MADALGO & CTIC Summer School 2011

Student Poster Session

Geometric t-spanner

Given a set of points, Construct a good network

Jungwoo Yang
KAIST, South Korea
yjwoo14@gmail.com

Network Measures

Stretch Factor (Dilation), Size, Weight, Degree, Diameter, Connectivity, Fault Tolerance, Genus, Numbers of Steiner Points, Load Factor ...

Known Algorithms

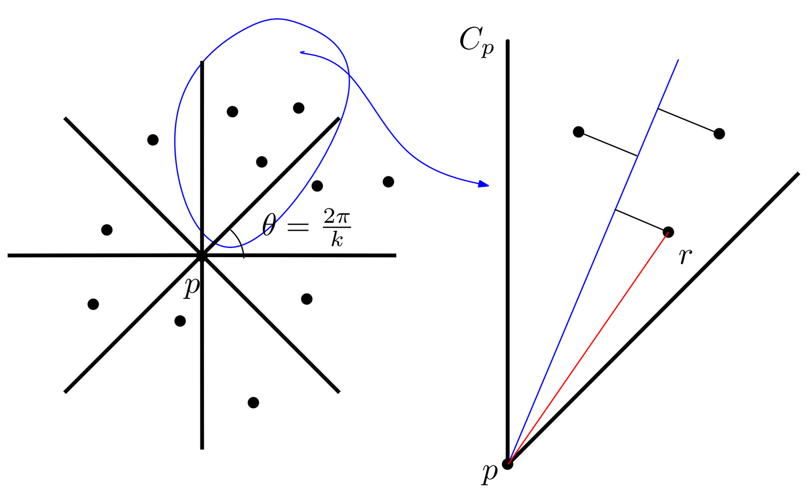
Greedy Algorithm O(n^3log$n$) → O(n^2log$n$)
WSPD-Spanner O($n$log$n$)
Skip-list O($n$log$n$)
Θ-Graph O($n$log$n$)
Variants of Θ-Graph

Greedy Algorithm

$G = (S, E = \emptyset)$
Sort all pair of points in nondecreasing order of their distance.
For each pair (p, q), (in sorted order)
  Compute shortest path $d_G(p,g)$ in G.
  if $|d_G(p,g)| > t|pq|$ then $E = E \cup (p,q)$.
return G

Θ−Graph

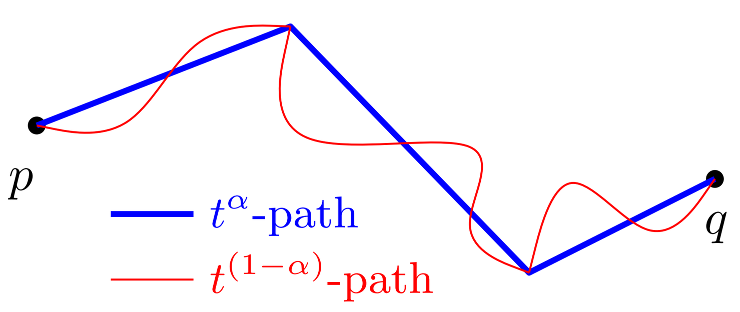
For each point p of S and each cone C_p the Θ-Graph contains one edge $\{p, r\}$, where r is a point in $C_p \cap S \setminus \{p\}$, whose orthogonal projection onto bisector of C_p is closest to p.



Hybrid Scheme

First, compute t^α-spanner $G_1(S, E_1)$.
($|E_1| = O(n)$)
Second, prune edges in E_1 such that every pair (p, q) in E_1 has $t^{(1-\alpha)}$ path.

→ Every pair (p, q) has $t^\alpha \times t^{(1-\alpha)} = t$ path.



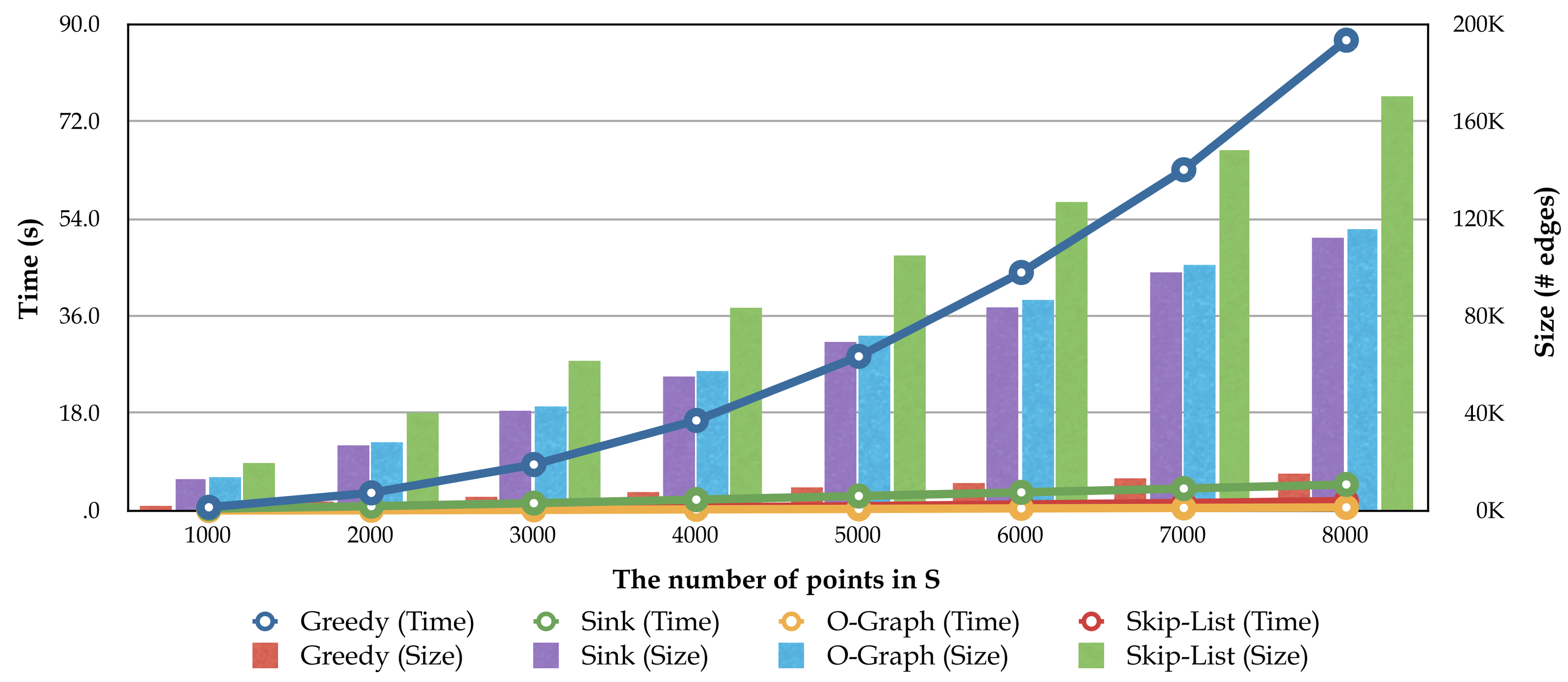
Theoretical bounds

Algorithms	Size	Weight	Degree	Time
Greedy	$O(n)$	$O(n)$	$O(1)$	$O(n^2 \log n)$
Θ−Graph	$O(n)$	$\theta(n)$	$\theta(n)$	$O(n \log n)$
O.Θ−Graph	$O(n)$	$O(n)$	$O(\log n)$	$O(n \log n)$
WSPD	$O(n)$	$O(\log n)$	$\theta(n)$	$O(n \log n)$
Sink	$\theta(n)$	$O(n)$	$O(1)$	$O(n \log n)$
Skip−list	$\theta(n)$	$\theta(n)$	$\theta(n)$	$O(n \log n)$
Hybrid1	$O(n)$	$O(n)$	$O(1)$	$O(n \log n)$

Definition (Spanner)

Let S be a set of n points in $\mathbb{R}^d$ and $t \geqq 1$ be a real number.
A ***t*-spanner** for S is an undirected graph G with vertex set S, such that for any two points p and q, there is a path in G between p and q, whose length is less than equal to $t|pq|$, where $|pq|$ is the Euclidean distance from p to q.

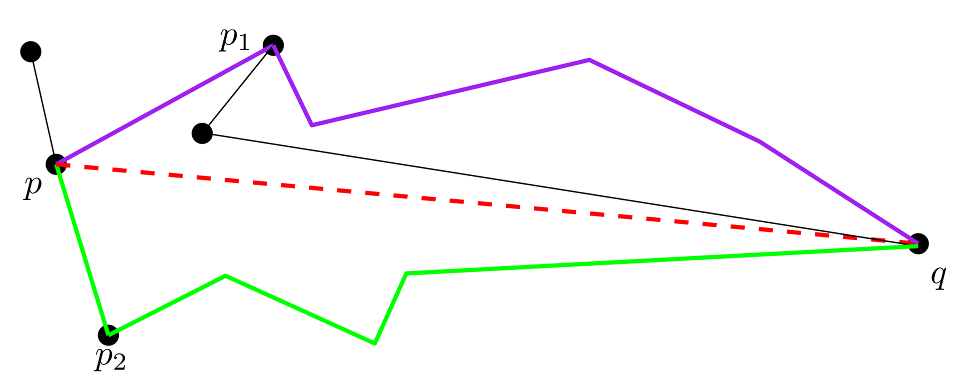
Problem



→ The greedy algorithm is too slow, the other algorithms generate bad networks.

Approach

Compute the **approximate shortest path** $d'_G(p,q)$ instead of computing Dijkstra alg.
Let (p,q) be a pair being considered.
Let $p_1, \dots, p_k$ and $q_1, \dots, q_l$ be neighbor points of p and q respectively.
$d'_G(p,q) = min(|pp_1| + d_G(p_1,q), \dots, |pp_k| + d_G(p_k,q), |pq_1| + d_G(p,q_1), \dots, |pq_l| + d_G(p,q_l))$.
If $d'_G(p,q) > t|pq|$ then $d_G(p,q) = |pq|$ else $d_G(p,q) = d'_G(p,q)$.



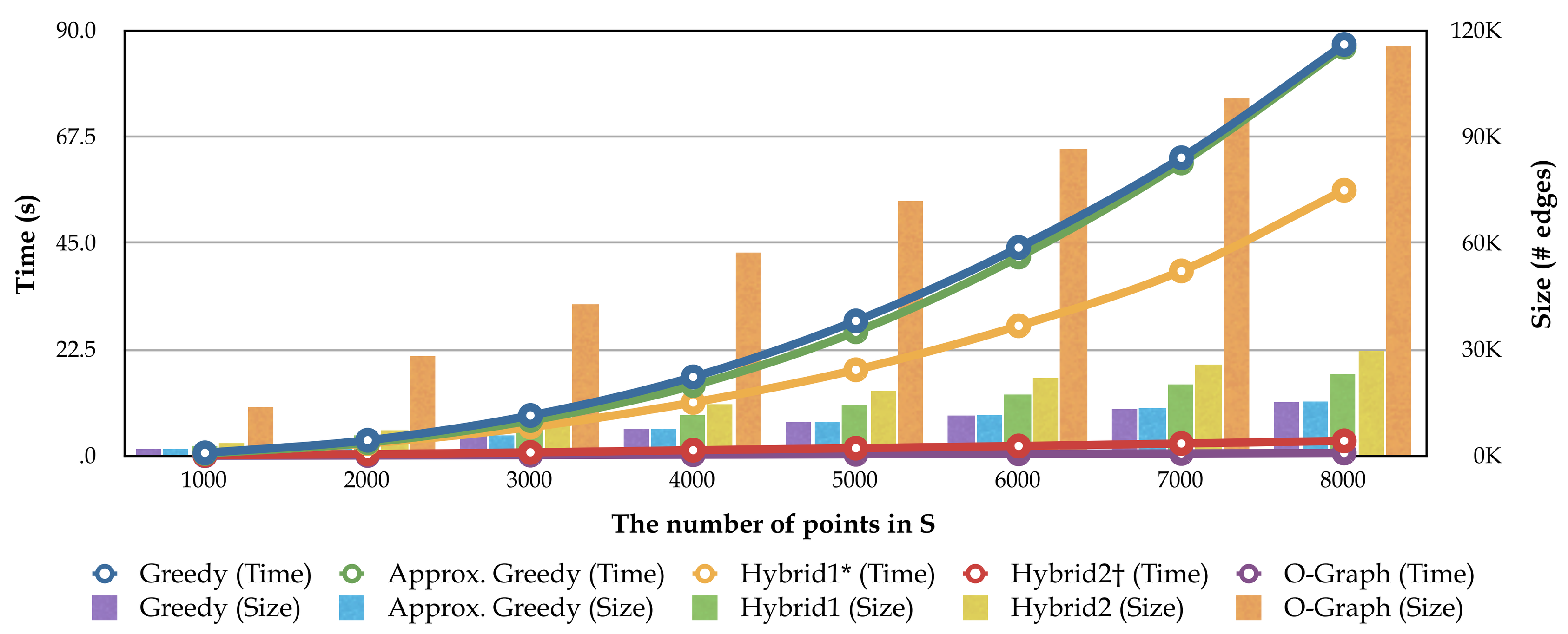
→ Still O(n^2log$n$) + O(n^2) $g(n)$ algorithm, where $g(n)$ is the degree complexity of G.

Approach

Using the hybrid scheme, improve time complexity.
First, generate t^α-spanner $G_1(S, E_1)$ using a variant of Θ-Graph.
Second, for each edge (p,q) in E_1 compute the approximate shortest path $d'_G(p,q)$.
  If $d'_G(p,q) > t|pq|$ then *add (p,q) to result graph $G(S,E)$.*

→ O($n$log$n$) + O(n) $g(n)$ algorithm, where $g(n)$ is the degree complexity of G.

Experiment



Questions

Does the hybrid2 algorithm really bound degree? (If so, it is definitely O($n$log$n$) alg.)
Can it be transformed into angle-constrained spanner? (If so, the degree is bounded by constant.)
How does the α factor affect the result?
Is there any other good preprocessing algorithm instead of Θ−Graph?

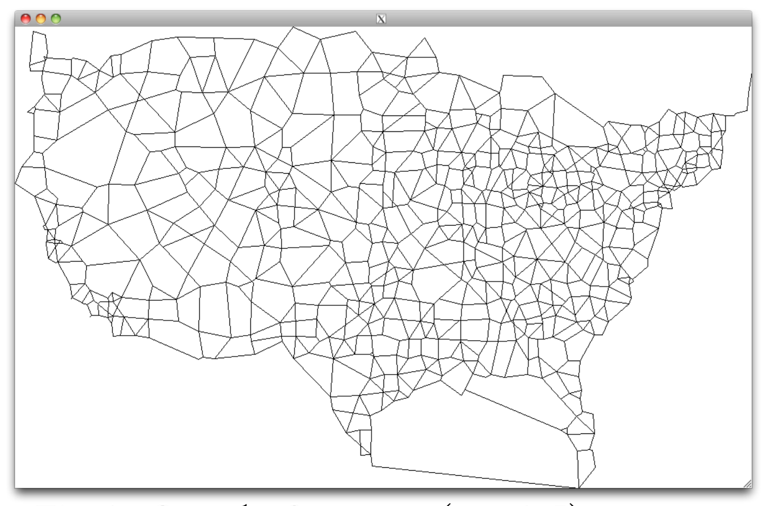


Fig 1. Greedy Spanner (t = 1.5)

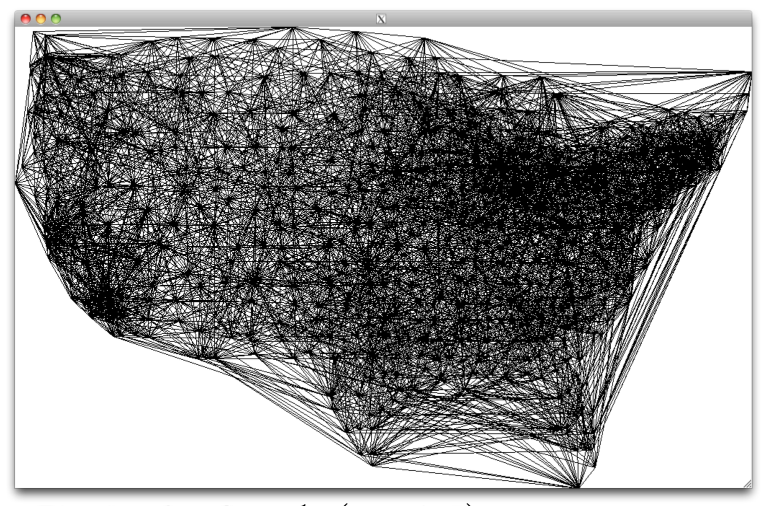


Fig 2. Θ−Graph (t = 1.5)

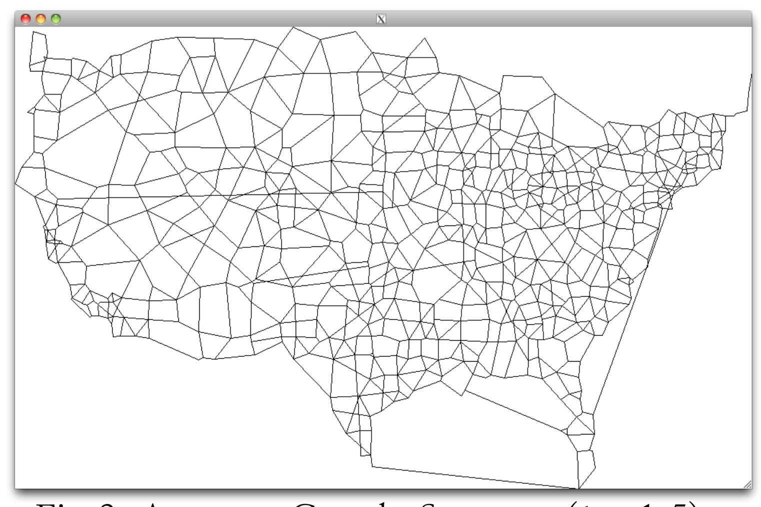


Fig 3. Approx. Greedy Spanner (t = 1.5)

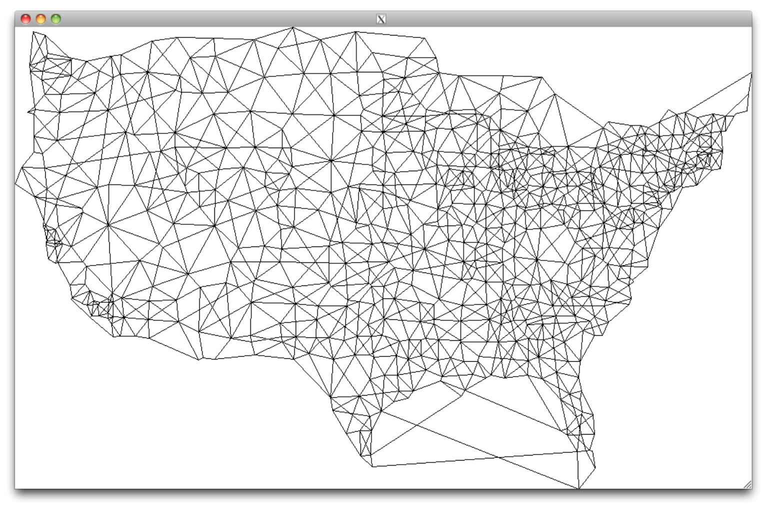


Fig 4. Hybrid1 Spanner (t = 1.5, α = 0.5)

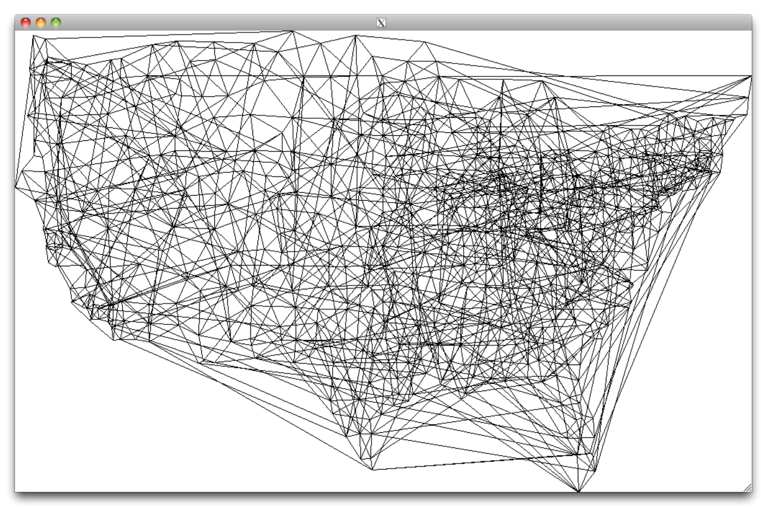


Fig 5. Hybrid2 Spanner (t = 1.5, α = 0.5)

* Hybrid1 uses ***Θ-Graph*** for t^α-spanner $G_1(S, E_1)$ in the first phase, then uses ***greedy algorithm*** with $t^{(1-\alpha)}$, E_1.
† Hybrid2 uses ***Θ-Graph*** for t^α-spanner $G_1(S, E_1)$ in the first phase, then uses ***approximate greedy algorithm*** with $t^{(1-\alpha)}$, E_1.