Light Spanners in Bounded Pathwidth Graphs

Michelangelo Grigni, Vincent Hung

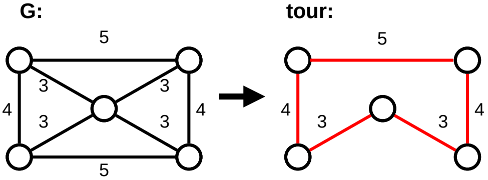
Department of Math & CS
Emory University

MFCS 2012

Outline

TSP in a Graph

Input: edge-weighted graph G, defining s.p. metric
Output: shortest cyclic tour of all vertices



Known:

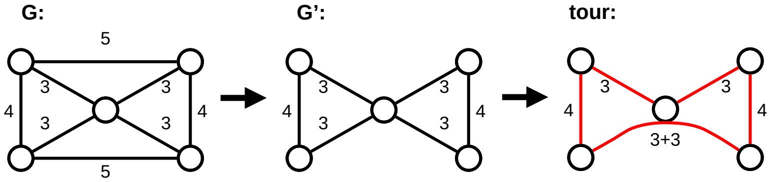
- NP-hard (even for planar)
- SNP-hard (in general)
- Easy 2-approx: MST $\leq$ TSP $\leq$ 2$\cdot$MST
- Approximation schemes for nice graph families:
  either EPTAS (planar, ...) or PTAS (forbidden minor)

Typical approach:

- graph decomposition (separate or contract)
- few portals (tour entry/exit points) per subgraph
- dynamic programming

Problem: we get error $\epsilon \cdot w(G)$, but we *want* $\epsilon{\cdot}$TSP.

Fix? replace G with lighter G', with a similar metric.



Light Spanners

Definition: If G is edge-weighted, a $(1+\epsilon)$-*spanner* is a subgraph G' with distances at most $(1+\epsilon)$ times larger.

Definition: Suppose $\mathcal{G}$ is a graph class, and for each $\epsilon > 0$ and weighted $G \in \mathcal{G}$, G has a $(1+\epsilon)$-spanner with weight at most $f(\epsilon)\cdot$MST. Then we say: $\mathcal{G}$ *has light spanners*.

Example: Planar graphs have light spanners.
(By greedy spanner algorithm, with $f(\epsilon) = 1 + 2/\epsilon$.)

Conjecture

$\mathcal{G}$ has a forbidden minor $\Rightarrow$ $\mathcal{G}$ has light spanners.

Remarks: Would imply TSP EPTAS on $\mathcal{G}$. Converse is true.

Towards the Conjecture

Suppose $\mathcal{G}$ has a forbidden minor.
By Robertson-Seymour, graphs in $\mathcal{G}$ can be built using:

- bounded genus graphs
- bounded # apices
- bounded # vortices (bounded pathwidth subgraphs)
- clique sums

Prior work: handles the first two.

Theorem (Today)

Bounded pathwidth graphs have light spanners.

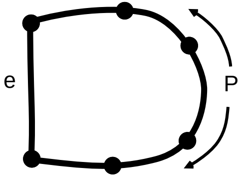
Open: clique sums, bounded treewidth, multiple vortices.

Outline

1. Background
   - Motivation: TSP and Spanner Conjecture
   - Spanners via Detour Charging Schemes
2. Our Work
   - Bounded Pathwidth: Simplify G and T
   - Putting it Together: the Hard Part

Detours

Given edge e and path P so $e + P$ is a simple cycle in G, we say (e, P) is a *detour*:



Greedy idea: if $w(P) \leq (1+\epsilon) \cdot w(e)$, we may omit e.

Greedy Algorithm: Given $G = (V, E)$, let $G' = (V, \emptyset)$.
For each edge $e \in E$, in $w(e)$ order:
if G' has no good enough detour for e, add e to G'.
Return G'.

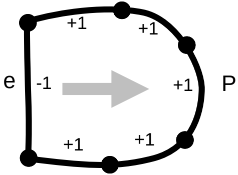
We use an LP to bound $w(G')/MST$.

Charging Moves

Dual idea: Each edge in G holds "charge", initially 0.

Each detour (e, P) defines a *charging move*:

- remove a unit of charge from e, and
- add a unit of charge to *each* $e' \in P$



A *charging scheme* is a nonnegative sum of such moves.
For each e, it defines in(e), out(e), and net(e)=in(e)-out(e).

Charging from G to T

Given (G, T), where tree T spans G, a *scheme of value v* is a charging scheme with:

$$\begin{aligned} \text{out}(e) &\geq 1 && \text{for } e \in G - T \\ \text{out}(e) &= 0 && \text{for } e \in T \\ \text{net}(e) &\leq 0 && \text{for } e \in G - T \\ \text{net}(e) &\leq v && \text{for } e \in T \end{aligned}$$

Intuition: the scheme moves all charges onto the tree.

Theorem (LP duality)

If G is a greedy $(1+\epsilon)$-spanner with a scheme of value v, then $w(G) \leq (1 + v/\epsilon) \cdot w(T)$.

This is usually applied with T as the MST. If schemes of value v exist for all $G \in \mathcal{G}$ and T spanning G, then $\mathcal{G}$ has light spanners.
Example: $v = 2$ works for planar graphs.

Outline

Reminder: Bounded Pathwidth

Represent each vertex by an interval, an edge is allowed when two intervals overlap. If at most $k + 1$ intervals overlap at once, then G has *pathwidth* $\leq k$.

Example: $K_{3,5}$

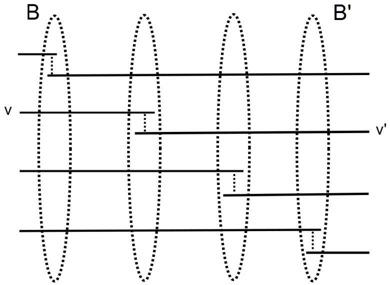
Equivalently, as a sequence of bags:

1,	1,	1,	1,	1,
2,	2,	2,	2,	2,
3,	3,	3,	3,	3,
4	5	6	7	8

Simplify: Make G Complete

The weighted G has pathwidth k. By reductions, we assume:

- G is *completed*: has every edge allowed by intervals.
  Trick: new edge length is s.p. distance, can undo later.
- each vertex is in $O(k)$ bags, with degree $O(k)$.
  Trick: split each v in two, reconnect with weight 0 edges.



Remark: first item requires us to find an *acyclic* charging scheme, to allow deletion of edges.

Simplify: Make T Monotone

Instead of the MST, we seek a light *monotone* spanning tree in G. A monotone tree can be rooted so all leaf-to-root paths go right to left. (Imagine edge connections at leftmost location.)

not monotone

monotone

Easy: find the lightest monotone tree in a completed G.

Outline

Two Theorems

Let T be a light monotone tree. We need a charging scheme to bound the weight of the greedy spanner in terms of $w(T)$, and we also must bound $w(T)$ in terms of the MST.

Theorem (1)

If T is monotone, there is an acyclic scheme of value $O(k)$.

Theorem (2)

Some monotone tree in G has weight $O(k^2) \cdot MST$.

Remarks:

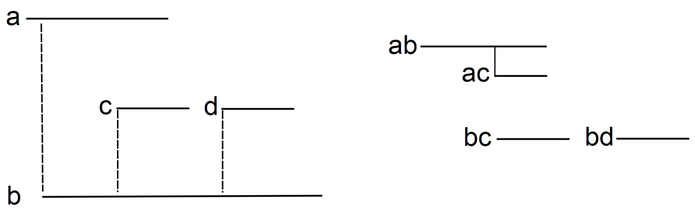
- we modify the greedy algorithm to force T into G'
- acyclic scheme applies to subgraph G' (by shortcutting)
- together these imply our main result, $f(\epsilon) = O(k^3/\epsilon)$
- in bounded *treewidth*, Theorem (2) fails.

Theorem (1): Sketch

We only use detours (e, P) where $e \notin T$ and P has at most one non-T edge. If P has length two, we call this move a *triangle*.

- Define $T^{(2)}$: vertices are G edges, edges are triangles.
- Show $T^{(2)}$ is a forest, each component rooted at a T edge.
- Euler tour each component of $T^{(2)}$, ending at its T edge.
- Shortcut out repeated non-tree edges for acyclic scheme.
- A T edge is charged at most once per triangle containing it.

Example T and $T^{(2)}$:

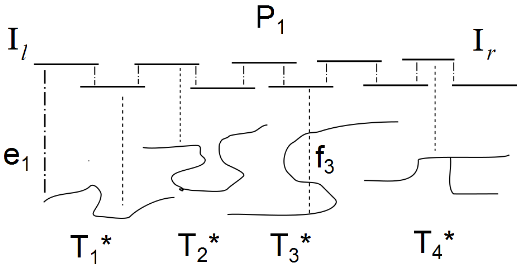


Theorem (2): Sketch

Construction:

- Let T^* be the MST.
- Find shortest path P_1 from leftmost to rightmost interval.
- Find components T_i^* of $T^* - V(P_1)$.
- Recursively convert each T_i^* to monotone T_i.
- Connect root of each T_i to P_1, result is monotone T.

Analysis: see paper.



Summary

Larger context:

- this work: view as progress towards the conjecture
- currently: vortex in planar graph
- open: bounded treewidth, path-like clique sums
  (these may be handled algorithmically)

Questions?