Multi-objective Genetic Algorithm for Solving the Multilayer Survivable Optical Network Design Problem

Huynh Thi Thanh Binh
School of Information and Communication Technology
Hanoi University of Science and Technology, Hanoi, Vietnam

Abstract - This paper considers the problem of designing a Multilayer Survivable Optical Network for the customers' Demands Problem called MSONDP. The network is modelled by two graphs: an undirected graph \( G_1 = (V_1, E_1) \) and a complete undirected and weighted graph \( G_2 = (V_2, E_2, c) \). The goal objective of this problem is to design connections based on customers' demands with the smallest a minimum network cost to protect the network against all failures. This paper introduces a multi-objective approach for MSONDP. These objectives are to minimize the network cost (\( \text{totalCost} \)) and the maximum number of connections passing over a link (\( \text{maxConn} \)). Further, this paper also proposes a multi-objective genetic algorithm to solve this problem. The experimental results on real world and random instances are reported to show the efficiency of the proposed algorithm comparing compared to the single genetic PGAMSONDP.

Keywords: Survivable Network Design, MSONDP, NSGA-II.

I. INTRODUCTION

The advent of networks, particularly the appearance of the Internet, marked an important juncture in the field of information technology and communications. Since then, networks have brought considerable benefits to all areas of life including, among others, the economy, culture, and the military. Therefore, occasionally, a single network failure can cause serious consequences, particularly economic. This issue obligates network providers to ensure the reliability for their products. It is also the reason for defining the Survivable Network Design Problem (SNDP).

Then, together with the advent and the on-going development of optical networks, and now multilayer optical networks, SNDP has reached a new stage in its evolution and improved significantly. The advantages of optical networks (such as heavy flow, less signal decline, etc.) make it increasingly popular than the traditional networks. Therefore, SNDP for optical networks is becoming essential and gaining significant attention. This paper solves SNDP for multilayer optical networks, which is the latest model of this problem, and is called the Multilayer Survivable Optical Network Design Problem (MSONDP).

As with many other network design problems, minimum network cost is an important objective of MSONDP. Utilizing available links is an excellent method to achieve this. This method, however, introduces another problem that affects the survivability of the network. In particular, when a link is considerably utilized, many connections will fail if this link breaks down. Thus, besides the cost objective, this paper also considers the problem of minimizing the number of connections passing over a link. Obviously, these objectives are in conflict with each other. Therefore, it is almost impossible to satisfy both of them simultaneously. Hence, this research aims to find acceptable solutions via a multi-objective genetic algorithm.

First, MSONDP is formulated as a multi-objective design problem, called M-MSONDP. Then, using individual representations and genetic operators described previously [12, 13], this research proposes a multi-objective genetic algorithm, which based on NSGA-II, called NSGA-II-MSONDP. To demonstrate the efficiency of NSGA-II-MSONDP, this paper experiments with the data used in [12, 13] and compares the results with the single objective (cost) genetic algorithm (PGAMSONDP) [13].

The rest of this paper is organized as follows. Related works are described in section 2. The problem formulation is introduced in section 3. The proposed algorithm is described in section 4. Section 5 details the experiments and computational results. Section 6 concludes the paper with discussions and future works.

II. RELATED WORKS

Because survivability is essential in any network system, considerable research has been undertaken on this topic. Many models of the problem were researched and solved using specific algorithms. According to previous research [3], this problem was solved in many ways using two different approaches, i.e., exact and approximate algorithms.

With the exact approach, Borne et al. studied the survivable IP-over-optical network design problem [2]. They gave a 0-1 integer programming formulation for this problem, described some valid inequalities, discussed separation algorithms for these inequalities, and introduced some reduction operations. Based on these, they proposed a Branch-and-Cut algorithm to solve the problem. They experimented with random instances that were generated with 10 to 45 nodes and 10, 15, 20, and 30 edges. The real instances used to test the algorithm were provided by the French telecommunications operator France Télécom. These instances had 18 to 60 nodes and 31 to 102 edges. However, their algorithm only solved small test sets (less than 35 nodes and 30 edges in random instances, and less than 60 nodes and 102 edges in real instances). With larger test sets, the running time...
is considerably long (approximately 5 h) and many of them still have no solution.

Then, in 2009, Borne et al. proposed the Branch-and-Cut and Branch-and-Cut-and-Price algorithms based on path formulation for solving the multilayer capacitated survivable network design problem [9]. However, they only solved it with graphs having 10 nodes, 20 edges, and 20 demands.

Further, Zymolka [8], who followed the exact approach, defined a cost-efficient design for the survivable optical telecommunication networks problem and proposed Branch-and-Price algorithm with four branching rules after modeling this problem in an integer linear program. His problem was separated into two individually difficult sub-problems, one of which was to route the connection with a corresponding dimensioning of capacities, and the other to seek conflict-free assignments of available wavelengths to the light paths (a common characteristic of optical networks) using a minimum number of involved wavelength converters. The first sub-problem was solved using a three-step algorithm as follows: 1. pre-processing transformation of the hardware model; 2. application of DISCNET as the main optimization routine; and 3. post-processing to adapt solutions for the original problem. In the second sub-problem, to derive a linear program formulation, Zymolka proposed an exact Branch-and-Price method, namely, an integer linear programming approach [8].

In 2011, MSONDP was proven to be NP-hard by Borne et al. [1]. They formulated this problem in terms of a 0-1 linear program based on path variables. Then, they discussed the pricing problem and proved that it reduces to a shortest path problem, which they used to propose a Branch-and-Price algorithm. However, this was an exact approach for an NP-hard problem, and therefore, they could only offer solutions for a maximum input of 17 nodes on G₁, 20 nodes on G₂, and 25 demands.

In 2012, Binh and Ly proposed a genetic algorithm called GAMSONDP to solve MSONDP [12]. This algorithm solved the large test sets that Branch-and-Price [1] could not. However, a drawback of GAMSONDP is its long running time to find solutions for the large data sets. Hence, Binh and Tung proposed a parallel genetic algorithm PGAMSONDP to improve the time [13]. However, they only considered the cost objective.

Therefore, to the best of our knowledge, MSONDP has only been solved as a single objective problem. Moreover, as mentioned in section 1, there is a potential risk if the research only considers minimizing the network cost. Therefore, this paper introduces a multi-objective model for M-MSONDP in the next section.

III. PROBLEM FORMULATION

As with the different models of SNDP, the aim of M-MSONDP is to protect the network against failures. To ensure survivability of the network, the goal of this research is to design a network such that each of the connection demands has two satisfactory node-disjoint paths. One of them is the working path; the other is the backup path. When the working path fails, the connection is switched to the backup path.

The following is the multi-objective formulation of M-MSONDP.

Input:

- An undirected graph $G_1 = (V_1, E_1)$ represents the logical layer, in which $V_1$ is the set of nodes and $E_1 = \{e^1 | e^1 = (u, v), u, v \in V_1\}$ is the set of edges.
- A complete undirected and weighted graph $G_2 = (V_2, E_2, c)$ represents the physical layer, in which $V_2 \supseteq V_1$ and $E_2 = \{e^2 | e^2 = (u, v'), u, v' \in V_2\} \supseteq E_1$ are the set of nodes and edges, respectively. Each weight $c$ is the cost to connect a corresponding edge in the graph.
- A set of connection demands $T = \{t_i | t_i = (L^1_i, L^2_i)\}, i = \{1, ..., |T|\}$, where
  - $L^1_i$ is the list of nodes that the found working path for $t_i$ must traverse in order.
  - $L^2_i$ is the list of nodes that the found backup path for $t_i$ must traverse in order.
  - The first node and the last node in $L^1_i$ are the same as the first node and the last node in $L^2_i$ respectively.

Constraints:

- All connection demands are satisfied.
- Each connection demand has two node-disjoint paths.

Objectives:

\[ \text{TotalCost} \rightarrow \min \]

With

\[ \text{totalCost} = \sum_{i=1}^{\left| T \right|} c(t_i), \quad (1) \]

where $c(t_i)$ is cost of the $i$-th demand and $|T|$ is the number of customers’ demands.

\[ \text{maxConn} \rightarrow \min \quad (2) \]

With

\[ \text{maxConn} = \text{MAX}(\text{conn}_1, \text{conn}_2, ..., \text{conn}_e), \]

where \( \text{conn}_i \) is the number of connections passing over $e^2_i$.

Output:

- A set of paths that corresponds with each of the demands.

Because two objectives are in conflict with each other, there is no completely optimal solution. Therefore, the next section introduces an algorithm to find acceptable solutions.

IV. PROPOSED METHODOLOGY

This section proposes a multi-objective genetic algorithm based on NSGA-II to solve M-MSONDP. The following is the main concept of this algorithm.

A. Genetic Algorithm

Individual Representation: We use the individual representation method that was presented in [12].
Each individual, which is a solution of the problem, has \(|T|\) chromosomes and each chromosome has two genes called the working gene and the backup gene. These two genes encode for the two node-disjoint paths corresponding with \(L^1\) and \(L^2\) of a demand and are represented by a list of nodes in those two paths.

We create individuals for the initial population with a “path-finding” algorithm [13]. This algorithm finds a path from source node A to destination node B that includes all required nodes. We then choose nodes that are different from the required nodes, insert them between A and B, and then, we insert the required nodes in turn into random positions.

Figure 1 depicts an individual representation. This individual has k chromosomes. Chromosome \(t_i\) encodes demand \(t_i\) that has \(L^1_i = \{4, 1, 3\}\) and \(L^2_i = \{4, 2, 6, 3\}\). The first and the last node in each row are the source and destination nodes. The other numbers represent nodes that are to be traversed. The representation and constraints are the same for the other chromosomes.

**Crossover operator:** Apply two types of crossover operators as in [12, 13], one of which is a “chromosome crossover operator,” and the other a “modified path crossover operator.”

The idea of a “chromosome crossover operator” is that some chromosomes of Parent 1 are replaced by some of Parent 2. This crossover operator is implemented as follows. First, select two different individuals in the population and choose a “cut-point” randomly. This “cut-point” divides the number of chromosomes of each parent into two parts. Then, join the first part of Parent 1 with the second part of Parent 2 and conversely, create two new children. If these children coincide with individuals in the current population, they will be discard. The chromosome crossover operator is shown in Figure 2.

If a conflict occurs in any chromosome, i.e., there is a pair of consecutive nodes in the working gene that is repeated in the backup gene, this chromosome retains its initial backup gene. This is known as unavailable replacement (Figure 3).

New individuals are checked to guarantee uniqueness.

**Mutation Operator:** There are three types of mutation: “delete a node” [12], “replace a path segment,” and “chromosome renew” [13].

To execute “replace a path segment,” choose a chromosome \(t_i\) in an individual randomly, and then refer it to the i-th demand; choose two consecutive nodes \((n_j, n_{j+1})\) either in \(L^1_i\) or in \(L^2_i\) randomly; use Dijkstra’s algorithm to find a path from \(n_j\) to \(n_{j+1}\); and replace \([n_j, n_{j+1}]\) segment in \(t_i\)’s working gene if \((n_j, n_{j+1})\) are selected in \(L^1_i\) or backup gene if \((n_j, n_{j+1})\) are selected in \(L^2_i\), by the found path.

The “chromosome renew” is implemented as follows: select an individual randomly, and then, choose k (0<k<\(|T|\)) chromosomes in this individual such that their total cost is not equal to the individual’s. Consequently, the \(|T|\)k remaining chromosomes are created by the aforementioned “path-finding” algorithm.

The new individual is checked to guarantee uniqueness. If a new individual already existed in the current population, implement the procedure again.

**B. NSGA-II**

In NSGA-II, denote the parent population as \(P_r\), the offspring population as \(Q_r\) is the number of individuals in \(P_r\) as \(N\). The NSGA-II algorithm can be divided into two main processes: fast non-dominated sorting and crowding distance sorting.

First, create the parent population \(P_1\) as the initial population with a size of \(N\). Second, select individuals randomly in \(P_1\), using the above crossover and mutation operators to create new individuals for the child population, \(Q_1\). If the stop condition is satisfied, return the individuals of the current population \(P_r\). Otherwise, form a combined population \(R_r = P_1 \cup Q_1\).

Sort the population \(R_r\) according to non-domination. Because all members of the previous and current population are included in \(R_r\), elitism is ensured. Now, solutions belonging to the best non-dominated set \(F_1\) are the best solutions in the combined population and must be emphasized.
more than any other solution. If the size of \( F_1 \) is smaller than \( N \), choose all members of \( F_1 \) for the new population \( P_{t+1} \). The remaining members of the population \( P_{t+1} \) are chosen from subsequent non-dominated fronts in the order of their ranking. Therefore, solutions from set \( F_2 \) are chosen next, followed by solutions from set \( F_3 \), and so on. This procedure is continued until no more sets can be accommodated.

After using crowding distance sorting, employ crossover and mutation operators to create a new population \( Q_{t+1} \) from the new population \( P_{t+1} \).

Use the binary tournament selection operator based on the crowded comparison operator to select the parent individuals from population \( P_{t+1} \).

The following is the pseudo-code for NSGA-II-MSONDP.

**Algorithm: NSGA-II-MSONDP**

**Input:**
- An undirected graph \( G_1 = (V_1, E_1) \)
- A complete and undirected and weighted graph \( G_2 = (V_2, E_2, c) \)
- A set of connection demands \( T = \{t_i|t_i = (L_i^1, L_i^2), i = 1, \ldots , |T|\} \)

**begin**
1. Initialize the parent \( P_1 \) with \( |P_1| = N \) and generate number \( t = 1 \)
2. while \( t < \) the terminate generation number
3. Combine the parent and offspring population via \( R_t = P_t \cup Q_t \)
4. Sort all solutions of \( R_t \) to get all non-dominated fronts \( F = \text{fast-non-dominated-sort}(R_t) \)
   where \( F = (F_1, F_2, \ldots) \)
5. set \( P_{t+1} = \emptyset \) and \( i = 1 \)
6. while \( |P_{t+1}| + |F_i| < N \)
7. calculate crowding distance of \( F_i \)
8. add the \( i^{th} \) non-dominated front \( F_i \) to the parent population \( P_{t+1} \)
9. \( i = i + 1 \)
10. end while
11. sort the \( F_i \) according to the crowding distance
12. fill the parent population \( P_{t+1} \) with the first \( (N - |P_{t+1}|) \) elements of \( F_i \)
13. generate the offspring population, \( Q_{t+1} \)
14. set \( t = t+1 \)
15. end while
16. the population \( P \) is the non-dominated solution.

**end**

V. EXPERIMENTAL RESULTS

A. Problem Instances

Both random and real world instances were used in the experiments. Random instances were taken from the TSP library [7] and real world instances were based on Borne’s Gravitory model [1].

Problem instances are denoted as name\(_{n2\_n1\_k}\), in which name is “a” if it is a random instance and is “g” or “Germany” if it is a real world instance; \( n2, n1 \) is the number of nodes in \( G_2 \) and \( G_1 \), respectively; and \( k \) is the number of demands.

We experimented with 32 random and 25 real world instances. The smallest data set employed was name6_4_2 and the biggest was name100_95_40 for both random and real world instances.

B. System Setting

The population size was 500. The number of generations was 600, the crossover rate in each generation was 40% (each of the crossover types’ rate is 20%) and mutation rate is 20% (“chromosome renew” mutation rate is 10%, and that of the others 5% each).

The system is executed 10 times for each problem instances. The program was run on a machine with Intel Core i5, 4 GB RAM, Windows 7 Ultimate, and was implemented in JAVA.

C. Computational Results

Figures 4–9 show the non-dominated solutions obtained by NSGA-II-MSONDP. The horizontal axis of each graph represents the cost (totalCost) and the target vertical axis represents the number of connections on one side (maxConn).
Obviously, no completely optimal solution could be found for M-MSONDP with the multi-objective approach.

Furthermore, the result produced by the multi-objective algorithm is a set of solutions instead of the best. In this case, the higher the totalCost, the lower is the maxConn, and vice versa. This means the selection of the solution depends on specific circumstances; for example, to design a survivable network in areas having a high risk of failures, where the impact of these failures is simultaneously large and difficult to overcome in a short time, a solution with a small maxConn should be considered.

In addition, this paper makes comparisons with the results presented in [13] in terms of the network cost (totalCost), which aims to evaluate the effectiveness of the multi-objective approach versus the single-objective approach to this problem. The comparison between the result found by PGAMSONDP [13] and NSGA-II-MSONDP can be found in Figures 10 and 11. On small real world and random instances, the results (cost) found by NSGA-II-MSONDP are better than the cost found by PGAMSONDP. There is slight difference between them. On large instances, the results (cost) found by PGAMSONDP are marginally better than the cost of NCGA-II-MSONDP.

VI. CONCLUSION

This paper proposed a multi-objective approach for M-MSONDP, a model of SNDP, and a multi-objective genetic algorithm to solve it. We experimented on 32 random and 25 real world instances. The results showed that our multi-
objective approach and our proposed algorithm were effective. This approach can be used to derive many methods to design a survivable network that is more practical owing to the fact that cost is not usually the only objective.

In the future, we first intend to improve the running time of NSGA-II-MSONDp using a distributed recursive wave. We will experiment with other objectives for this problem, as well as for M-MSONDp, in the hope of finding better survivable network designs.

ACKNOWLEDGEMENT

This work was partially supported by the project “Models for the next generation of a robust Internet,” Grant number 12/2012/HĐ-NDT, funded by the Ministry of Science and Technology, Vietnam.

REFERENCES


BIOGRAPHIES

Dr. Huynh Thi Thanh Binh is now a lecturer at the School of Information and Communications Technology, Hanoi University of Science and Technology (Vietnam). Her research focuses on computational intelligence (heuristics and genetic algorithms). She is now the treasurer of IEEE Vietnam. She is also an ACM member.