2005

K Pair of disjoint paths Algorithm for Protection in WDM Optical Networks

Quoc Viet Phung
Edith Cowan University

Daryoush Habibi
Edith Cowan University

Hoang N. Nguyen
Edith Cowan University

Kungmang Lo
Edith Cowan University

10.1109/APCC.2005.1554044

2005 Asia-Pacific Conference on Communications, Perth, Western Australia, 3 - 5 October 2005.

K Pairs of Disjoint Paths Algorithm for Protection in WDM Optical Networks

Quoc V. Phung, Daryoush Habibi, *Senior Member*, *IEEE*, Hoang N. Nguyen, and Kungmeng Lo
School of Engineering and Mathematics
Edith Cowan University
Perth - Australia

***Abstract*— Survivable Routing in Wavelength Division Multiplexing (WDM) optical networks has been proven to be NP-hard problem. There is a trade-off between the *computational time* and the *optimality* of solutions in existing approaches to the problem. Existing heuristic approaches purely based the graph algorithms are efficient in computational time but do not offer optimal solutions and may fail in some cases even when a solution exists. Meanwhile, the Integer Linear Programming (ILP) models offer optimal solutions but are intractable even with moderate scale networks. In this paper, we introduce a new algorithm for finding K pairs of disjoint paths which are employed as K candidate pairs for each connection in the ILP models. We introduce an ILP model for dedicated path protection in which the number of decision variables is mainly dependant on traffic requests and the constant K, not on the network size.**

I. INTRODUCTION

Survivability in WDM optical networks is defined as the capability of the network to maintain the quality of services against network failures. Most studies in the literature have focused on survivability mechanisms against single failures [1], [2] in which the affected working routes of connections, referred to as the working paths, are rerouted through their alternative paths, referred to as the backup paths. Routing to find such working/backup paths is referred to as the survivable routing (SR). For optimization purposes, pairs of working/backup paths of traffic connections are joined and routed simultaneously using Integer Linear Programming (ILP) models [1], [2] to obtain optimal solutions. This approach, however, is intractable even with moderate scale networks because the number of decision variables and constraints quickly increases with network size and the number of traffic connections required, and hence, the number of searching combinations is explosive.

Possible alternative approaches for survivable routing are based on graph algorithms. The preliminary approach is known as two-step approach in which the working path is first determined over the original network, then the backup path is determined over residual network, being the network resulting from the removal of working path from the original network. Both the working and backup paths are usually determined using shortest path algorithms such as Dijkstra's algorithm and Bellman-Ford's algorithm [3]. The two step approach is simple in both theory and implementation. However, there is no guarantee that the total cost of the found pair of disjoint paths is minimum. In addition, this approach may fail in some network topologies such as the "trap topology' [4]. The one-step approach [5], [6] introduced as an alternative solution. This approach - as the name suggests - implies determination of both working path and backup path simultaneously. One-step approach can resolve the above disadvantages of two-step approach. However, since these approaches provision traffic connections sequentially without backtracking, a non-optimal solution or no solution may be found even when an optimal solution does exist in joint approaches such as ILP formulation. Finding K pairs of shortest disjoint-path as candidate path-pairs for a connections is a possible approach to improve the optimality of solutions.

In this paper, we first propose an algorithm for finding K disjoint-path pais between source and destination nodes of network connections, named as KDPPs. Next, a suitable candidate route will be selected for each connection using an ILP formulation for dedicated path protection, named as ILPS. This approach can combines the computational advantages of graph algorithms and the optimality of solutions of the ILP solver with small number of integer variables and constraints. The benefits of our approach are as follows: the time complexity in our approach is taken to be the total computational time of the KDPPs and ILPS steps. The number of decision variables in ILPS does not change with network size and the number of constraints is a linear function of the number of network links with slope 1. Therefore, the size of ILPS formulation in our approach does not increase dramatically with the network size, and hence the computational time of ILPS will follow the same while the computational time of the KDPPs step is polynomial. The optimality of solutions can be controlled by the value of constant K in the first step. Even when the wavelength resource is much larger than the traffic demand, all optimal solutions can be achieved when $K = 1$ or $K = 2$.

The rest of this paper is organized as follows. Section II reviews solutions to the survivable routing problem. We propose our heuristic approach in Section III. Next, simulation results of the proposed approach are presented and compared with the ILP formulation in Section IV. Finally, Section V summarizes the paper and proposes some future directions for this research.

II. RELATED WORK

ILP formulations have used to obtain optimal solutions for survivable routing. Two categorizes of ILP model have

0-7803-9132-2/05/$20.00 ©2005 IEEE.

existed in the literature, namely link-formed and path-formed formulations. The number of variables in link-formed [7] is dependant on the number of network links and the number of required traffic connections and hence quickly increases with those factors. The ILP models in path-formed [2], [4], [1], in contrast, are based on candidate routes that are pre-determined for each connections. The number of variables in this model is only dependant on traffic connections and candidate routes, not dependent on the networks size (network links and nodes). However, the optimality of the solutions depends on the pre-determined candidate routes for primary and backup paths. Mauz in [2] proposed a unified ILP formulation for node/link failures and dedicated/shared protection schemes in which there are k_1 candidate primary routes and k_2 candidate backup route for each connection. The ILP models in [4], [1], on the other hand, required K pairs of primary/backup candidate routes for each connection. The task of the ILP models in this approach is to select appropriate routes for connections. These models, however, did not mention strategies for finding pre-determined candidate routes.

Alternative, solutions to the survivable routing can be achieved using heuristic approaches purely based on graph algorithms. Finding a pair of disjoint paths for each traffic connection is key issue in survivable routing, which has been extensively studied in literature [5], [6], [8]. The approach is, however, only appropriate for dynamic routing where the knowledge of future connections is unknown. In protection schemes when a set of traffic connections is given, the K disjoint-paths ($K \geq 3$) in [6], [9] offers more choice for finding optimal solutions. To our best knowledge, there is no approach that mentions to finding the K pairs of disjoint-paths as K candidate pairs of primary/backup paths for connections.

III. Propose our approach

In this section, we propose an algorithm for finding a set of K pairs of disjoint-paths between a source node and destination node. Using this algorithm, we find sets of candidate routes for all required connections. Since a pair of disjoint-paths, referred to as a path-pair, uses a link in the network once and only once, we model the sets of candidate routes in path-formed of the path-pairs. The number of variable for a given traffic pattern depends only on the value of constant K. This constant can be used to balance the tradeoff between the computational time and the optimality of solutions.

Our approach for dedicated path protection includes two steps: the K disjoint-paths pairs (KDPPs) and the Integer Linear Programming Selection (ILPS) formulations. We shall now introduce the notation used in these steps.

A. Notations

- Let a undirected graph $G(V, E)$ be a physical network topology, where $V = \{v_1, v_2, \ldots, v_N\}$ is the set of N vertices representing network nodes, and $E = \{e_1, e_2, \ldots, e_M\}$ is set of M undirected edges representing the bi-directional optical fibers.
- Let W be the weight of each link e_l, representing the maximum number of available wavelengths on the link.
- Let pr be a pair of disjoint-paths between two nodes in G, denoted by $pr = [b_1, b_2, \ldots, b_M]$, where b_l is the link indicator constant of link l, given by:

$$b_l = \begin{cases} 1 & \text{if path-pair } pr \text{ uses link } e_l \\ 0 & \text{otherwise} \end{cases}$$

- Let c be the cost of pair pr, determined by

$$c = \sum_{l=1}^{M} b_l$$

  In our model, the cost of a path-pair is defined as the number of wavelength channels taken by that path-pair.
- Let $T = \{t_1, t_2, \ldots, t_D\}$ be the set of D traffic requirements (traffic connections) over the network, where t_i denotes the connection between node pair (s_i, d_i).
- K denotes the number of shortest paths between a connection node pair.
- $PR_i = \{pr_i^{(1)}, pr_i^{(2)}, \ldots, pr_i^{(K)}\}$ is the set of K candidate primary/backup pairs of connection t_i, where $pr_i^{(j)}$ denotes the j^{th} primary/backup pair of connection t_i.

B. K disjoint-path pairs (KDPPs)

Algorithm 1 K Disjoint-Paths Pairs (KDPPs)

Input : A undirected graph $G(V, E)$, a pair of source and destination nodes (s, d) and K, the number of shortest disjoint-paths pairs required.

Output: A set of K-shortest disjoint-paths pairs.

1: Take a shortest paths between the source s and the target d, denoted by p.
2: Define the direction of each link traversed in p from s toward d as positive.
3: Remove all directed links on the shortest path p and replace them with reverse direction links by multiplying the original link cost with -1.
4: Find K least cost paths from s to d in the modified graph using an algorithm given in the Appendix. Denote them as the set of paths $S = \{s_1, s_2, \ldots, s_K\}$.
5: For each pair of paths (p, s_i), remove any link of the original graph traversed by both p and s_i. These are called interlacing links. Identify all path segments identified by the link removal from path p and s_i. These form K disjoint path-pairs, denoted as $\{(w_1, r_1), (w_2, r_2), \ldots, (w_K, r_K)\}$.

Given a physical topology $G(V, E)$ of a network, a set of D traffic connections denoted by T, and a constant $K \in \mathbb{Z}^+$, the main objective of KDPPs is to compute D sets of K disjoint path-pairs (disjoint lightpath path-pairs) corresponding to D traffic connections. In this part, we propose a new algorithm for finding such K pairs of disjoint-paths between any two nodes in the networks. This algorithm is an enhancement of

the one-step approach. We refer this algorithm to as the one-step K disjoint-paths pairs (KDPPs). A brief description of KDDPs is stated in Algorithm 1.

We prove that this algorithm can yield K disjoint-paths pairs. Since a path pair found by Algorithm 1 was proven to be disjoint in [6], we only need to prove that the K found pairs are not coincide each others.

Proof: Let $P = \{P_k | k = [1 \dots K]\}$ be a set of K pairs of disjoint paths yielded from Algorithm 1, where $P_k = \{w_k, r_k\}$ is the k^{th} pair.

Let $P_i = \{w_i, r_i\}$ and $P_j = \{w_j, r_j\}$ be disjoint-paths pairs yielded from the first shortest path p and two arbitrary paths s_i and s_j ($\forall i, j \in [1 \dots K]$), respectively. We prove that P_i and P_j are not coincide.

Since s_i and s_j are not coincide, there exists at least one edge e_k contained in s_i but not contained in s_j. We prove that e_k only belongs to either P_i or P_j. If e_k is contained in p then e_k is obviously not contained in P_i. However, since e_k is not contained in s_j, e_k is not a common link of p and s_j, and hence e_k is in P_j. Conversely, if e_k is not contained in p, then e_k is in P_i. In addition, since e_k is not contained in both p and s_j, e_k is not contained in P_j. Therefore, P_i and P_j are not coincide and the K found disjoint-paths pairs are not coincide each others.

■

The time complexity of the one-step approach in Bhandari's algorithm is $O(M + N \log N)$ [6] and the K shortest path algorithms is $O(K \times M)$, and hence, the time complexity of the algorithm is $O((K+1)M + N \log N)$ which is polynomial.

We now present a path-form of sets of K candidate routes over which the number of variables for dedicated path protection is the same with those without protection.

1) Path-formed model: Let PR represent the path-formed constant matrix for pairs of primary/backup candidate routes. This matrix is of KD rows and M columns. We represent this matrix as follows:

$$PR = \begin{pmatrix} PR_1 & PR_2 & \dots & PR_D \end{pmatrix}^T, \quad T : \text{Transpose}$$

where $PR_i = \begin{pmatrix} pr_i^{(1)} & pr_i^{(2)} & \dots & pr_i^{(K)} \end{pmatrix}^T, \forall i \in [1..D]$, are sub-matrices of $(K \times M)$ in which row $pr_i^{(j)}$ represents the j^{th} path of connection t_i, and is expressed as:

$$pr_i^{(j)} = [b_{p,1}^{(ij)}, b_{p,2}^{(ij)}, \dots, b_{p,M}^{(ij)}]$$

Note that in our study, path $p_i^{(j)}$ is modeled using link indicator constants $b_{p,l}^{(ij)}$, where $b_{p,l}^{(ij)}$ is defined as:

$$b_{p,l}^{(ij)} = \begin{cases} 1 & \text{if pair primary/backup } pr_i^{(j)} \text{ uses link } e_l \\ 0 & \text{otherwise} \end{cases}$$

Therefore, sub-matrices P_i are represented as:

$$PR_i = \begin{pmatrix} b_{pr,1}^{(i1)} & b_{pr,2}^{(i1)} & \dots & b_{pr,M}^{(i1)} \\ b_{pr,1}^{(i2)} & b_{pr,2}^{(i2)} & \dots & b_{pr,M}^{(i2)} \\ & \dots\dots\dots & & \\ b_{pr,1}^{(iK)} & b_{pr,2}^{(iK)} & \dots & b_{pr,M}^{(iK)} \end{pmatrix}, \forall i \in [1..D]$$

2) The path cost matrix: Path cost matrices denote the cost of primary/backup candidate pairs. The entries of the matrix is the number of wavelengths used in all possible lightpaths of primary and backup paths. This matrix is employed to model the objective function and constraints in ILPS.

Let C_{PR} be the path cost matrix of candidate primary/backup pairs. This matrix is $(D \times K)$ matrix, and given by:

$$C_PR = \begin{pmatrix} c_{1,1}^{pr} & c_{1,2}^{pr} & \dots & c_{1,K}^{pr} \\ c_{2,1}^{pr} & c_{2,2}^{pr} & \dots & c_{2,K}^{pr} \\ & \dots\dots\dots & & \\ c_{D,1}^{pr} & c_{D,2}^{pr} & \dots & c_{D,K}^{pr} \end{pmatrix}$$

and, where $c_{i,j}^{pr}$ denotes the number of wavelength channels taken by primary/backup path-pairs $pr_i^{(j)}$.

C. ILPS formulation

The goal of ILPS formulation is to select suitable path-pairs from outcomes of the KDPPs step. The selection process has to satisfy the following conditions:

- The objective function f is to minimize the total wavelength channels required.
- For each connection, only one path-pair is selected.
- The total number of wavelength channels used per link does not exceed the link capacity.

Let $x_i^{(j)}$ be a path-pair indicator variable defined as:

$$x_i^{(j)} = \begin{cases} 1 & \text{if } t_i \text{ uses path-pair } (pr_i^{(j)}) \\ 0 & \text{otherwise} \end{cases}$$

The objective of the ILPS is given by:

$$f = \sum_{i=1}^{D} \sum_{j=1}^{K} c_{i,j}^{pr}$$

This formulation is subject to the following constraints:

1) Capacity constraint:

$$\sum_{i=1}^{D} \sum_{j=1}^{K} b_{pr,l}^{(ij)} x_i^{(j)} \leq W, \qquad \forall l \in E$$

2) Selection constraint:

$$\sum_{j=1}^{K} x_i^{(j)} = 1, \qquad \forall i \in [1..D]$$

3) Integer constraint:

$$x_i^{(j)} = \{0, 1\}, \qquad \forall i \in [1..D], j \in [1..K]$$

IV. Simulation Results

We examine our heuristic approach proposed in Section III in terms of two performance metrics: *the time complexity* and *the optimality of solutions*. Since the ILP formulation offers optimal solutions for survivable wavelength routing problem, we use these metrics as a comparison between our approach and the original ILP formulation. The ILP solver based on LP-relaxation and *branch and bound* techniques [10], [11]

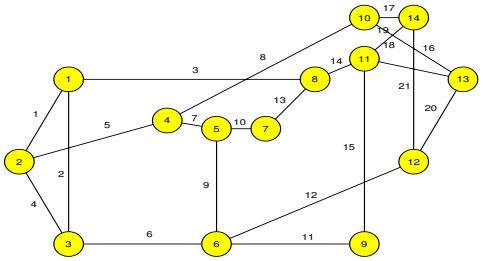

Fig. 1. NFSNET topology

is developed in MATLAB environment. The simulation is performed over the typical topology NSFNET (the National Science Foundation Network) with $N = 14$ nodes and $M = 21$ links as in Fig. 1. This is modeled as an undirected graph in which the capacity of each link is $W = 16$ wavelength channels. The value of K is varied in range $[1 \dots 5]$ to compare solutions.

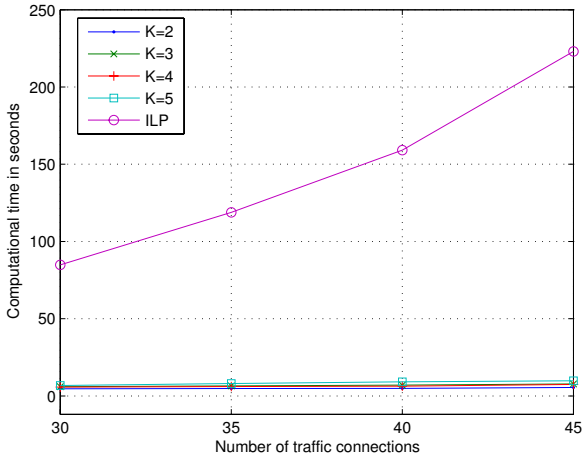

Fig. 2. Computational time versus the number of traffic connections

1) Time complexity:: The computational time is measured in different scenario of traffic connections with a fix physical topology (NSFNET topology). The number of traffic connection D is varied from 30 to 45. Fig. 2 shows the simulation results comparing our approach with the ILP formulation. For each value of D, we randomly generate 100 traffic matrices and measure the computational time as the average computational time for these matrices. We observe an outstanding improvement in computational time of our approach compared to the classical ILP approach. The time complexity for the ILP rapidly increases almost exponentially with the number of traffic connections while time complexity curves in our approach stay nearly flat. The computational time of the ILP formulation for $D = 30$ connections is around 85 seconds and quickly increases to about 230 seconds when D=45, while the

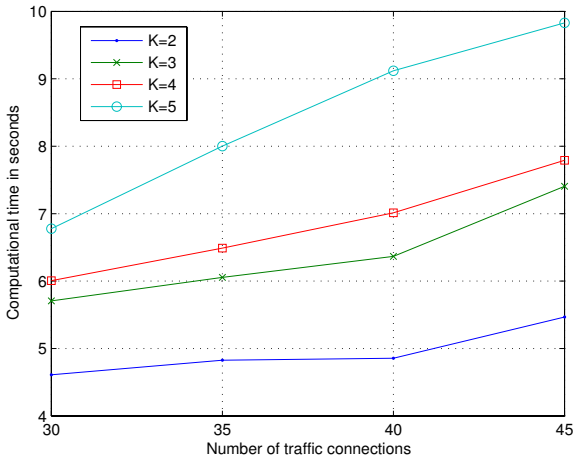

Fig. 3. Computational time versus the number of traffic connection - KDPPs cases

increase of those in our approach is not significant staying in the range of $[4 \dots 9]$ seconds as shown in Fig. 3. In addition, it is worth noting the small differences for different values of K in our approach. The computational time only increases a couple of seconds for an increase of 1 unit in the value of K.

2) Optimality of solutions:: This simulation is also implemented in the undirected NSFNET in Fig. 1. 50 traffic matrices are randomly generated for D from 30 to 45 connections. Our approach is implemented with $K = 1 \dots 5$ and the outcomes are compared with the ILP formulation.

TABLE I
COMPARING THE NUMBER OF OPTIMAL SOLUTIONS BETWEEN ILP AND OUR APPROACH

	K	Feasible solutions	Optimal solutions
ILP	–	49	49
our proposed approach	1	0	–
	2	10	6
	3	44	39
	4	47	47
	5	49	49

The results are presented in table I in which column 3 shows the number of feasible solutions achieved and column 4 represents the number of optimal solutions out of the 50 randomly generated traffic requirements.

These results show that the number of feasible solutions and optimal solutions generally increase when K increases. With $K = 1$, the objective of our approach is basically to find the shortest disjoint paths between node pairs (s, d) of traffic connections. This is only suitable for low traffic requirements. In fact, for $K = 1$, no feasible solutions are achieved in this simulation. With $K = 2$ our approach yields 10 of the 49 feasible solutions, and yields 44 out of the 49 feasible solutions when $K = 3$. In regard to the optimality of solutions, 100% of the solutions are optimal for $K \geq 4$, while for $K = 2$

and 3 the number of optimal solutions are 6 out of 10 and 39 out of 44 a respectively. In summary, the number of optimal solutions increases monotonically with K. The value of K can be used to control the balance between the optimality of solutions and the computational time.

V. CONCLUSION

Although survivable wavelength routing problem has been extensively studied in the literature, it still remains a critical problem and has been proven to be NP-complete. The time complexity and the optimality of solutions are two conflicting metrics, and it is important to find approaches that can balance these. In this paper we proposed a new algorithm for finding K disjoint-path pairs and applied this to a path-form ILP model for dedicated path protection in which the constant K can be controlled to balance the above metrics. Our approach has achieved significant improvements in terms of time complexity whilst still able to obtain optimal solutions.

The approach used in this paper, however, is still based on ILP solver which is computational expensive. We are now working for a heuristic approach based on graph algorithms to tackle the same problem in which the ILP solver will be replaced by an graph algorithm solver. This allow us to have further investigation in protection schemes (dedicated/shared schemes) in large scale networks.

APPENDIX

In this appendix we introduce an algorithm that allows us to generate K-shortest (minimum) paths. The key idea of this alogrithm is adopted from [12]. However, the algorithm in [12] was applied to directed graphs. In this paper, we base on the idea and develop an alogrithm to applied to undirected graphs that compromise with network models in our study. The pseudo code of this algorithm is stated as follows.

This algorithm runs with the time complexity of $O(K \times M)$, where K is the number of shortest paths required and M is the number of undirected edges in graph G.

Algorithm 2 K-Shortest Paths - KSPs

Input : An undirected graph $G(V, E)$, a pair of source and destination nodes (s, t), and K: the number of shortest paths required.

Output: A set of K-shortest paths over the network G from s to t.

1: To assure the possible repetition of the algorithm in a path between a pair of source-destination nodes (s, t), the given network is enlarged with a super source node S, and super destination nodes T, with zero cost arcs (S, s) and (t, T). We find the shortest tree from source node s to other nodes in the network and mark the shortest path $p_1 = \{s_0(= s), s_1, \ldots, s_{r-1}, s_r(= t)\}$ from s to t as the first shortest path.

2: Determine the first node s_h in p_1 such that s_h has more than a single incoming arc. If s'_h, of which the incoming arcs are the incoming arcs of s_h, except those coming from s_{h-1}, does not exist, then generate the node s'_h, else determine the next node s_i in p that has not alternate yet. The shortest path from s to s'_h $(d(s, s'_h))$ is calculated as:

$$d(s, s'_h) = \min_x\{d(s, x) + d(x, s'_h)\}$$

where (x, s'_h) are incoming arcs of s'_h.

3: For each $s_j \in \{s_i, \ldots, s_{r-1}\}$, generate s'_j following the same rules as s'_h, but with one more incoming arc of (s'_{j-1}, s'_j). Clearly, the shortest path from s to s'_j is the second shortest path from s to s_j. Therefore $p_2 = \{s_0, \ldots, s'_i, \ldots, s'_{r-1}, s_r(= t)\}$ is the second shortest path. Repeat step 2 for the shortest path $p_k(k = 2, 3, \ldots)$ to find the next shortest path until $k = K$.

REFERENCES

[1] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, "Survivable WDM mesh network," *Journal of Lightwave Technology*, vol. 21, no. 4, pp. 870–883, 2003.

[2] C. Mauz, "Unified ILP formulation of protection in mesh networks," in *Proceedings of the 7th International Conference on Telecommunication COMTEL)*, vol. 2, pp. 737–741, 2003.

[3] E. Horowitz, S. Sahni, and S. Rajasekaran, *Computer Algorithms*. Computer Science Press, 1998.

[4] W. D. Grover, *Mesh-based Survivable Networks: Options and Strategies for Optical, MPLS, SONET/SDH, and ATM networking*. Prentice Hall PTR, 2004.

[5] J. Surballe, "Disjoint paths in a network," *Networks*, vol. 4, pp. 125–145, 1974.

[6] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing*. Kluwer Academic Publishers, 1999.

[7] M. Pioro and D. Medhi, "Routing flow, and capacity design in communication and computer networks," in *The Morgan Kaufmann Series in Networking* (D. Clark, ed.), pp. 43–45, Morgan Kaufmann, 2004.

[8] C. Xin, Y. Ye, S. S. Dixit, and C. Qiao, "A joint lightpath routing approach in surviavble optical networks," *Optical Networks Magazine*, vol. 3, no. 3, pp. 13–20, 2002.

[9] M. Pioro and D. Medhi, *Routing flow, and Capacity Design in Communication and Computer Networks*. The Morgan Kaufmann Series in Networking, Morgan Kaufmann, 2004.

[10] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, pp. 497–520, 1960.

[11] S. Zionts, *Linear and Integer Programming*. Prentice-Hall International Series in Management, 1974.

[12] E. D. Q. V. Martins and J. L. E. D. Santos, "A new shortest paths ranking algorithm," *Investigao Operacional*, vol. 20, no. 1, pp. 47–62, 2000.