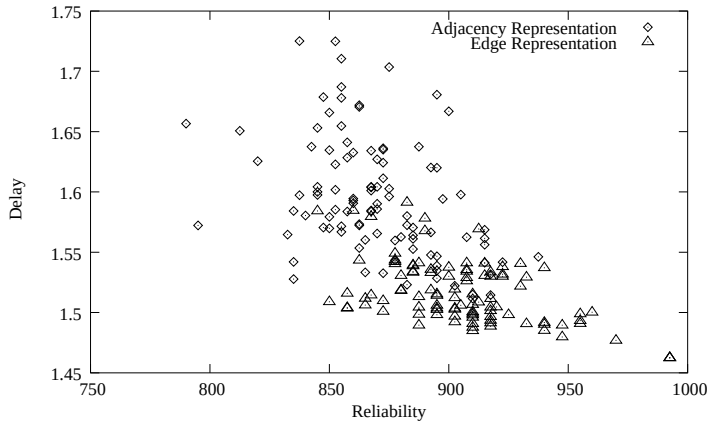
Figure B.1: Performances of Representations in Multi-Objective Network Domain. Rank-based selection with rank being 2 is used on data set 1.



(a) Cost vs Delay.

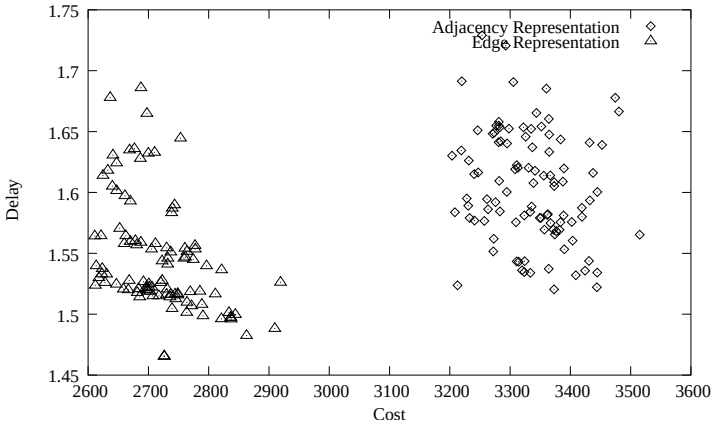


(b) Cost vs Reliability.

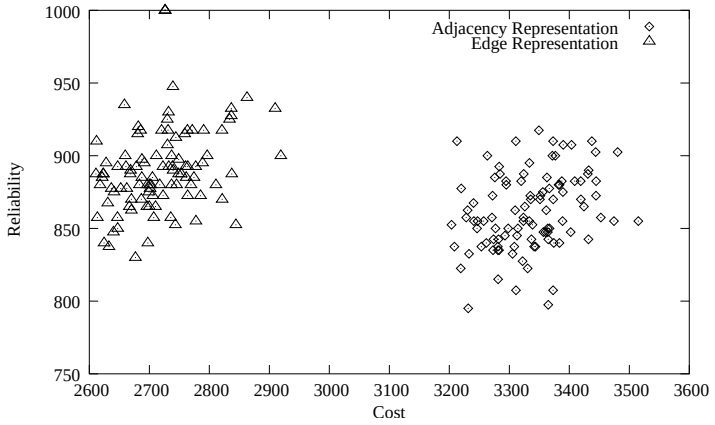


(c) Reliability vs Delay.

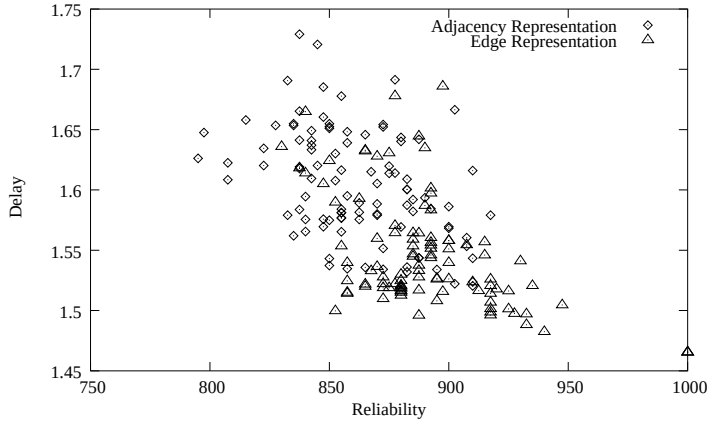
Figure B.2: Performances of Representations in Multi-Objective Network Domain. Rank-based selection with rank being 5 is used on data set 1.



(a) Cost vs Delay.



(b) Cost vs Reliability.



(c) Reliability vs Delay.

Figure B.3: Performances of Representations in Multi-Objective Network Domain. Rank-based selection with rank being 10 is used on data set 1.