Bicolored Shortest Paths in Graphs with Applications to Network Overlay Design

Hongsik Choi and Hyeong-Ah Choi

Department of Electrical Engineering and Computer Science
George Washington University
Washington, DC 20052
{hongsik,choi}@seas.gwu.edu

Weizhen Mao and Rahul Simha

Department of Computer Science
College of William and Mary
Williamsburg, VA 23185
{wm,simha}@cs.wm.edu

Abstract

This paper considers a graph theoretic problem motivated by a telecommunications application. When a private organization wishes to connect two sites by leasing physical lines from a public telecommunications network, it is often the case that several categories of lines are available, at different costs. Typically, a faster (low delay) line costs more than a slower (high delay) line. In addition, a bound on the end-to-end delay is also often desired, thereby preventing the exclusive use of low cost lines. Overlaying a path on the public network therefore involves two diametrically opposing factors: cost and delay. The following variation of the standard shortest path problem is thus of interest: what is the shortest (cheapest) route between the two sites that meets a given bound on the end-to-end delay. We formulate a graph-theoretic problem that has both a shortest path and a coloring component. The general problem is shown to be NP-complete. Polynomial-time algorithms and approximations are given for some special cases.

Keywords: telecommunications network design, graph theory, shortest paths, routing, overlay networks.

1 Introduction

Today, several private organizations operate wide-area networks that connect various geographically dispersed sites. These sites are typically linked in a point-to-point fashion using physical links rented or leased from long-distance telecommunications carriers. A private wide-area network is thus *overlaid* on a public network. In the future, the availability of physical links for lease from various carriers is expected to increase significantly both because the number of carriers are thought to increase but also, within a single carrier's menu of physical links, an organization can pay for various levels of service. As might be expected, a low delay line costs more than a high delay line. It is not always possible to use low cost (high delay) lines exclusively, since a bound on the end-to-end delay is often desired. With this tradeoff, a natural question thus arises: given certain an end-to-end performance requirement, which physical links should be used to minimize leasing costs while meeting the stipulated performance? For example, if it is desired to maintain an end-to-end delay of one second between sites A and B and if these sites are two nodes in a large network consisting of traditional 64 Kbit and T1 lines (with different leasing costs), what is the cheapest route between A and B that meets the delay constraint?

In this paper, we formulate a graph theoretic problem to address the question raised above – namely, how does one find low cost routes that meet performance constraints? We consider two types of links with attendant costs and performance: high cost links with low delays, and low cost links with higher delays than the high cost links. Our formulation is presented as a combination of a two-color graph coloring problem and a shortest path problem. The general problem is shown to be NP-complete even when the graph is a simple line; it is also NP-complete when the costs and delays have a simple structure. Optimal algorithms are presented for some special cases of interest.

In graph theoretic terminology, we are given a graph $G = (V, E)$ with $|V|$ vertices and $|E|$ edges and two designated vertices s and t between which a path is to be constructed using edges in E. Each edge can be colored either *red* (r) or *black* (b). The choice of color determines the cost of the edge and the delay across the edge. For example, if edge e is colored red, it has delay $d(r, e)$ and cost $c(r, e)$. Thus, a set of four weights for each edge e, $w(e) = (d(r, e), c(r, e), d(b, e), c(b, e))$ describes the edge completely. Finally, an end-to-end delay bound D is also given as input to the problem. We wish to find a path and a coloring of edges on the path such that the total delay along the path is less than D, and the cost of the path is the least among paths that meet the delay constraint. This problem turns out to be NP-complete even when the graph is a simple line, or for general graphs, even when both colorings of an edge result in the same cost and delay. An $O(|V| + |E|)$ algorithm is presented for the case in which all edges are identical. Sometimes, the colorings of some edges are fixed (no choice is available for the physical lines) even with identical edges; for this instance, we provide an $O(|E|^2 + |E||V| \log |V|)$ algorithm. For the special case when G is a tree, we show how our problem can be translated into an equivalent 0/1 knapsack problem [2], for which several approximation algorithms are known.

A vast literature exists on the shortest path problem, with several varieties of the general problem [6], but to the best of our knowledge, the particular problem we study has not been previously considered. Our problem is generally related to other problems involving paths or subgraphs in weighted graphs, such as the Steiner tree and minimum spanning tree problems. Our work also appears to be related to the problem of finding graph spanners [1]: a subgraph G' of graph G is a D-spanner of G, if for every $u, v \in V$, the distance from u to v in G' is at most D times the distance in G. Thus, loosely speaking, spanner problems also have a delay bound, although there is no notion of choice or costs. Another related subgraph problem appears

in [12] in which a spanning tree of minimal weight is desired that satisfies end-to-end delay bounds from a source to all destinations. Our application lies in the area of topological design of telecommunications networks (see [3, 4, 11] and references therein). In particular, there has been related work on constrained spanning trees [11, Chapter 5]. Finally, a special case of our problem can be reduced to a 0/1 knapsack problem [13]. Indeed, our problem may be considered a graph-generalization of a knapsack problem.

The rest of this paper is structured as follows. A precise formulation is given in Section 2 and complexity results are presented in Section 3. Following this, exact algorithms for some special cases are given in sections 4 and 5. Approximation algorithms for trees based on heuristics for the knapsack problem are given in Section 6, before concluding in Section 7.

2 Problem definition

Our graph problem of interest may be formally defined using the following notation.

- $G = (V, E)$ is a connected undirected graph.
- s and t are two designated vertices. We wish to find a path (route) between s and t in G.
- F is a set of edge colors. We will only be interested in the case $F = \{r, b\}$ (r=red, b=black).
- $f : E \to F$ is a coloring that assigns a color to each edge.
- $W(E)$ is a collection of 4-tuples, each element of which represents four non-negative edge weights associated with an edge in E. For each $e \in E$, $w(e) = (d(r, e), c(r, e), d(b, e), c(b, e))$ denotes the 4-tuple associated with e. Here, if edge e is colored red, the delay associated with the edge (link) is $d(r, e)$ and the cost incurred in using the link is $c(r, e)$. The case when e is colored black is similarly defined. Typically, $d(b, e) < d(r, e)$ and $c(b, e) > c(r, e)$, a condition that will be assumed throughout the paper. Intuitively, if $e = (u, v)$ lies on a path from s to t, then we are considering two choices: use a line from u to v with a low delay $d(b, e)$ and a high cost $c(d, e)$ or use a line with a higher delay $d(r, e)$ and lower cost $c(r, e)$. Thus, a coloring of edges along a path from s to t completely determines the cost and performance of the connection from s to t.
- Let $P = P(s, t) \subseteq E$ denote a path from s to t in G.
- D is a desired bound on the overall delay between s and t.
- For a coloring f and path P, define $C(f, P) = \sum_{e \in P} c(f(e), e)$ and $D(f, P) = \sum_{e \in P} d(f(e), e)$.

We are now in a position to formally define the problem: find a path P^* between s and t and a coloring f^* such that (i) $D(f^*, P^*) \leq D$ and (ii) $C(f^*, P^*) \leq C(f, P)$ for any other path P (between s and t) and coloring f satisfying the constraint $D(f, P) \leq D$. Observe that we only need to color the path and not the entire set of edges; however, we retain the general notation for convenience of description.

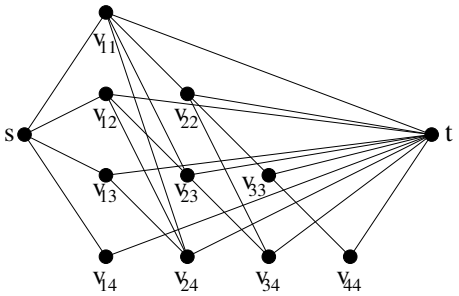

Figure 1: The graph for $n = 4$

3 Complexity results

We now address the complexity of our Bicolored Shortest Path Problem (henceforth BSSP), the decision version of which can be stated as follows: given a graph $G = (V, E)$, weights $W(E)$, with two designated vertices $s, t \in V$, and integers D and C, is there a path P between s and t and a coloring f such that $D(f, P) \leq D$ and $C(f, P) \leq C$? For a general graph, we have the following result.

Theorem 1 *BSPP is NP-complete even when $d(r, e) = d(b, e) = d(e)$, $c(r, e) = c(b, e) = c(e)$ for any $e \in E$. (Coloring can be ignored in this case.)*

Proof. BSPP is obviously in NP since any solution can be verified in polynomial time. Let us consider the NP-complete Subset Sum (SS) [8, 10], in which given a set of positive numbers $A = \{a_1, \ldots, a_n\}$ and a bound $K \geq 0$, the problem asks whether there is a subset $A' \subseteq A$ with $\sum_{a_i \in A'} a_i = K$. We next prove that SS is polynomially reducible to BSPP with $d(r, e) = d(b, e) = d(e)$, $c(r, e) = c(b, e) = c(e)$ for any $e \in E$.

We construct the following instance for BSPP. Let $C > K$ and $D > na$, where $a > \max_i\{a_i\}$. Define the graph $G = (V, E)$, where $V = \{s, t\} \cup \{v_{ij} : i = 1, \ldots, n, i \leq j \leq n\}$ and $E = E_1 \cup E_2 \cup E_3$ with

$E_1 = \{(s, v_{1i})$ with delay $a - a_i$ and cost $a_i : i = 1, \ldots, n\}$,

$E_2 = \{(v_{ij}, v_{(i+1)l})$ with delay $a - a_l$ and cost $a_l : i = 1, \ldots, n-1, i \leq j \leq n-1, j+1 \leq l \leq n\}$,

$E_3 = \{(v_{ij}, t)$ with delay $D + K - ia$ and cost $C - K : i = 1, \ldots, n, i \leq j \leq n\}$.

See Figure 1 for an example of G when $n = 4$ (The delay and cost weights are not shown in the figure). Since $|V| = 2 + \sum_{i=1}^{n} i = 2 + \frac{1}{2}n(n+1)$ and $|E| = n + \sum_{i=1}^{n-1}\sum_{j=1}^{n-i} j + \sum_{i=1}^{n} i = \frac{1}{6}n(n^2 + 3n + 8)$, the graph can be constructed in polynomial time. We next prove that there is $A' \subseteq A$ with $\sum_{a_i \in A'} a_i = K$ if and only if there is a path $P = P(s, t)$ such that $\sum_{e \in P} d(e) \leq D$ and $\sum_{e \in P} c(e) \leq C$.

Assume that $A' = \{a_{i_1}, \ldots, a_{i_h}\}$ with $\sum_{j=1}^{h} a_{i_j} = K$. Without loss of generality, assume that $i_1 < \cdots < i_h$. We define the path P to be s—v_{1i_1}—v_{2i_2}—$\cdots$—v_{hi_h}—t. Obviously,

$$\sum_{e \in P} d(e) = (a - a_{i_1}) + (a - a_{i_2}) + \cdots + (a - a_{i_h}) + (D + K - ha) = D$$

and

$$\sum_{e\in P} c(e) = a_{i_1} + a_{i_2} + \cdots + a_{i_h} + C - K = C.$$

Assume that there is a path P between s and t that satisfies the delay (D) and cost (C) constraints. Because of the special structure of G, P must be s—v_{1i_1}—v_{2i_2}—$\cdots$—v_{hi_h}—t with $i_1 < \cdots < i_h$. Since $\sum_{e\in P} d(e) = (a - a_{i_1}) + (a - a_{i_2}) + \cdots + (a - a_{i_h}) + D + K - ha \le D$, then

$$\sum_{j=1}^{h} a_{i_j} \ge K.$$

Since $\sum_{e\in P} c(e) = a_{i_1} + a_{i_2} + \cdots + a_{i_h} + C - K \le C$, then

$$\sum_{j=1}^{h} a_{i_j} \le K.$$

We must have $\sum_{j=1}^{h} a_{i_j} = K$. Let $A' = \{a_{i_1}, \ldots, a_{i_h}\}$. This is clearly the subset in SS. ∎

Unfortunately, as the following theorem shows, the problem remains NP-complete even if the graph structure is simple.

Theorem 2 *BSPP is NP-complete even when G is a line and $d(b, e) = c(r, e) = 0$ for any $e \in E$.*

Proof. Since it is not difficult to see that BSPP belongs to NP, we will only give a polynomial reduction from a known NP-complete problem, the 0/1 Knapsack Problem [8], in which given $U = \{1, \ldots, n\}$, a size $w_i \in Z^+$ and a value $p_i \in Z^+$ for each $i \in U$, and positive integers M and K, the problem asks whether there is $X = \{x_1, \ldots, x_n\}$ with $x_i \in \{0, 1\}$ such that $\sum_{i=1}^{n} x_i w_i \le M$ and $\sum_{i=1}^{n} x_i p_i \ge K$.

From a given instance of the 0/1 Knapsack problem, we construct an instance of BSPP in the following way. Let $G = (V, E)$ be a graph such that $V = \{v_0, \ldots, v_n\}$ and $E = \{e_i = (v_{i-1}, v_i) : 1 \le i \le n\}$. Let $s = v_0$ and $t = v_n$. For each i, $1 \le i \le n$, let $d(b, e_i) = 0, d(r, e_i) = w_i, c(b, e_i) = p_i$, and $c(r, e_i) = 0$. Set $D = M$ and $C = \sum_{i=1}^{n} p_i - K$.

We now claim that there exists a solution $X = \{x_1, \ldots, x_n\}$ with $x_i \in \{0, 1\}$ for the 0/1 Knapsack such that $\sum_{i=1}^{n} x_i w_i \le M$ and $\sum_{i=1}^{n} x_i p_i \ge K$ if and only if there exists a coloring $f : E \to \{r, b\}$ such that $\sum_{i=1}^{n} d(f(e_i), e_i) \le D$ and $\sum_{i=1}^{n} c(f(e_i), e_i) \le C$.

Suppose there exists such a solution X for the 0/1 Knapsack problem. Let $f(e_i) = r$ if $x_i = 1$ and $f(e_i) = b$ otherwise. We then note that

$$\sum_{i=1}^{n} d(f(e_i), e_i) = \sum_{i=1}^{n} x_i w_i \le M = D$$

and

$$\sum_{i=1}^{n} c(f(e_i), e_i) = \sum_{i=1}^{n} (1 - x_i) p_i = \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} x_i p_i \le \sum_{i=1}^{n} p_i - K = C.$$

Hence, f is a desired solution for the instance of BSPP.

Conversely, let $f : E \to \{r, b\}$ be a coloring such that $\sum_{i=1}^{n} d(f(e_i), e_i) \leq D$ and $\sum_{i=1}^{n} c(f(e_i), e_i) \leq C$. Set $x_i = 1$ if $f(e_i) = r$ and $x_i = 0$ otherwise. Then, observe that

$$\sum_{i=1}^{n} x_i w_i = \sum_{i=1}^{n} d(f(e_i), e_i) \leq D = M$$

and

$$\sum_{i=1}^{n} x_i p_i = \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} (1 - x_i) p_i = \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} c(f(e_i), e_i) \geq \sum_{i=1}^{n} p_i - C = K.$$

This gives the desired solution to the 0/1 Knapsack problem. ∎

The two results above provide some insight into the structure of BSPP. The complexity of the problem arises from both the underlying graph structure as well as the weights: keeping the weights identical does not help (with arbitrary graph structure) and using a simple graph is not sufficient either (with arbitrary weights). This complexity stands in sharp contrast to that of the weighted shortest path problem, which is polynomial with arbitrary graph structure and weights [15]. However, some special cases of practical interest are polynomially solvable exactly or solvable approximately with a low constant bound. We turn to these cases next.

4 Graphs with identical edges

We say that a graph $G = (V, E)$ in BSPP has *identical edges* if for any $e \in E$, $d(r, e) = d_r$, $c(r, e) = c_r$, $d(b, e) = d_b$, and $c(b, e) = c_b$, where d_r, c_r, d_b and c_b are non-negative constants. We observe that if there exists an optimal solution it must be the path between s and t with the fewest edges. ¿From our assumptions on the weights, $d_b < d_r$ and $c_b > c_r$. The following algorithm finds the optimal solution.

Algorithm 1 IDENTICAL-EDGE

Input: *A graph $G = (V, E)$ and weights $W(E)$.*
Output: *Optimal path P^* and coloring f^*.*

1. *Find $P^* = P^*(s, t) = (e_1, \ldots, e_m)$ with fewest edges by Breadth-First-Search;*
2. **if** $d_b|P^*| > D$ **then** *there is no feasible solution;*
3. **elseif** $d_r|P^*| \leq D$ **then** $\forall e_i \in P^* : f^*(e_i) \leftarrow r$;
4. **else**
5. $\quad k \leftarrow \lceil \frac{d_r|P^*| - D}{d_r - d_b} \rceil$;
6. $\quad \forall i \in \{1, \ldots, k\} : f^*(e_i) \leftarrow b$;
7. $\quad \forall i \in \{k + 1, \ldots, m\} : f^*(e_i) \leftarrow r$;
8. **endif**
9. **return** (P^*, f^*);

It is clear that if $d_b|P^*| > D$, there is no solution for the instance, and that if $d_r|P^*| \leq D$, coloring all edges red gives an optimal solution with cost $c_r|P^*|$. To explain the rest of the algorithm, we first show that k is a positive integer no larger than $|P^*|$. Executing line 5, we must have $d_b|P^*| \leq D$ and $d_r|P^*| > D$, which implies that k is a positive integer. To prove $k \leq |P^*|$, assume the contrary. We then have $\lceil \frac{d_r|P^*|-D}{d_r-d_b} \rceil > |P^*|$. So $\frac{d_r|P^*|-D}{d_r-d_b} > \lceil \frac{d_r|P^*|-D}{d_r-d_b} \rceil - 1 \geq |P^*|$. Therefore, $d_b|P^*| > D$, which is impossible. Obviously, k is the smallest number of black edges needed to satisfy the delay constraint. Since $c_r < c_b$, coloring k edges black and the rest red induces the minimum total cost, which is $c_r(|P^*| - k) + c_b k$.

The time complexity of the algorithm, dominated by the Breadth-First-Search in line 1, is $O(|V| + |E|)$. Thus, we have

Theorem 3 *BSPP is solvable in $O(|V| + |E|)$ time when G has identical edges.*

5 Identical edges with restricted colorings

In our application of interest, it is often the case that some physical links do not come with a choice between high performance and low performance lines. Motivated by these instances, we consider the special case when edges have identical weights but the coloring for some edges is fixed (some are immutably colored red and some are immutably colored black). More precisely, for each edge $e \in E$, $w(e) \in \{(d_r, c_r, d_b, c_b), (d_r, c_r, \infty, \infty), (\infty, \infty, d_b, c_b)\}$. The first type of edge cost indicates a colorable edge, the second an edge which is necessarily colored red and the last, an edge which is colored black. In this special case, a path from s to t will typically use some red and black edges that are already colored and colors other edges in the path to meet the performance constraint with minimal cost.

We will use a vertex-labeling algorithm to compute the optimal path and coloring. For each vertex v, define the following two-component label: $a(v) = (a_1(v), a_2(v))$ to be used in the sequel. We adapt Dijkstra's shortest path algorithm (written as a "relaxation" algorithm [5]) to allow computation of shortest paths that meet the delay constraint. For any fixed coloring of all edges, the algorithm Find-Shortest finds the least-cost path from s to t which also meets the performance constraint. Whenever each edge is relaxed, the label at each node v at the end of the edge accurately reflects the least cost from s to that node (in $a_1(v)$) and total delay (performance) from s to that node (in $a_2(v)$). We start with an all black coloring (for colorable edges) and successively color edges red (to make them cheaper); for each such edge colored red, we execute the shortest path algorithm in order to find the least cost path that meets the constraint. If the constraint can no longer be met by any path, we have colored too many edges red and searched past the optimal solution.

Algorithm 2 RELAX $(e,\ f)$

Input: *An edge $e = (u, v)$ and a coloring f.*
Output: *New label for vertex v.*

1. **if** $(a_1(u) + c(f(e), e) < a_1(v))$ **and** $(a_2(u) + d(f(e), e) \leq D)$ **then**
2. $\quad a_1(v) \leftarrow a_1(u) + c(f(e), e)$;
3. $\quad a_2(v) \leftarrow a_2(u) + d(f(e), e)$;
4. $\quad$ *decrease-key*$(v, a_1(v))$;
5. **endif**

Algorithm 3 FIND-SHORTEST (f)

Input: *Coloring f.*
Output: *Cost of least cost path that meets performance bound under f.*

1. $a_1(s) \leftarrow 0$; $a_2(s) \leftarrow 0$;
2. $\forall v \neq s : a_1(v) \leftarrow \infty$; $a_2(v) \leftarrow \infty$;
3. *Priority-queue* $Q \leftarrow \{(v, a_1(v)) : v \in V\}$;
4. $u \leftarrow s$;
5. **while** Q *not empty* **and** $a_2(u) \neq \infty$ **do**
6. $\quad (u, a_1(u)) \leftarrow$ *extract-min*(Q);
7. $\quad$ **if** $a_2(u) \neq \infty$ **then**
8. $\quad\quad$ **for** *each adjacent edge* $e = (u, v)$
9. $\quad\quad\quad$ RELAX(e, f);
10. **endwhile**
11. **return** $a_1(t)$;

Algorithm 4 FIND-OPTIMAL

Input: *A graph $G = (V, E)$ and edge weights $W(E)$.*
Output: *The cost of the optimal path.*

1. *Let* $\bar{E} \leftarrow \{e : w(e) = (d_r, c_r, d_b, c_b)\}$;
2. $\forall e \in \bar{E} : f(e) \leftarrow r$;
3. $min \leftarrow \infty$;
4. **while** $|\bar{E}| > 0$ **do**
5. $\quad cost \leftarrow$ FIND-SHORTEST(f);
6. $\quad$ **if** $cost < min$ **then** $min \leftarrow cost$;
7. $\quad$ *Pick an edge $e \in \bar{E}$ and set $f(e) \leftarrow b$;*
8. $\quad \bar{E} \leftarrow \bar{E} - \{e\}$;
9. **endwhile**
10. **return** *min;*

Let P^* be an optimal path from s to t and f^* be its optimal coloring. Suppose $P^* = e_1 \dots e_k$ has k edges. Next, define the following sets for coloring f:

$$\begin{aligned} P_1 &= \{e \in P^* : w(e) = (d_r, c_r, d_b, c_b)\} \\ P_2 &= P^* - P_1 \\ P_b(f) &= \{e \in P_1 : f(e) = b\} \end{aligned}$$

Then,

$$\begin{aligned} \text{cost}(P^*) &\stackrel{\triangle}{=} \sum_{e \in P_2} c(f^*(e), e) + \sum_{e \in P_1} c(f^*(e), e) \\ &= \sum_{e \in P_2} c(f^*(e), e) + c_b |P_b(f^*)| + c_r |P_1 - P_b(f^*)| \end{aligned}$$

Now, if f' is any other coloring such that $|P_b(f')| = |P_b(f^*)|$, then the cost with f' is also optimal. Next, observe that during the iterations of FIND-OPTIMAL, the colorable edges are colored black one at a time. Thus, at some iteration, the path P^* will have $P_b(f^*)$ black edges. The cost is recorded in the variable *min*. With the addition of each black edge in the loop, the cost can only increase monotonically. Hence, the algorithm finds the optimal cost. Note that the algorithm is easily modified to record the actual paths and colorings in order to output the optimal path and coloring. Note that the loop in FIND-OPTIMAL is executed at most $O(|E|)$ times. For each iteration, each edge is relaxed once and there are $O(|V| \log |V|)$ operations made with the priority queue (using a Fibonacci heap [7], say). Thus, we have

THEOREM 4 *BSPP with identical edges and partially fixed coloring can be optimally solved in* $O(|E|^2 + |E||V| \log |V|)$ *time.*

6 Approximation algorithms for trees

In this subsection, we will first show that BSPP can be reduced to the 0/1 Knapsack problem when the input graph $G = (V, E)$ is a tree with arbitrary values of $d(b, e), d(r, e), c(b, e)$ and $c(r, e)$ for each $e \in E$. Using existing heuristics for the 0/1 Knapsack, we then discuss performance bounds for BSPP. First note that we need only consider line graphs, since the non-negativity of the weights implies the optimal path is simply the set of edges connecting s to t in the tree. In this case, it is only the coloring that needs to be worked out.

Algorithm 5 BSPP-To-0/1 Knapsack

Input: *A graph* $G = (V, E)$ *with* $V = \{v_0, \ldots, v_n\}$ *and* $E = \{e_i = (v_{i-1}, v_i) : 1 \leq i \leq n\}$, $s = v_0$ *and* $t = v_n$, *and weights* $W(E)$.
Output: *An instance to the 0/1 Knapsack problem:* $(w_1, \ldots, w_n)$, $(p_1, \ldots, p_n)$, *and* M.

1. *Let* $f_0 : E \to \{r, b\}$ *such that* $f_0(e_i) = b$ *for all* $e_i \in E$.
2. *Compute* $D_b = \sum_{i=1}^{n} d(b, e_i)$.
3. **if** $D_b > D$ **then** *there is no feasible solution;*
4. **else**
5. $D_0 \leftarrow D - D_b$;
6. $w_i \leftarrow d(r, e_i) - d(b, e_i)$ $\forall e_i \in E$;
7. $p_i \leftarrow c(b, e_i) - c(r, e_i)$ $\forall e_i \in E$;
8. $M \leftarrow D_0$;
9. **endif**
10. **return** $((w_i, \ldots, w_n), (p_1, \ldots, p_n), M)$;

Observe that the initial coloring f_0 defined in line 1 assigns b to each e_i giving larger cost than the optimum. Thus, the problem is to select an edge set $E_0 \subseteq E$ such that by changing the color of selected edges to r, the sum $\sum_{e_i \in E_0} p_i$ of the reduced cost is maximized while the sum $\sum_{e_i \in E_0} w_i$ of the increased time does not exceed D_0. Such a selection is equivalent to solving the 0/1 Knapsack problem using the output of the above algorithm as an input. Let $X = \{x_1, \ldots, x_n\}$ be an optimum solution to the 0/1 Knapsack problem. Then, $e_i \in E_0$ if and only if $x_i = 1$. Therefore, an optimal coloring $f^* : E \to \{r, b\}$ can be defined by setting $f^*(e_i) = r$ if $x_i = 1$ and $f^*(e_i) = b$ otherwise.

In the following we will summarize the results of existing heuristics for the 0/1 Knapsack problem which give various bounds on our BSPP. Given an instance $(w_1, \ldots, w_n)$, $(p_1, \ldots, p_n)$, and M to the 0/1 Knapsack, it is known that (1) there exists an $O(nM)$ time dynamic programming algorithm which finds an optimal solution [15]; (2) there exists an $O(n)$ time greedy algorithm for which the ratio of the optimal solution to the algorithm's output is bounded by 2 [2]; and (3) there exists an $O(kn^{k+1})$ [14] or $O(n + k^2 n)$ [9] time algorithm for which the ratio of the optimal solution to the algorithm's output is $1 + 1/k$. Here, k is a constant that controls accuracy. These results together with Algorithm 5 establish the following theorem.

Theorem 5 *BSPP when the graph* $G = (V, E)$ *is a line can be*

(1) optimally solved in $O(|E|D_0)$ *time, where* $D_0 = D - \sum_{e \in E} d(b, e)$;

(2) solved in $O(|E|)$ *time for which the ratio of the optimal solution to the algorithm's output is bounded by 2; and*

(3) solved in $O(k|E|^{k+1})$ *or* $O(|E| + k^2|E|)$ *time for which the ratio of the optimal solution to the algorithm's output is* $1 + 1/k$.

7 Summary

We have considered a graph-theoretic problem motivated by a telecommunications network design application. The problem was seen to be a mixture of a two-color coloring problem and a weighted shortest path problem. The complexity of the general problem was found to depend on both the graph structure as well as the collection of 4-tuple edge weights. Polynomial-time algorithms were given for some special cases and approximations based on known knapsack heuristics were identified. Future extensions include generalizing the problem definition to consider connecting multiple sets of nodes and heuristics for the general graph.

References

[1] I. Althofer, G. Das, D. Dobkin and D. Joseph. Generating Sparse Spanners for Weighted Graphs. *Discrete and Computational Geometry*, Vol. 9, 1993.

[2] S. Baase. *Computer Algorithms: Introduction to Design and Analysis*. Addison Wesley, 1988.

[3] S.-G. Chang and B. Gavish. Telecommunications Network Topological Design and Capacity Expansion: Formulations and Algorithms. *Telecommunication Systems*, Vol. 1, pp. 99-131, 1994.

[4] S.-G. Chang and B. Gavish. Lower Bounding Procedures for Multi-Period Telecommunications Network Expansion Problems. *Operations Research*, to appear.

[5] T. Cormen, C. Leiserson and R. Rivest. *An Introduction to Algorithms*. MIT Press, 1992.

[6] N. Deo and C. Pang. Shortest-Path Algorithms: Taxonomy and Annotation. *Networks*, Vol. 14, pp. 275-323, 1984.

[7] M. L. Fredman and R. E. Tarjan. Fibonacci Heaps and their Uses in Improved Network Optimization Algorithms. *J. ACM*, Vol. 34, pp. 596-615, 1987.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. WH Freeman, San Francisco, 1979.

[9] O. Ibarra and C. Kim. Fast Approximation Algorithms for the Knapsack and Sum of Subset Problems. *Journal of the ACM*, Vol. 22, pp. 463-468, 1975.

[10] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*. R. E. Miller and J. W. Thatcher (eds.), Plenum Press, New York, 1972.

[11] A.Kershenbaum. Telecommunications Network Design Algorithms. *McGraw-Hill*, New York, 1993.

[12] S. Khuller, B.Raghavachari and N. Young. Balancing Minimum Spanning and Shortest Path Trees. *ACM Symp. Discrete Algorithms*, 1993.

[13] Martello and P. Toth. *Knapsack Problems*. Wiley and Sons, 1993.

[14] S. K. Sahni. Approximation Algorithms for the 0/1 Knapsack Problem. *Journal of the ACM*, Vol. 22, pp. 115-124, 1975.

[15] R. Sedgewick. *Algorithms*. Addison Wesley, 1988.