Exact Algorithms for Robust $k$-Path Routing Problems

April K. Andreas$^1$ and J. Cole Smith$^2$

$^1$Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ, april@email.arizona.edu
$^2$Department of Industrial and Systems Engineering, The University of Florida, Gainesville, FL

Abstract

Many routing problems in the literature examine stochastic problems in which costs and/or travel times are the uncertain elements. However, an implicit assumption is often made that all arcs and nodes operate with perfect reliability. Suppose instead that there exist independent arc failure probabilities. We may wish to ensure that a path exists from some origin to some destination that survives with a certain threshold probability. One manner of doing this is to construct $k$ origin-destination paths at a minimum cost, while ensuring that at least one path remains operational with a sufficiently large probability. However, the reliability constraint induces a nonconvex feasible region (even when the integer variable restrictions are relaxed). We begin by examining the Robust Two-Path Problem, which seeks to establish two paths between a source and destination node wherein at least one path must remain fully operable with some threshold probability. We consider the case where both paths must be arc-disjoint and the case where arcs can be shared between the paths. We prove that these problems are both strongly NP-hard, and then examine various strategies for solving the resulting nonlinear integer program, including pruning, coefficient tightening, lifting, and branch-and-bound partitioning schemes. We then examine the Robust $k$-Disjoint-Path problem, and note that the previous scheme for two-path problems does not generalize well to the $k$-path situation. Instead, we propose an alternative model for the problem in which the variables correspond to paths and the nonconvex reliability constraints are replaced by an exponential set of linear constraints. Accordingly, we propose a branch-and-cut-and-price algorithm for handling the new problem.

Keywords: Integer programming, nonlinear programming, branch-and-bound, branch-and-price-and-cut.

1. Introduction

Traditional routing problems, such as shortest path and travelling salesman problems, assume perfect operability of all arcs and nodes. However, real-world applications often call for a more diverse routing strategy. For example, consider a cell phone company that has determined that its customers will tolerate having up to 5% of their calls dropped. The cell phone company can respond by either sending each call over one highly reliable path, or by sending two signals per call along less-reliable paths and requiring that at least one signal reaches the destination with a probability of 95%. It might be cheaper for the company to buy two relatively unreliable paths than to buy one highly-reliable path. Such redundant routing takes place in Synchronous Optical Networks, for instance, by routing along rings such that a single line break will not affect client service. Another application is in achieving military missions, such as destroying a particular target which can be accessed via any number of paths, each path with its own risks. A commander may choose to deploy two groups along different paths such that at least one of the groups will arrive at and destroy the target within some acceptable level of reliability.

Hence, when these independent arc failure probabilities exist, a secondary constraint would retain some measure of expected functionality, introducing nonlinear, nonconvex constraints.
In this session, we introduce mathematical programming techniques to impose these requirements on path selection problems on a directed graph $G(N, A)$, where $N$ is the set of nodes $\{1, \ldots, n\}$ and $A$ is the set of arcs.

Single shortest path reliability problems have received much attention in the literature, both in their methodological development and in their applications. Bard and Miller [2] address a research and development project selection problem, in which spending additional money on projects could increase their probability of success. Zabarankin, Uryasev, and Pardalos [12] consider a related problem in optimizing path risk in the context of aircraft flight trajectory through a threat environment. Elimam and Kohler [4] describe some unique applications of the resource-constrained shortest-path problem, such as the determination of optimal wastewater treatment processes and thermal resistance of building structures.

Too many papers exist to cite on multiple-path routing; we briefly mention two of the most relevant ones here. Suurballe [9] presents a polynomial-time labeling algorithm to find $k$ node-disjoint paths between two given nodes. Fortune, Hopcroft, and Wyllie [5] provide an important characterization of NP-complete vertex-independent routing problems on directed graphs via the graph homeomorphism problem.

This problem can also be viewed in the context of a network survivability problem. In networks for which survivability is of concern, an assumption is made that the probability of two arcs failing is sufficiently small. Hence, they create active and backup paths that do not share any arcs. (If vertices are subject to failure as well, the paths must be node-disjoint as well, except at the origins and destinations.) Therefore, the objective is to minimize the total bandwidth reserved to meet a given demand, allowing the sharing of the backup links among disjoint connections. Li et al. [6] approach the problem in terms of networking for multi-protocol label switched networks, developing extensions to the signaling protocol to distribute additional link usage information. The authors also explain how to extend the algorithm to account for single node failures (multiple link failures) and fiber span failures. Several other groups explore this problem, including Liu and Tipper [7], Xu, Qiao, and Xiong [10], and Yee and Lin [11].

2. Models and Algorithms

The authors present their research on two-path routing problems in [1]. We can begin this analysis by examining a shortest path problem with the side constraint that the probability of successfully traversing the path is at least as large as some threshold probability. The natural formulation of this problem is a mixed-integer nonlinear program, though with some care the problem can be transformed to a mixed-integer linear program. In any case, these side constraints prevent a polynomial-time application of Dijkstra’s algorithm, and make this problem ordinarily NP-hard.

An even more interesting and applicable version of this problem seeks to establish two arc-independent paths between two nodes subject to some minimum reliability measure. Many practical scenarios simply require that at least one path survives between a pair of nodes, a problem which we call the Robust Two-Path Problem with Disjoint arcs (RTP-D). In this case, we stipulate that the probability that at least one path survives is sufficiently large. This probability is given by one minus the probability that both paths fail, which is an inherently nonlinear constraint. Even worse, such a constraint induces a nonconvex feasible region. The problems are aggravated further if we allow high-reliability, low-cost arcs to be shared between the paths (RTP-S). Note that in this case, the failure of a shared arc causes both paths to fail. We show in [1] that both RTP-D and RTP-S are strongly NP-hard.

Let the network be denoted by $G(N, A)$, and let the cost and reliability of each arc $(i, j) \in A$ be given by $c_{ij}$ and $p_{ij}$, respectively. Define $FS(i)$ and $RS(i)$ to be the forward and reverse stars of node $i \in N$, respectively (i.e., the set of arcs exiting and entering node $i$). Finally, let $\tau$ be the minimum permissible reliability that at least one path survives.
We can define \( x_{ij}^q \) for all \((i,j) \in A\) and \( q = 1, 2 \) to be a binary variable equal to one if path number \( q \) utilizes arc \((i,j)\), and zero otherwise. Also, let variables \( s_i^n \) represent the probability that path \( q \) successfully reaches node \( i \) from node 1, given that path \( q = 1, 2 \) visits node \( i \). The RTP-D can be thus modelled as a union of minimum cost flow problem constraints, linear path-probability calculation constraints, and a single nonlinear, nonconvex constraint as follows.

\[
\begin{align*}
\text{minimize} & \quad \sum_{(i,j) \in A} c_{ij} (x_{ij}^1 + x_{ij}^2) \\
\text{subject to} & \quad \sum_{j \in FS(1)} x_{1j}^q = 1 \quad \forall q = 1, 2 \\
& \quad \sum_{j \in FS(i)} x_{ij}^q = \sum_{h \in RS(i)} x_{hi}^q \quad \forall i \in \{2, \ldots, n-1\}, \forall q = 1, 2 \\
& \quad x_{ij}^1 + x_{ij}^2 \leq 1 \quad \forall (i,j) \in A \\
& \quad s_i^q \leq p_{ij} s_i^q + (1 - x_{ij}^q) \quad \forall q = 1, 2, \forall (i,j) \in A \\
& \quad s_i^1 = s_i^2 = 1 \\
& \quad s_n^1 + s_n^2 - s_n^1 s_n^2 \geq \tau \\
& \quad s_n^1 \geq s_n^2 \\
& \quad x_{ij}^q \in \{0, 1\} \quad \forall q = 1, 2, \forall (i,j) \in A.
\end{align*}
\]

The first part of this formulation is straightforward and standard in the literature. Our strategy in (5) and (6) uses a set of constraints to compute the \( s \)-variables. Note that the path-probability calculation constraints are linear, but employ binary variables that give the problem its combinatorial nature. The nonlinear constraint (7) enforces the condition that at least one path remains survivable with sufficiently large probability. Constraint (8) removes some problem symmetry in the model to improve its solvability.

Our approach to handling the single nonlinear constraint constructs a convex hull relaxation of the feasible region by relaxing the integrality constraints, and by replacing (7) with

\[
(1 - \sqrt{1 - \tau}) s_n^1 + (\sqrt{1 - \tau} - 1 + \tau) s_n^2 \geq \tau (1 - \sqrt{1 - \tau}).
\]

We can then utilize a divide-and-conquer recursive approach to force the final solution to reside within the original nonconvex feasible region. Our approach is to reinstate the integrality constraints and solve the problem by relaxing only the constraint (7) and replacing it with (10). If the resulting solution is feasible to the original problem, an optimal solution has been identified. Else, we can tighten the relaxation by creating a disjunction in which at least one of two cutting planes is valid. With this in mind, we solve the relaxed problem in a branch-and-bound fashion in which each subproblem is an integer program over some particular interval. We provide an illustration of this technique in Figure 1.

We can enhance the model further by using a simple variation of Dijkstra’s algorithm to establish an upper bound on the probability of successfully reaching a given node, as well as a lowest-permissible probability of reaching an intermediate node and still reaching node \( n \) with the required probability. Information about these probabilities can then be used to reduce graph \( G(N,A) \) appropriately for each path. Directed acyclic graphs provide yet another opportunity for pruning, employing a variation on Dijkstra’s algorithm where the least reliable path to (or from) each node is stored. We can also use this information to tighten our constraints over all feasible solutions by using coefficient tightening. Our computational experience supports the strategy of using pruning and coefficient tightening, but shows that using lifting on the constraints is not effective.

If the solution in a given interval is infeasible to the original problem, we must recursively solve the problem over the interval by splitting it up into smaller intervals. There are nu-
merous ways that the interval could be split to prevent the nonlinear-infeasible solution from recurring (Figure 1 shows a “vertical branching” rule). We examine the advantages and disadvantages of these methods and present computational results to support the use of one particular method; specifically, choosing a partitioning for which the distance from the nonlinear-infeasible solution to either partitioning plane is equal, thereby maximizing the minimum distance to either partitioned region. This comparatively effective method of branching does not rely on the axes to generate the cuts, and generalizes the Reformulation-Linearization Technique [8] (for continuous-variables optimization problems) on this particular problem.

Relaxing the edge-disjoint constraints will necessitate adjustments to be made to RTP-D. The formulation and algorithm adjustments to solve the shared-arcs variation of our problem (RTP-S) must then be modified. Our approach now requires the introduction of an additional decision variable and moves our linear chord relaxation into three-dimensional space. (Details are omitted in this paper due to space limitations.) Model tightening via pruning and lifting, as well as preprocessing is also less straightforward for RTP-S, though it is still possible. A specialized partitioning procedure proves difficult, however, for this particular problem due to the fact that joint reliability constraint is now neither convex nor concave. We work around this problem by implementing the Reformulation-Linearization Technique [8] to assist in partitioning our space for the branch-and-bound procedure. We provide computational results to recommend the best pruning, lifting, and partitioning strategies for solving RTP-S.

A natural extension to the two-disjoint-path problem is the k-disjoint-path problem for $k > 2$. The previous approach would suffer due to the increasing difficulties of relaxing multiple nonlinear terms as $k$ increases. An alternative approach is to consider a path-based formulation, in which variables $x_p$ equal to one if path $p \in P$ is selected, and zero otherwise, where $P$ is the entire set of origin-destination paths in the network. However, there are too many such paths to generate, and a column generation approach would have to deal with the nonlinear constraint governing reliability. Instead, we can replace the reliability constraint...
with a set of covers $C^j, |C^j| \geq k$, such that the reliabilities of paths in $C^j$ are too small to be used in a feasible solution. Hence, we can restrict
\[ \sum_{i \in C^j} x_i \leq k - 1 \quad \forall \text{ covers } C^j. \] (11)

However, at this point, we have generated an exponential number of variables and constraints. Accordingly, we employ a branch-and-price-and-cut algorithm to solve the problem. Note that after generating some $m$ covers and set of paths $\overline{P}$, and after relaxing integrality, we have the formulation

\[ \text{Minimize } \sum_{p \in \overline{P}} C_p x_p \] (12)

subject to
\[ \sum_{p \in \overline{P}} x_p = K \] (13)
\[ \sum_{p \in \overline{P}(i,j)} x_p \leq 1 \quad \forall (i, j) \in A \] (14)
\[ \sum_{p \in \overline{P}} a_{rp} x_p \leq k - 1 \quad \forall r = 1, \ldots, m \] (15)
\[ 0 \leq x_p \leq 1 \quad \forall p \in \overline{P}, \] (16)

where $C_p$ is the total cost of a path (sum of its edge costs), $\overline{P}(i,j)$ is the set of paths generated thus far that use arc $(i,j)$, and $a_{rp}$ is a constant equal to 1 if $p$ belongs to cover $r$, and zero otherwise. The column generation phase of the algorithm attempts to find variables that enter the basis of problem (12)-(16), but is complicated by the presence of constraints (15). Those constraints require us to determine whether or not a new path (column) is a member of an existing cover constraint. We can solve this problem in pseudopolynomial time, however, by using algorithms designed for resource-constrained shortest path problems. After this problem is solved, the row generation portion is straightforward. This process continues until no columns or rows can be generated.

Following the row/column generation phase, the branching phase must occur in a manner that forces the algorithm to converge. Simply branching on an $x_p$-variable is problematic, because a variable that is forced to equal to zero will reappear under a different index in the column generation subroutine that follows the branching phase. Hence, we prove that if the previous solution to the linear programming relaxation is fractional, there either exists an optimal solution in which at least one arc has a fractional flow, or an optimal integer solution can be found based on this fractional solution. In the former case, we can ascertain that fractional arc flow, and then branch by insisting that this arc either contains a total flow of either zero or one.

However, the latter case induces dual variables that could force the column generation subproblem to solve a shortest path problem with negative arc costs. While unlikely, this makes the column generation problem strongly NP-hard. Instead, we prescribe a slightly different multicommodity flow model based on the work of Barnhart, Hane, and Vance [3] for our problem. This new model avoids the NP-hard column generation routine, but at the expense of a much larger model with symmetry complications. We will examine the computational effectiveness of these methods in our future research.

3. Summary

We have examined the two-path routing problem with reliability considerations, which turns out to be NP-hard in the strong sense, as opposed to the ordinary NP-completeness of the
single-path problem. Our algorithms for these two-path problems are based on relaxing a complicating nonconvex constraint and providing a (hopefully tight) series of disjunctive convex approximations to the problem. For the multiple path problem, we instead turn to a path-based formulation instead of the arc-based formulation for the two-path problem, and replace the reliability constraint with a set of linear cover constraints. We propose the use of a branch-and-price-and-cut framework to solve these problems. Our future research will involve providing the details of the branch-and-cut-and-price algorithm, and stating computational results comparing the efficacy of using our algorithm on model formulations.

Acknowledgments

The authors gratefully acknowledge the support of the Office of Naval Research under Grant Number N00014-03-1-0510 and the Air Force Office of Scientific Research under Grant Number F49620-03-1-0377.

References