Shortest Anisotropic Paths with Few Bends is NP-complete

Mustaq Ahmed* Anna Lubiw*

In the *shortest anisotropic path (SAP) problem* [7], the goal is to minimize the weighted length of a path on a triangulated terrain, where the weight of a path segment ab depends both on the face containing ab and the direction of ab. The problem is a generalization of the weighted region problem. Several papers [3, 5, 8, 9] give approximation algorithms for SAPs. It is known (see, e.g., Lanthier et al. [5]) that a SAP can have many bends in the interior of a face, and there can be infinitely many SAPs even between two points in a common face. In practice, however, not all of these SAPs are equally good. For example, for robot motion planning, a SAP with fewer bends is preferable because a robot needs extra time and energy to change its direction at a bend. It is therefore natural to look for a SAP with a limited number of bends. We show here that the problem is hard:

Theorem. *Deciding if there exists a SAP of length at most L that has at most k bends is NP-complete.*

Our proof uses a reduction from 3-SAT following the idea of Canny and Reif's hardness result [2] for shortest paths among obstacles in 3D, though our gadgets are different as our paths lie on a 2D surface.

Our problem is a *bicriteria* shortest path problem. See Mitchell [6, Sec. 4] for a survey and Daescu et al. [4] for more recent approximation algorithms.

Many shortest path problems including anisotropic and bicriteria problems share the interesting feature that although approximation algorithms are the best results known, hardness proofs seem elusive. For example, the complexity of finding a shortest k-link path in a polygon is still open. Arkin et al. [1] give some hardness results for bicriteria shortest path problem, in particular proving that in a polygon with holes, minimizing length and total turn is NP-hard.

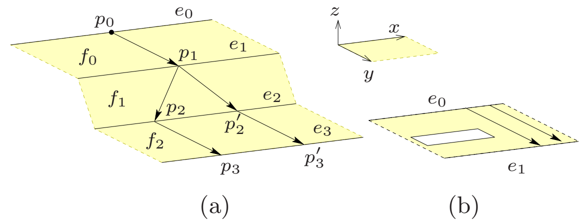
Construction

We use a simple direction-dependent weight function, defined as follows. Given an angle $\psi \in [0, \frac{\pi}{2})$, a line segment is *steep* if it makes an angle less than ψ with a vertical line. For any two points a and b in a common face, the weight of segment ab is $|ab|$ if ab is not a steep segment, and is infinity otherwise.

A direction in a face is called a *critical direction* if it makes an angle ψ with a vertical line. A bend on path P is a point where P changes its direction. To be precise, segments ab and bc bends at b if ab and bc are not collinear (our proof works even when a bend is defined in a planar unfolding of the faces).

We use two elementary gadgets in our construction. The first one is a *Splitter* which "splits" a SAP into two SAPs with exactly two bends each. A Splitter consists of three faces bounded by edges parallel to the x-axis, as shown in Fig. 1(a). Faces f_0 and f_2 are horizontal, and the slope of face f_1 is such that the angle between the two critical directions in this face is $\frac{\pi}{2}$. Clearly, any SAP from a point $p_0 \in e_0$ to edge e_3 leaves e_0 along the direction of the y-axis and reaches e_3 in the same direction. Moreover, the SAP reaches e_3 at a point in p_3p_3'. Now the SAP from p_0 to any point in the interior of p_3p_3' has at least three bends, but the SAP from p_0 to either p_3 or p_3' has exactly two bends. Thus each SAP from a point in e_0 to edge e_3 splits into two SAPs with two bends. Using the same gadget, two SAPs from edge e_0 to edge e_3 can be merged together to reach a single point in e_3. We can have any desired gap between the two SAPs in f_2 since the gap is twice the distance between e_1 and e_2.

The other elementary gadget is a *Blocker*, which is simply a rectangular hole in a horizontal face, as shown in Fig. 1(b) (we can also use rectangular pyramids instead). Any SAP from e_0 to e_1 that encounters the Blocker must travel around it, which



(a) (b)

Figure 1: (a) A Splitter. (b) A Blocker.

*David R. Cheriton School of Computer Science, University of Waterloo, Canada, {m6ahmed,alubiw}@uwaterloo.ca

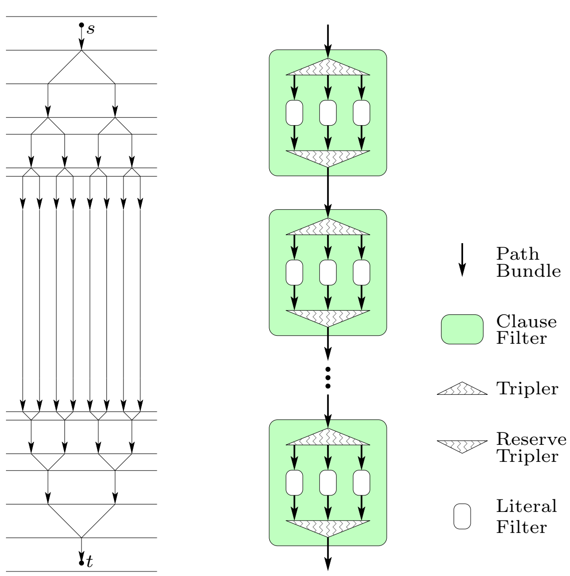
costs more in terms of length and bends.

Given a 3-SAT instance with n variables and m clauses, we first construct as follows a polynomial sized terrain that has exactly 2^n SAPs with a limited number of bends. We pick two points s and t on the y-axis, and then add n Splitters near s and n Splitters near t as shown in Fig. 2(a), so that we have from s to t exactly 2^n SAPs with $4n$ bends each (note that for the ease of discussion, we unfold our Splitters onto the xy-plane). We call this sequence of SAPs the *path bundle*.

The paths in the path bundle are then labeled with the integers from 0 to $2^n - 1$, from right to left in Fig. 2(a). Each path now represents one possible truth assignment for the n variables, with the ith bit of the label giving the "value" of the ith variable.

We then add to the middle face of the terrain in Fig. 2(a) a sequence of *Clause Filters*, one for each of the m clauses—a SAP can pass through such a filter if the truth assignment corresponding to the SAP satisfies the clause corresponding to the filter. A Clause Filter consists of the following gadgets: (i) A *Tripler* that splits the path bundle into three bundles, each containing the labeled paths in their original ordering. The Tripler consists of two Splitters and a Blocker. (ii) A *Reverse Tripler* that merges the three bundles from a Tripler into one path bundle. It is similar to a Tripler. (iii) Three *Literal Filters*, one for each literal in the corresponding clause.

A Literal Filter in a Clause Filter blocks the SAPs



(a) (b)

Figure 2: (a) The path bundle in the initial terrain. (b) Clause Filters along the path bundle.

whose truth assignments do not conform to the corresponding literal. A Literal Filter consists of a Blocker and a sequence of n *Shufflers*. A Shuffler makes a perfect shuffle of the paths in the path bundle (Fig. 3). The position of the Blocker in a Literal Filter depends on the corresponding literal, in the same manner as in Canny and Reif [2].

The paths in the path bundle that "survive" all the blockers (i) have exactly $4mn + 8m + 4n$ bends each, and (ii) are of equal length. Although each Shuffler halves the gap between consecutive paths in the path bundle, each coordinate in the entire construction can still be specified using an $O(mn)$-bit integer. The proof of the theorem follows.

(We thank Anil Maheshwari for useful discussion.)

References

[1] E. Arkin, J. Mitchell, and C. Piatko. Bicriteria shortest path problems in the plane. In *Proc. of 3rd CCCG*, pp. 153–156, 1991.

[2] J. Canny and J. Reif. New lower bound techniques for robot motion planning problems. In *Proc. of 28th FOCS*, pp. 49–60, 1987.

[3] S.-W. Cheng, H.-S. Na, A. Vigneron, and Y. Wang. Approximate shortest paths in anisotropic regions. In *Proc. of 18th SODA*, pp. 766–774, 2007.

[4] O. Daescu, J. Mitchell, S. Ntafos, J. Palmer, and C.-K. Yap. k-link shortest paths in weighted subdivisions. In *Proc. of 9th WADS*, pp. 325–337, 2005.

[5] M. Lanthier, A. Maheshwari, and J.-R. Sack. Shortest anisotropic paths on terrains. In *Proc. of 26th ICALP*, pp. 524–533, 1999.

[6] J. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, eds, *Handbook of Comput. Geom.*, pp. 633–701. 2000.

[7] N. Rowe and R. Ross. Optimal grid-free path planning across arbitrarily-contoured terrain with anisotropic friction and gravity effects. *IEEE Trans. Robot. Autom.*, 6(5):540–553, 1990.

[8] Z. Sun and T.-M. Bu. On discretization methods for approximating optimal paths in regions with direction-dependent costs. *Inform. Process. Lett.*, 97(4):146–152, 2006.

[9] Z. Sun and J. Reif. On finding energy-minimizing paths on terrains. *IEEE Trans. on Robotics*, 21(1):102–114, 2005.

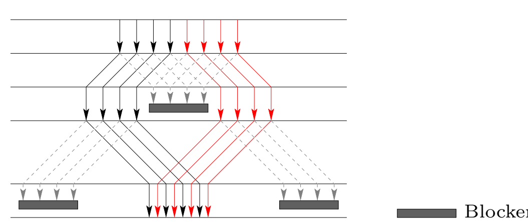


Figure 3: A Shuffler.