A Note on the Complexity of Sliding Shortest Paths

B. Engels[1] and G. Pardella[2]

[1] Zentrum für Angewandte Informatik, Universität zu Köln
engels@zpr.uni-koeln.de
[2] Institut für Informatik, Universität zu Köln
pardella@informatik.uni-koeln.de

Abstract. We address a shortest path problem in a given uncapacited and undirected network $N = (V, E)$ with positive edge costs. In addition we are given a single source-destination pair (s, t), a shortest path π_{st} connecting s and t and a new edge $e = (p, q) \notin \pi_{st}$. The task is to find a minimum number of edges $E_c \subseteq E$ and the minimum weight increase for each edge $e_c \in E_c$ such that the shortest path π_{st} between s and t traverses edge e. We show that the problem is NP-hard and give a heuristic scheme for the problem.

1 Introduction

Bhandari [2] introduced a shortest path problem occuring in network administration and routing: Given a network, a source-sink-pair s, t and an edge (p, q), we are interested in the minimum number of edges on which costs have to be increased such that the shortest path from s to t contains (p, q). We demonstrate the NP-hardness of the problem via reduction of the length-bounded cut problem, which gained attention in recent years [3]. This note is organized as follows: In the next section, we give some preliminary definitions and state the problem formally. In Section 3 we present the reduction, in Section 4 we sketch a heuristic scheme and Section 5 concludes the note.

2 Preliminaries and Problem Statement

We consider undirected graphs $G = (V, E)$ with node set V and edge set E and a cost function on the edges, $c : E \to \mathbb{N}_{>0}$. If not stated otherwise, we assume $|V| = n$ and $|E| = m$.

A *simple path*, $\pi = v_1, v_2, \ldots, v_k,\ v_i \in V,\ i \in \{1, \ldots, k\},\ k \leq n$, is a sequence $\{v_1, v_2, ..., v_k\}$ of pairwise different nodes v_i such that (v_1, v_2), (v_2, v_3), $\ldots$, (v_{k-1}, v_k) are edges of G.

A path $\pi_{vw} = v, v_1, v_2 \ldots, v_k, w$ between two nodes $v, w \in V$ of minimum costs $\sum_{e=(v_i,v_j)\in\pi_{vw}} c(v_i, v_j)$ is called *shortest path* between nodes v and w.

2.1 Problemdefinition

Let $N = (V, E)$ be an uncapacited network, a biconnected undirected graph with edge costs. Moreover, let simple shortest paths π_{uv} between all node pairs $(u, v) \in V \times V$ exist. Let $E_c \subseteq E$ be a set of edges such that suitably increasing costs on all edges $e \in E_c$ induces the shortest path $\pi_{st} = \pi_{spqt}$ to be the one traversing through edge (p, q). Being more formal we define

Definition 1 (Sliding Shortest Path Cut Problem). *Given $N = (V, E)$, a source-sink-pair $s, t \in V$ and an edge $(p, q) \in E$, find the minimum cardinality subset $E_c \subseteq E$ of edges such that the shortest path π_{st} in $\bar{N} = (V, E \setminus E_c)$ equals π_{spqt}, i.e. traverses (p, q).*

Moreover we formulate

Definition 2 (Δ Cost Increase). *Given $N = (V, E)$, a source-sink-pair $s, t \in V$, an edge $(p, q) \in E$, and the minimum cardinality subset $E_c \subseteq E$ of edges such that the shortest path π_{st} in $\bar{N} = (V, E \setminus E_c)$ equals π_{spqt}. Minimize the value $\Delta_e \forall e \in E_c$, i.e. the cost increase $c(e) = c(e) + \Delta_e$ for edge $e \in E_c$, such that $\pi_{st} = \pi_{spqt}$.*

The above definitions imply a problem hierarchy where the minimization of edges in E_c outweighs the minimization of the sum of weight increases on these edges. Studying [2] it is not evident if this is intended or if (and how) the solution of the original problem requires a multiobjective optimization. However, in this paper we concentrate on the given problem statement.

2.2 Cut and L-Cut

We call a subset of edges C_e of G an $s - t - cut$ (cut), if no path remains from s to t in $\bar{G} = (V, E \setminus C_e)$. Analoguosly, given a length bound $L \in \mathbb{N}_{>0}$, we call a subset of edges C_L of G an $s - t - L - cut$ $(L - cut)$, if no path π_{st} remains from s to t in $\bar{G} = (V, E \setminus C_L)$ with $|\pi_{st}| \leq L$.

The value (resp. capacity) of a $(L-)cut$ $C_e(C_L)$ is the number of cut-edges (resp. the total capacity of cut-edges, if edge-capacities are not unit, i.e. we are given a capacity function $k : E \to \mathbb{N}_{>0}$). In our case, we will assume unit capacities unless stated otherwise.

We further define

Definition 3 (Minimum Length Bounded Cut). *Given $N = (V, E)$, a source-sink-pair $s, t \in V$ and a length bound $L \in \mathbb{N}_{>0}$, find the minimum cardinality $L - cut$.*

3 Complexity

We show the NP-hardness of the sliding shortest path cut problem by a reduction of the minimum length-bounded cut problem: Given an instance $N = (V, E)$,

$s, t \in V$, unit capacities, a length function on the edges and a length bound $L \in \mathbb{N}_{>0}$, we add two new nodes a, b to V obtaining V' and three edges $(s,a),(a,b),(b,t)$ to E obtaining E' ($N' = (V', E')$), see Figure 1. All edges receive unit capacity and the following weights or lengths: $c(s,a) = c(b,t) = 0,\ (a,b) = L$.

Theorem 1 (Reduction). *Let E_c be the solution to the sliding shortest path cut problem on $N' = (V', E')$ with source-sink-pair s, t and edge (a,b), then $C_L = E_c$ is the solution to the minimum length bounded cut problem on $N = (V, E)$ with source-sink-pair s, t and length bound L.*

Proof. From the construction it is obvious that there is only one path $\pi_{sabt} = sabt$ in $N' = (V', E')$, which has to be chosen as desired shortest path and which has length L. Therefore if $\pi_{sabt} = sabt$ is the shortest $s-t-$ path in N', the selection of E_c must clearly destroy each $s-t-$path in the rest graph $(V' \setminus \{a,b\}, E' \setminus \{(s,a),(a,b),(b,t)\}) = N$ with length less or equal to L, which is exactly the definition of the L-cut. The minimum cardinality of the $L-cut$ is given by the minimum cardinality of E_c.

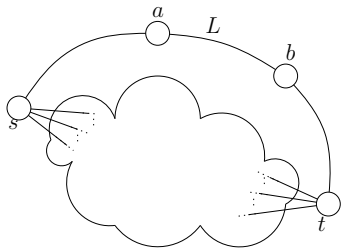

Fig. 1. A Minimum Length Bounded Cut is obtained by the solution of a Sliding Shortest Path Cut Problem.

Further the authors of [1] give the following theorem:

Theorem 2 (Approximability of Minimum L-Cut). *For any $\varepsilon > 0$ and $L \in \{4, \ldots, \lfloor n^{1-\varepsilon} \rfloor\}$, it is NP-hard to approximate the minimum length-bounded edge-cut in (un-)directed simple graphs within a factor of* 1.1377.

This transfers to:

Corollary 1 (Approximability of Sliding Shortest Path Cut). *The sliding shortest path cut problem is NP-hard to approximate in (un-)directed simple graphs within a factor of 1.1377, if all paths π_{spqt} have length $L \in \{4, \ldots, \lfloor n^{1-\varepsilon} \rfloor\}$ for any $\varepsilon > 0$.*

Algorithm 1 Heuristic Scheme

Input: NETWORK $N = (V, E),\ s, t \in V,\ (p, q) \in E$
Ouput: EDGES E_c
1. CHOOSE DESIRED SHORTEST S-T-PATH π_{spqt}
2. SET $L = |\pi_{spqt}|$
3. INCREASE CAPACITIES ON EDGES OF π_{spqt} APPROPRIATELY
4. REMOVE (p, q) FROM NETWORK
5. COMPUTE APPROXIMATION TO L-CUT C_L ON NETWORK
6. **return** $E_c = C_L$

4 Heuristics

A general scheme to solve the sliding shortest path cut problem heuristically would be the following:

To obtain an appropriate path π_{spqt}, the 2-disjoint paths problem can be solved as proposed in [3] for the pairs s, p and q, t and/or s, q and p, t, depending on whether the directed or undirected case is considered.

The capacity increase on the edges of π_{spqt}, i.e. of the order of m, ensures that none of its edges will be cut.

Choosing an appropriate shortest path from s to t via p, q does not mean the absolute shortest possible path $\pi_{\tilde{spqt}}$. Intuitively $\pi_{\tilde{spqt}}$ would lead to a minimum Δ Cost Increase. On the other hand, this -or any other- assumptive choice can increase $|E_c|$ artificially. Thus the approximative factor of a resulting L-cut solution is not likely to be preserved.

5 Conclusion

In this note we answered the open question concerning the complexity of the sliding shortest path cut problem which is a variant or a special case (depending on the original intention) of the sliding shortest path problem as R. Bhandari posed it in his talk at the Cologne Twente Workshop in Paris, 2009.

We hope the established connection to the length bounded cut problem helps in designing practical solutions to the original problem or other variants.

References

1. G. Baier, T. Erlebach, A. Hall, E. Khler, H. Schilling, M. Skutella "Length-Bounded Cuts and Flows" Proceedings of ICALP 2006, Part I, Lecture Notes in Computer Science 4051, pp. 679-690, 2006.
2. R. Bhandari "The Sliding Shortest Path Algorithms." Talk at 8th Cologne Twente Workshop on Graphs and Combinatorial Optimization, 2009.
3. T. Tholey "Solving the 2-Disjoint Paths Problem in Nearly Linear Time" Proceedings of STACS 2004, Lecture Notes in Computer Science 2996, pp. 350-361, 2004.