

Finding Euclidean Shortest Path

R. Inkulu

Assistant Professor

Department of Computer Science

IIT Guwahati

rinkulu@iitg.ac.in

<http://www.iitg.ac.in/rinkulu/>

Outline

1 Problem Description

2 Approach 1: Reduce to find a SP in graphs

Characterizations

Compute Tangents that lie in F between Convex Holes

Apply Graph Algorithm to find SP

3 Approach 2: Computing directly in geometric domain

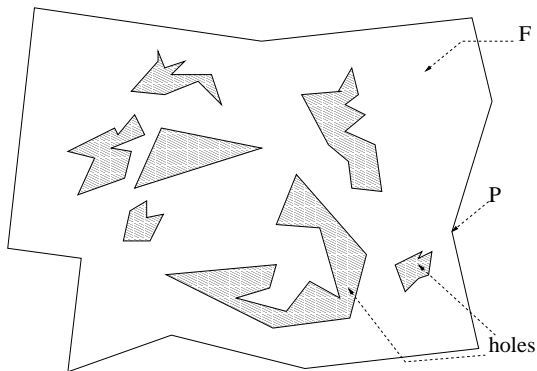
Dijkstra's Algorithm for Graphs

Continuous Dijkstra Method

4 Conclusions

Polygonal domain

- A simple polygon P containing disjoint simple polygonal holes (a.k.a. obstacles) in R^2 is termed as the **polygonal domain** D .
- Polygon P sans interior of the holes is termed as the **free space** F .

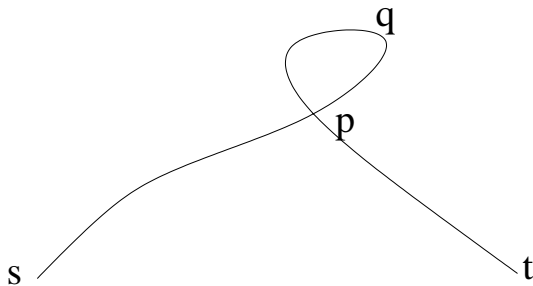


n : number of vertices

h : number of holes

Shortest paths are simple paths

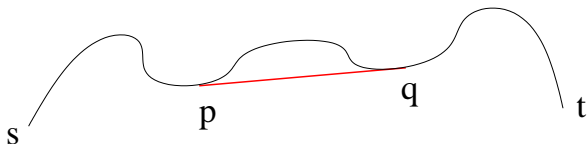
SP_{st} is a simple path.



Proof by contradiction

Optimal Substructure: SPs are locally shortest

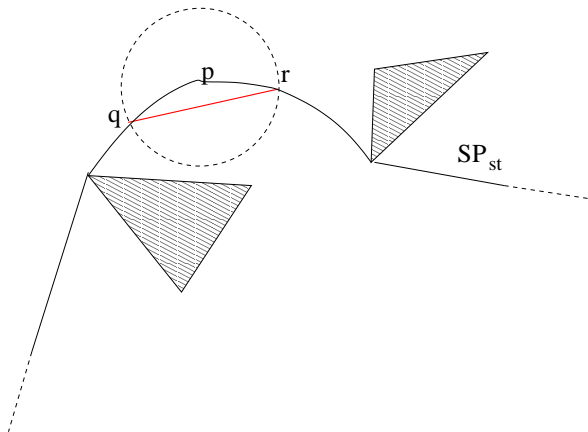
SP_{st} contains subpath L from p to $q \Rightarrow L$ must be SP_{pq}



Proof by contradiction

SP is polygonal

Every shortest path is a polygonal path.

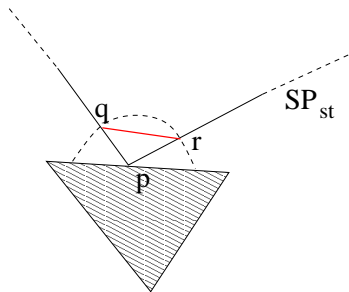
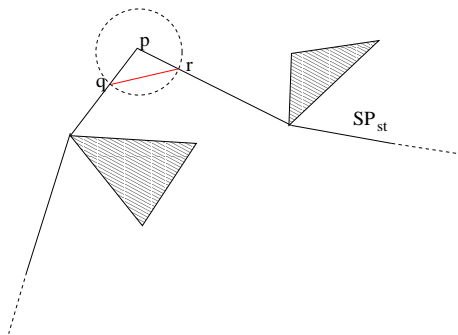


Proof by contradiction

Every internal vertex of SP is a vertex of D

No internal vertex of a SP can lie either:

- in the free space, or
- interior to an edge.



Proof by contradiction.

Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a SP in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find SP
- 3 Approach 2: Computing directly in geometric domain
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

High Level Description

- 1 Compute a weighted graph G from D so that an edge $e \in SP_{st}$ in D then $e \in G$.
- 2 Compute SP_{st} in G .

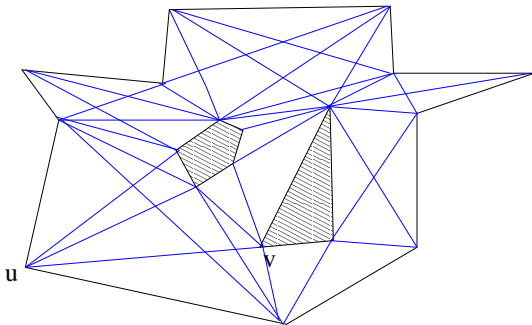
Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a *SP* in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find *SP*
- 3 Approach 2: Computing directly in geometric domain
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

Visibility Graph of D

The weighted undirected graph $VG_D(V, E')$ is defined over D such that:

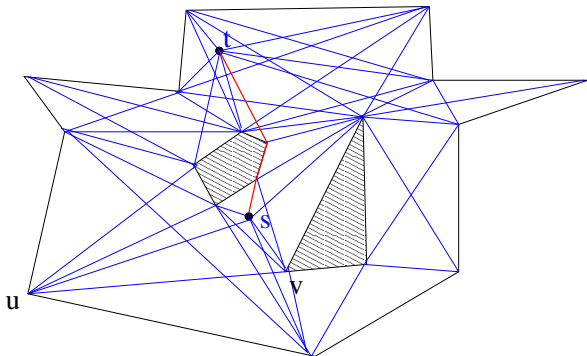
- V is the set of vertices in D ,
- an edge $e(u, v) \in E'$ whenever u and v are visible to each other in D , and
- for every edge $e(u, v)$, the weight of e is the Euclidean distance along the line segment uv in D .



Geometric Shortest Paths and Visibility Graphs

Considering s and t as degenerate holes in D , $e \in SP_{st}$ in $D \Leftrightarrow e \in SP_{st}$ in VG_D .

- SP_{st} is polygonal with the internal vertices chosen from D , and
- Every edge $e(u, v)$ in SP_{st} belongs to F .

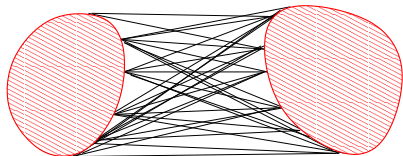


Time complexity is dominated by VG_D computation and $|E'|$.

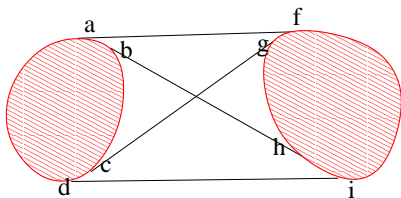
Tangent Visibility Graph

The **tangent visibility graph** $TVG_D(V, E)$ for D is defined whenever each hole in D is convex. It is same as $VG_D(V, E')$ except that:

an edge $e(u, v) \in E$ iff uv is either an edge of a hole in D or a *tangent between two convex holes*.



$|E|$ is $\Omega(n^2)$.

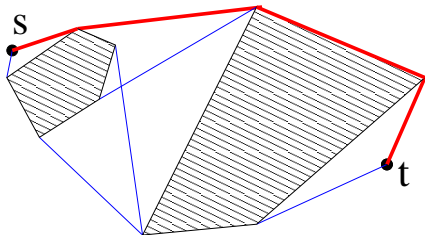


$|E|$ is $\Omega(h^2)$.

Computing tangents between two convex hulls CH' and CH'' takes $O(\lg |CH'| + \lg |CH''|)$ (from [Edelsbrunner, 1985]).

Geometric Shortest Paths and Tangent Visibility Graphs

Considering s and t as degenerate holes in D , $e \in SP_{st}$ in $D \Leftrightarrow e \in SP_{st}$ in TVG_D .



(Yet Another) High Level Description

Suppose that all holes are convex and the boundary of outer polygon is convex.

- 1 Compute the weighted tangent visibility graph TVG corresponding to D .
- 2 Find a shortest path from s to t in TVG .

Approach is due to [Rohnert, 1986].

Outline

1 Problem Description

2 Approach 1: Reduce to find a SP in graphs

Characterizations

Compute Tangents that lie in F between Convex Holes

Apply Graph Algorithm to find SP

3 Approach 2: Computing directly in geometric domain

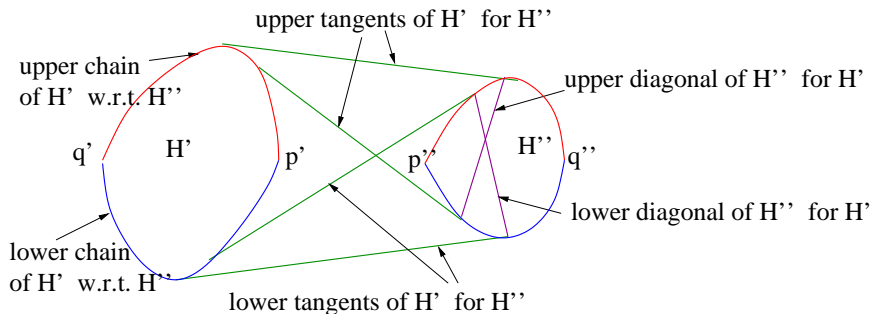
Dijkstra's Algorithm for Graphs

Continuous Dijkstra Method

4 Conclusions

Computing Tangents between every two Convex Holes

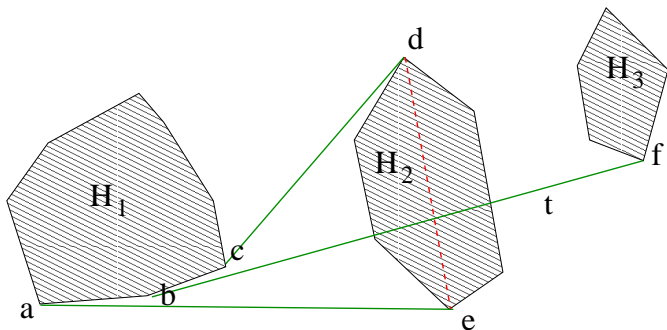
- Divide each convex chain into upper and lower convex chains.
- Compute all the four possible tangents between every two convex holes.



Time complexity: $O(h^2 \lg(n))$
from $\sum_{i,j} O(\lg |H'| + \lg |H''|)$

One lower tangent vs other lower tangents

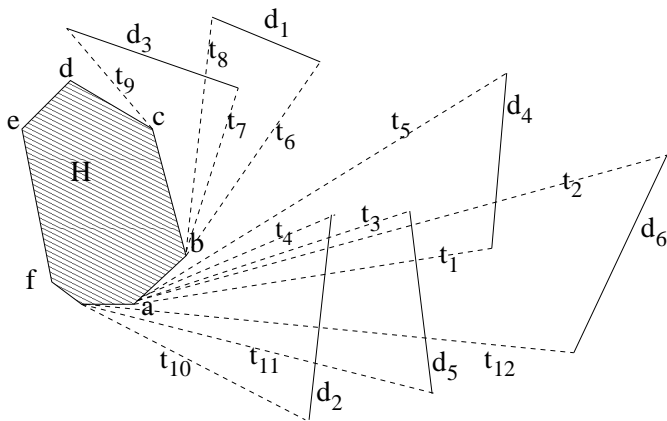
The lower tangent t of H_1 for H_3 intersects a hole H_2 iff t intersects the lower diagonal of H_2 for H_1 .



t intersects H_2 iff t intersects de .

Computing lower tangents that lie in F

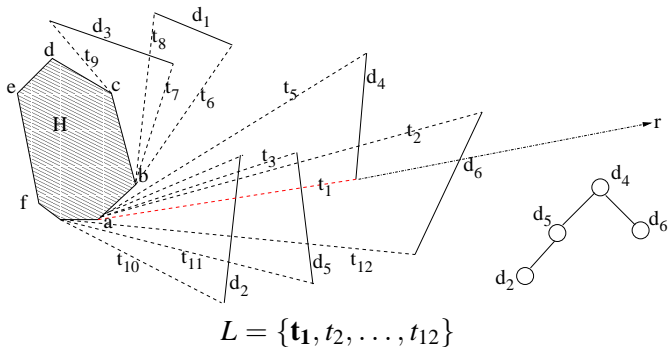
Traverse the boundary of H in counterclockwise order starting from any vertex a , and at each vertex v of H , add the lower tangents of H incident at v to the ordered list L according to their counterclockwise angle at v with the clockwise edge of H at vertex v .



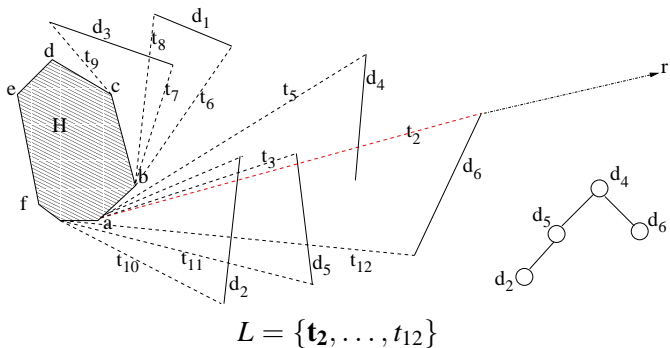
$$L = \{t_1, t_2, \dots, t_{12}\}$$

Computing lower tangents that lie in F (cont)

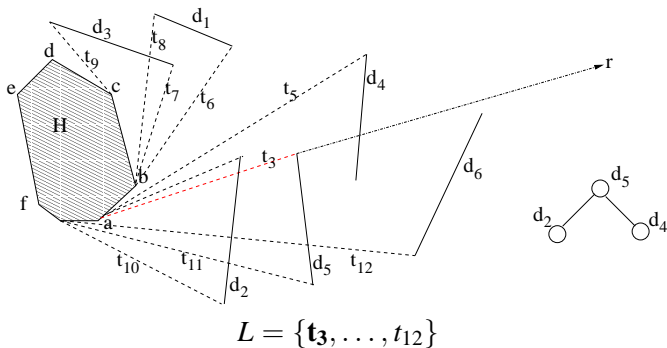
- Construct a BBST T to reflect the order in which the diagonals intersect the ray r .
- While exploring each such ray r , remove t_i from L if d corresponding to t_i is not the leftmost leaf of T .



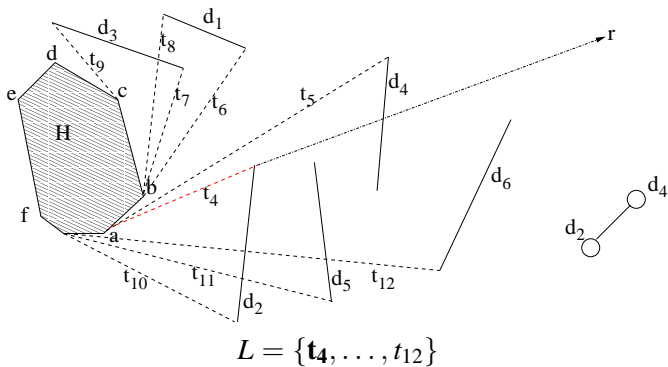
Computing lower tangents that lie in F (cont)



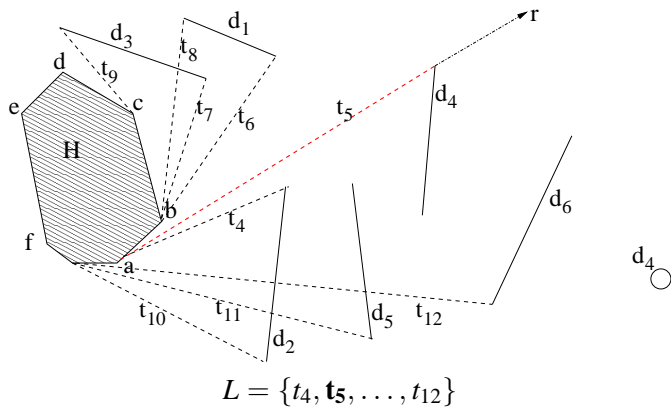
Computing lower tangents that lie in F (cont)



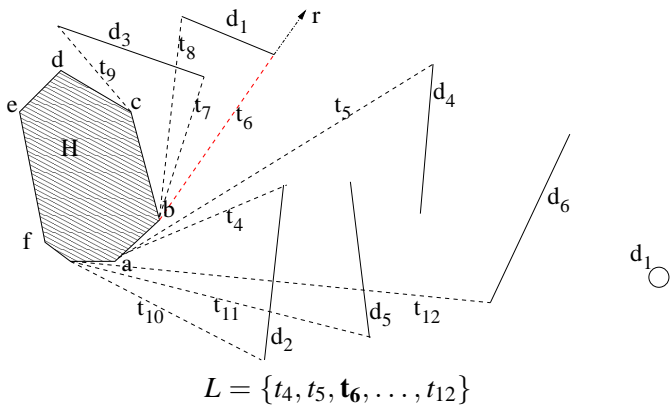
Computing lower tangents that lie in F (cont)



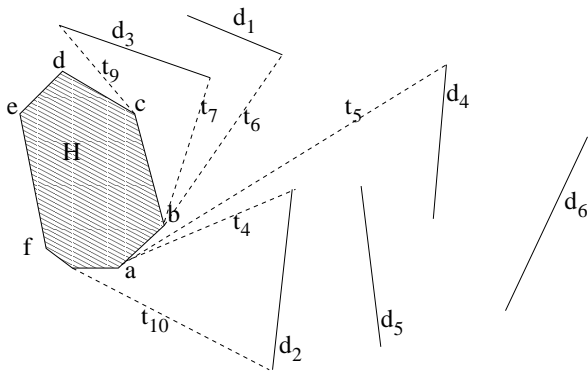
Computing lower tangents that lie in F (cont)



Computing lower tangents that lie in F (cont)



Computing lower tangents that lie in F (cont)



Output: $L = \{t_4, t_5, t_6, t_7, t_9, t_{10}\}$

- Time to process one hole: $O(|V_H| + h \lg h)$
- Time to process all holes: $O(n + h^2 \lg h)$

Computing upper tangents that lie in F (cont)

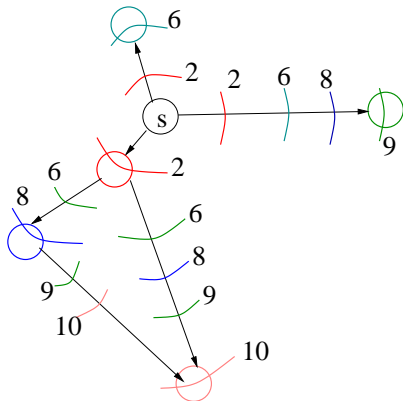
Applying similar procedure as in identifying lower tangents that lie in F :

- Time to process one hole: $O(|V_H| + h \lg h)$
- Time to process all holes: $O(n + h^2 \lg h)$

Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a *SP* in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find *SP*
- 3 Approach 2: Computing directly in geometric domain
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

Computing SP in $TVG_D(V, E)$ using Dijkstra's Algorithm



- $|V|$ is n
- $|E|$ is $O(h^2 + n)$
- Applying Dijkstra's Algorithm to compute SP takes $O(h^2 + n + n \lg n)$.

Time complexity of the suggested algorithm

- Computing all possible tangents between every two holes: $O(h^2 \lg n)$
- Computing all tangents that lie in F : $O(n + h^2 \lg h)$
- Computing SP over tangent visibility graph: $O(h^2 + n \lg n)$

Total time: $O(n + (n + h^2) \lg n)$.

Assuming $h^2 > n$, it is $O(n + h^2 \lg n)$.

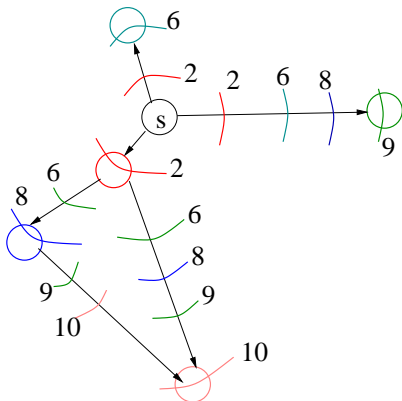
Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a SP in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find SP
- 3 Approach 2: Computing directly in geometric domain**
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a SP in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find SP
- 3 Approach 2: Computing directly in geometric domain**
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

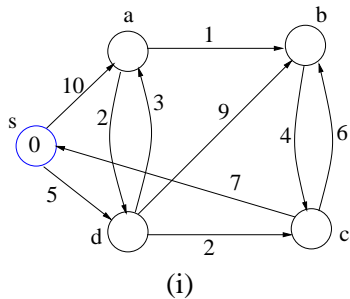
High-level Description



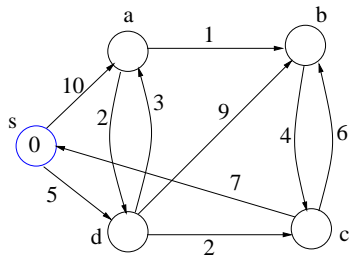
- Like in BFS, progress a shortest-path wavefront W from s in the graph.

Greedy approach.

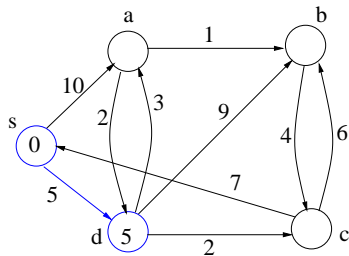
Dijkstra's Algorithm in Execution



Dijkstra's Algorithm in Execution

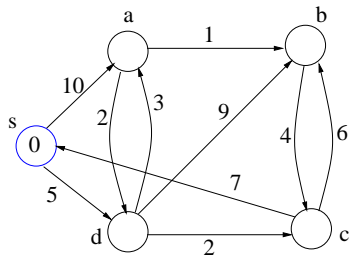


(i)

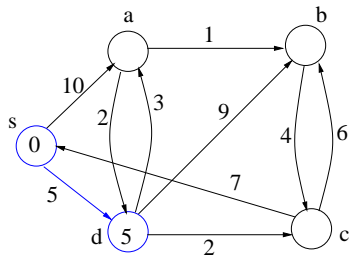


(ii)

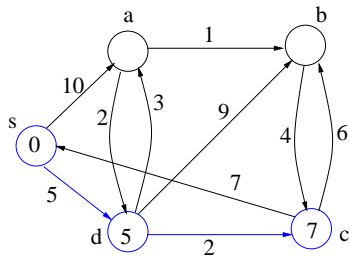
Dijkstra's Algorithm in Execution



(i)

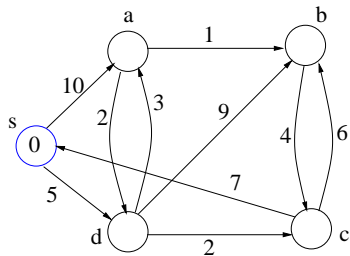


(ii)

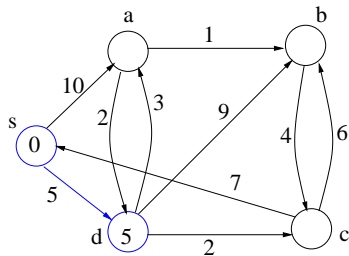


(iii)

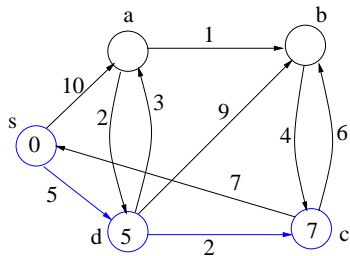
Dijkstra's Algorithm in Execution



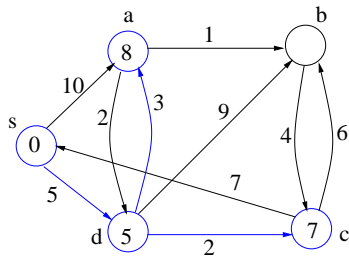
(i)



(ii)

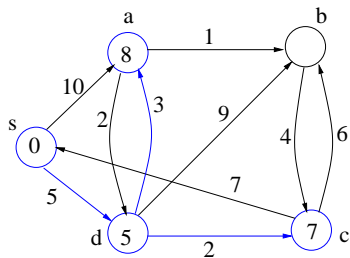


(iii)



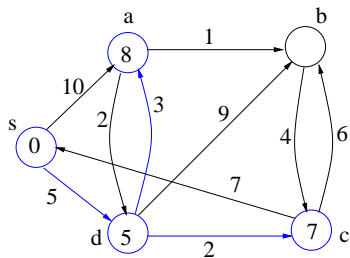
(iv)

Dijkstra's Algorithm in Execution (cont)

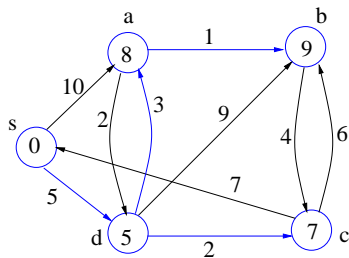


(iv)

Dijkstra's Algorithm in Execution (cont)



(iv)

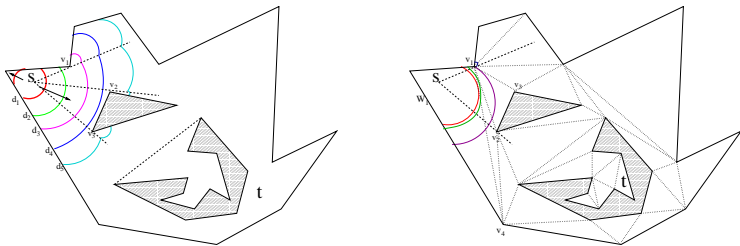


(v)

Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a SP in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find SP
- 3 Approach 2: Computing directly in geometric domain
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

Wavefront progression over geometric environment



- Wavefront, initiated at s , is expanded until it strikes t .
- Edges of a triangulation guide the wavefront.
- Initially wavefront comprises of single arc; with time, more arcs are added.

Based on this approach, [Herishberger, Suri, 1999] and gave $O(n \lg n)$ time and $O(n \lg n)$ space algorithm.

Recent improvement that uses this approach: [Inkulu, Sanjiv Kapoor, S. N. Maheshwari, 2010]

- Triangulation is coarsened to reduce the event points in guiding the wavefront.
- Exploits coherence both in the wavefront and in the polygonal domain.
- A solution with $O(T + h(\lg h)(\lg n))$ time and $O(n)$ space.

Outline

- 1 Problem Description
- 2 Approach 1: Reduce to find a SP in graphs
 - Characterizations
 - Compute Tangents that lie in F between Convex Holes
 - Apply Graph Algorithm to find SP
- 3 Approach 2: Computing directly in geometric domain
 - Dijkstra's Algorithm for Graphs
 - Continuous Dijkstra Method
- 4 Conclusions

Summary

- Building tangent visibility graph and running SP algorithm for graphs: $O(n + (n + h^2) \lg n)$ and $O(n)$ space.
- Running continuous Dijkstra's algorithm in geometric domain: $O(T + h(\lg h)(\lg n))$ and $O(n)$ space.
- Problem 21 of The Open Problems Project (TOPP) of Computational Geometry which intends for a solution with $O(n + h \lg h)$ time and $O(n)$ space.



Rohnert, H.

Shortest paths in the plane with convex polygonal obstacles

Information Processing Letters, 23:71-76, 1986



Bar-Yehuda, R. and Chazelle, B.

Triangulating disjoint Jordan chains

International Journal of Computational Geometry and Applications,

4:475-481, 1994



Lee, D. T. and Preparata, F. P.

Euclidean shortest paths in the presence of rectilinear barriers

Networks, 14:393-415, 1984



Dijkstra, E.

A note on two problems in connection with graphs






Numerische Mathematik, 1(1): 269-271, 1959



Edelsbrunner, H.

Finding extreme distance between convex polygons

Journal of Algorithms, 6: 213-224, 1985

-  Kapoor, S. and Maheshwari, S. N. and Mitchell, J. S. B.
An efficient algorithm for Euclidean shortest paths among polygonal obstacles in the plane
Discrete and Computational Geometry, 18:377-383, 1997
-  Hershberger, J. and Suri, S.
An optimal-time algorithm for Euclidean shortest paths in the plane
SIAM Journal on Computing, 28(299), 2215-2256.
-  Inkulu, R. and Kapoor, S. and Maheshwari, S. N.
A near optimal algorithm for finding Euclidean shortest path in polygonal domain
CoRR abs/1011.6481, 2010
-  Franco P. Preparata and Michael Ian Shamos
Computational Geometry: An Introduction
Springer-Verlag, New York, USA, 1st Edition, 1985
-  Mark de Berg, Otfried Cheong, Marc van Kreveld, Mark Overmars
Computational Geometry: Algorithms and Applications
Springer, New York, USA, 3rd Edition, 2010



Subir Kumar Ghosh

Visibility Algorithms in the Plane

Cambridge University Press, UK, 1st Edition, 2007.