Distributed Graph Algorithms for Planar Networks

Bernhard Haeupler
CMU

joint work with Mohsen Ghaffari (MIT)

ADGA, Austin, October 12th 2014

Distributed Graph Algorithms for Planar Networks

Distributed: CONGEST($\log n$) model

Distributed: CONGEST($\log n$) model

Network Optimization Problems:

- shortest path (single/multiple source(s), (non-)negative weights, ...)
- network flows (min-cost, multi-commodity, ...)
- trees (MST, decomposition, steiner-tree, ...)
- TSP, min (st-)cut, facility location, ...

Distributed Graph Algorithms for **Planar Networks**

Distributed: CONGEST($\log n$) model

Network Optimization Problems:

- shortest path (single/multiple source(s), (non-)negative weights, ...)
- network flows (min-cost, multi-commodity, ...)
- trees (MST, decomposition, steiner-tree, ...)
- TSP, min (st-)cut, facility location, ...

Planar Networks:

- Many real-world networks/problems have planar like structure.
- Studying planar like networks led to:
  - a rich theory,
  - a powerful algorithmic toolbox, and
  - drastically improved algorithms.

Distributed Graph Algorithms for **Planar Networks**

Distributed: CONGEST($\log n$) model

Network Optimization Problems:

- shortest path (single/multiple source(s), (non-)negative weights, ...)
- network flows (min-cost, multi-commodity, ...)
- trees (MST, decomposition, steiner-tree, ...)
- TSP, min (st-)cut, facility location, ...

Planar Networks:

- Many real-world networks/problems have planar like structure.
- Studying planar like networks led to:
  - a rich theory,
  - a powerful algorithmic toolbox, and
  - drastically improved algorithms.

Goal:

Distributed toolbox/algorithms/theory for planar networks.

Algorithms in Planar vs. General Networks, e.g.:

Sequential Toolbox/Algorithms for Planar Networks

Algorithms in Planar vs. General Networks, e.g.:

- Much Faster Algorithms
  - SS-shortest-path: $O(n \log^2 n)$ vs. $O(mn)$ / $O(n)$ vs. $O(n \log n)$
  - max-flow: $O(n \log n)$ vs. $O(nm)$

Sequential Toolbox/Algorithms for Planar Networks

Algorithms in Planar vs. General Networks, e.g.:

- Much Faster Algorithms
  - SS-shortest-path: $O(n \log^2 n)$ vs. $O(mn)$ / $O(n)$ vs. $O(n \log n)$
  - max-flow: $O(n \log n)$ vs. $O(nm)$
- $1 + \epsilon$ Approximations vs.
  - TSP: $3/2 - \epsilon$
  - multi-way cut: $O(\log k)$
  - Independent Set: $\Omega(n^{1-\epsilon})$

Sequential Toolbox/Algorithms for Planar Networks

Algorithms in Planar vs. General Networks, e.g.:

- Much Faster Algorithms
  - SS-shortest-path: $O(n \log^2 n)$ vs. $O(mn)$ / $O(n)$ vs. $O(n \log n)$
  - max-flow: $O(n \log n)$ vs. $O(nm)$
- $1 + \epsilon$ Approximations vs.
  - TSP: $3/2 - \epsilon$
  - multi-way cut: $O(\log k)$
  - Independent Set: $\Omega(n^{1-\epsilon})$
- Fixed Parameter Tractability, EPTAS, ...

Sequential Toolbox/Algorithms for Planar Networks

Algorithms in Planar vs. General Networks, e.g.:

- Much Faster Algorithms
  - SS-shortest-path: $O(n \log^2 n)$ vs. $O(mn)$ / $O(n)$ vs. $O(n \log n)$
  - max-flow: $O(n \log n)$ vs. $O(nm)$
- $1 + \epsilon$ Approximations vs.
  - TSP: $3/2 - \epsilon$
  - multi-way cut: $O(\log k)$
  - Independent Set: $\Omega(n^{1-\epsilon})$
- Fixed Parameter Tractability, EPTAS, ...

Theory: topology, bounded genus graphs, graph minors, Roberson-Seymor Theorem, tree/path width, ...

Sequential Toolbox/Algorithms for Planar Networks

Algorithms in Planar vs. General Networks, e.g.:

- Much Faster Algorithms
  - SS-shortest-path: $O(n \log^2 n)$ vs. $O(mn)$ / $O(n)$ vs. $O(n \log n)$
  - max-flow: $O(n \log n)$ vs. $O(nm)$
- $1 + \epsilon$ Approximations vs.
  - TSP: $3/2 - \epsilon$
  - multi-way cut: $O(\log k)$
  - Independent Set: $\Omega(n^{1-\epsilon})$
- Fixed Parameter Tractability, EPTAS, ...

Theory: topology, bounded genus graphs, graph minors, Roberson-Seymor Theorem, tree/path width, ...

Toolbox:

- Planarity Testing / Embedding / Graph Drawing
- Decompositions (Separators, RS-decomp., tree width, ...)
- Bidimensionality, ...

Distributed Setting: Round Complexity in CONGEST

Trivial Bounds:

- $O(m)$ vs. $\Omega(D)$, Big Gap!

Distributed Setting: Round Complexity in CONGEST

Trivial Bounds:

- $O(m)$ vs. $\Omega(D)$, Big Gap!

General Graphs: Minimum Spanning Tree:

- $\tilde{O}(D + \sqrt{n})$ [KP'95]
- Strong $\tilde{\Omega}(\sqrt{n})$ Lower Bound [RP'99,E'04,DHK^{+}'11]
  - despite tiny diameter, e.g., $D = \log n$
  - even for any approximation
  - even for sparse/bounded degree graphs

Distributed Setting: Round Complexity in CONGEST

Trivial Bounds:

- $O(m)$ vs. $\Omega(D)$, Big Gap!

General Graphs: Minimum Spanning Tree:

- $\tilde{O}(D + \sqrt{n})$ [KP'95]
- Strong $\tilde{\Omega}(\sqrt{n})$ Lower Bound [RP'99,E'04,DHK$^+$'11]
  - despite tiny diameter, e.g., $D = \log n$
  - even for any approximation
  - even for sparse/bounded degree graphs

Question:

What natural graph classes avoid the $\tilde{\Omega}(\sqrt{n})$ bound?

Distributed Setting: Round Complexity in CONGEST

Trivial Bounds:

- $O(m)$ vs. $\Omega(D)$, Big Gap!

General Graphs: Minimum Spanning Tree:

- $\tilde{O}(D + \sqrt{n})$ [KP'95]
- Strong $\tilde{\Omega}(\sqrt{n})$ Lower Bound [RP'99,E'04,DHK$^+$'11]
  - despite tiny diameter, e.g., $D = \log n$
  - even for any approximation
  - even for sparse/bounded degree graphs

Question:

What natural graph classes avoid the $\tilde{\Omega}(\sqrt{n})$ bound?

Goal:

$\tilde{O}(D)$ network optimization algorithms for planar networks.

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

Standard planar toolbox does not extend:

- Triangulation

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

Standard planar toolbox does not extend:

- Triangulation
- Geometric Dual Graph

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

Standard planar toolbox does not extend:

- Triangulation
- Geometric Dual Graph
- DFS-based algorithms

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

Standard planar toolbox does not extend:

- Triangulation
- Geometric Dual Graph
- DFS-based algorithms
- Separator guarantees

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

Standard planar toolbox does not extend:

- Triangulation
- Geometric Dual Graph
- DFS-based algorithms
- Separator guarantees
- Data Structures (e.g., Dynamic Trees, PQ-trees, ...)
- ...

A Distributed Perspective on Planar Network (Algorithms)

Distributed Perspective:

- rich classical theory/toolbox to be inspired by
- BUT entirely new approaches and tools are needed
- provides new aspects, viewpoints, solutions, and problems

Standard planar toolbox does not extend:

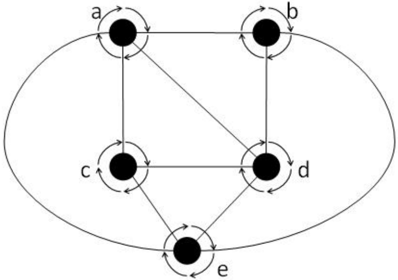
- Triangulation
- Geometric Dual Graph
- DFS-based algorithms
- Separator guarantees
- Data Structures (e.g., Dynamic Trees, PQ-trees, ...)
- ...

Goal:

Develop a new **distributed** algorithmic toolbox for planar networks.

New Tools I: A Distributed Planar Embedding Algorithm

An embedding is often necessary (not just knowing its existence).



Claim 1:

There is a $\tilde{O}(D)$ distributed planar embedding algorithm.

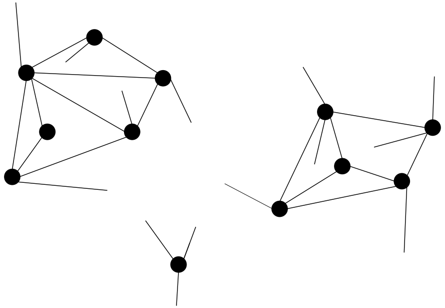
Efficient (Distributed) Planar Embedding Algorithm

General Idea [LEC'67,HT'08]

- Build embedding incrementally by adding vertices
- Track all possible partial embeddings

Efficient (Distributed) Planar Embedding Algorithm

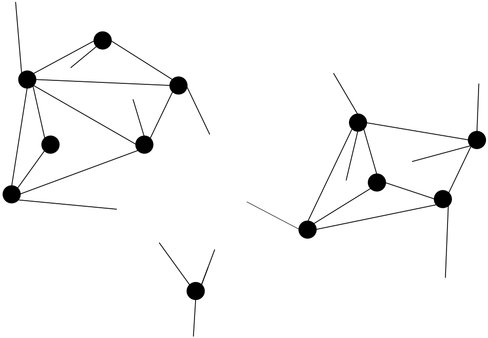
Partial Embedding:



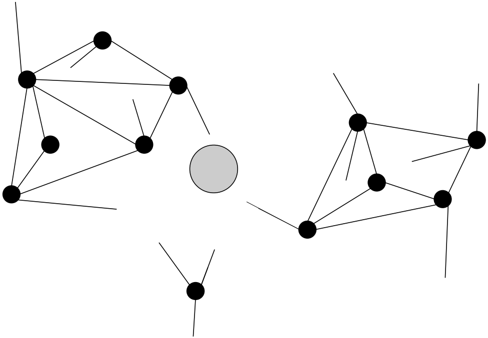
General Idea [LEC'67,HT'08]

- Build embedding incrementally by adding vertices
- Track all possible partial embeddings

Partial Embeddings

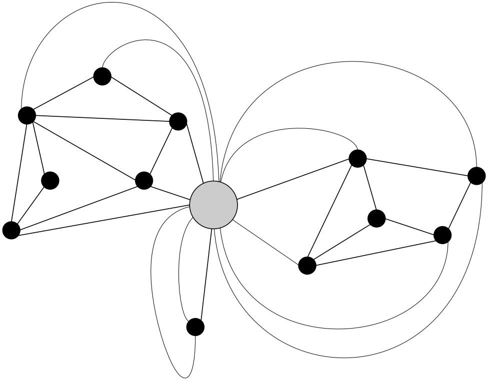


Partial Embeddings



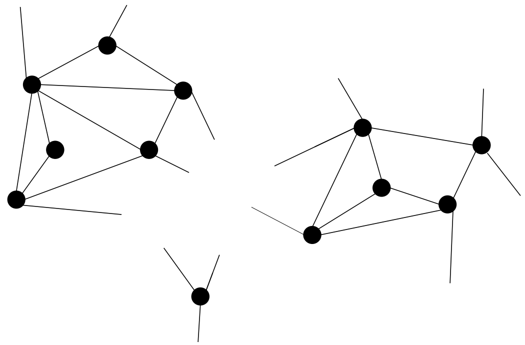
Choose the (parallel) embedding order such that the non-embedded vertices are connected.

Partial Embeddings



Choose the (parallel) embedding order such that the non-embedded vertices are connected.

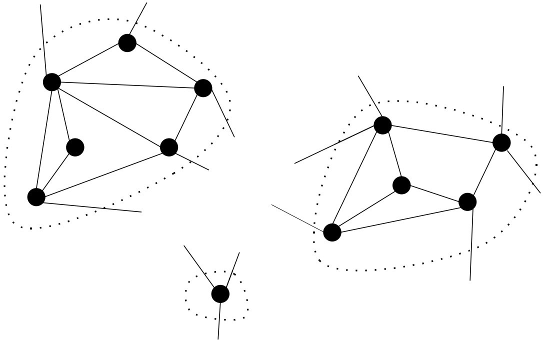
Partial Embeddings



Choose the (parallel) embedding order such that the non-embedded vertices are connected.

- all half-embedded edges lie in one face

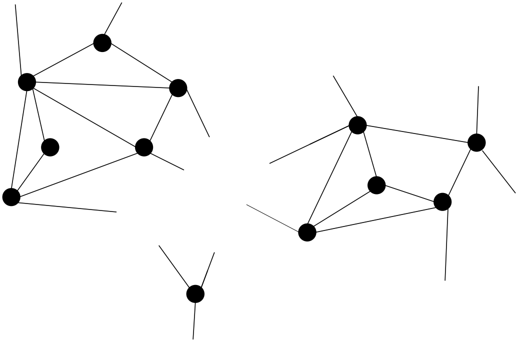
Partial Embeddings



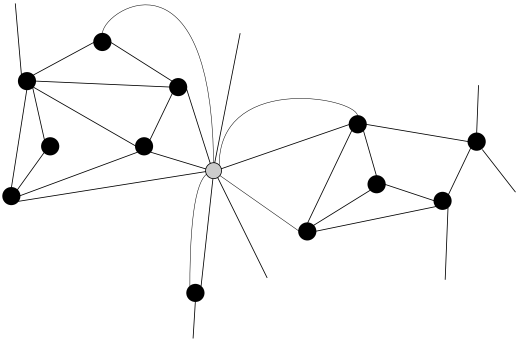
Choose the (parallel) embedding order such that the non-embedded vertices are connected.

- all half-embedded edges lie in one face
- partial embedding: rotation of half-embedded edges around each connected component

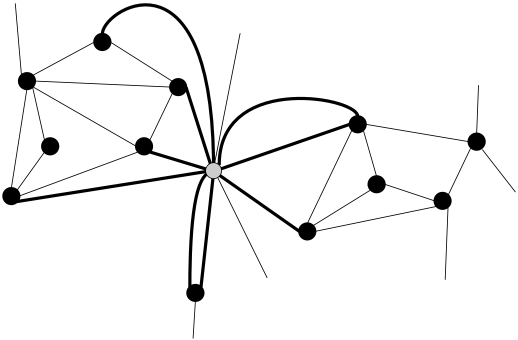
Vertex Addition



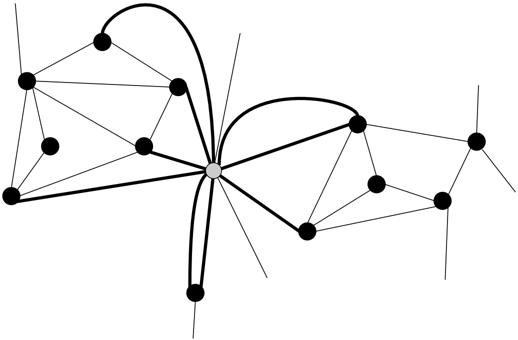
Vertex Addition



Vertex Addition



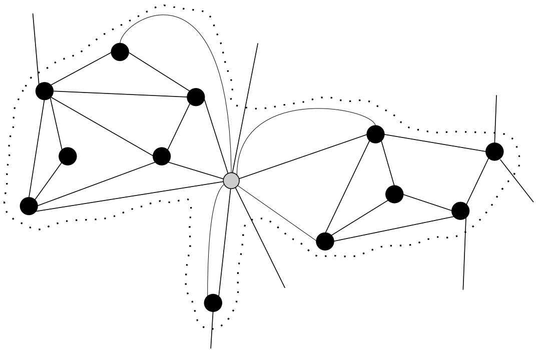
Vertex Addition



Observations:

- new-embedded edges have to be consecutive

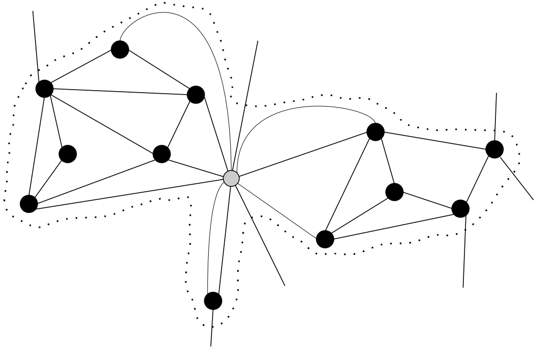
Vertex Addition



Observations:

- new-embedded edges have to be consecutive
- all possible cyclic orders of a component can be obtained by
  - flipping biconnected components
  - permutation on cut-vertices

Vertex Addition



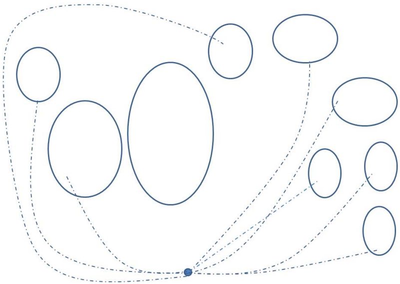
Observations:

- new-embedded edges have to be consecutive
- all possible cyclic orders of a component can be obtained by
  - flipping biconnected components
  - permutation on cut-vertices

Efficient **Distributed** Planar Embedding Algorithm

Algorithm:

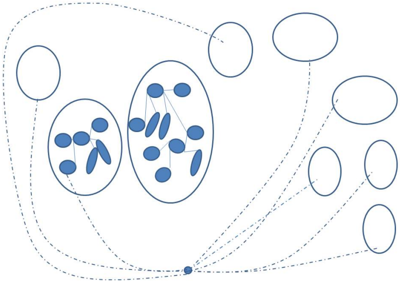
- recursively subdivide into balanced, low-diameter subproblems



Efficient **Distributed** Planar Embedding Algorithm

Algorithm:

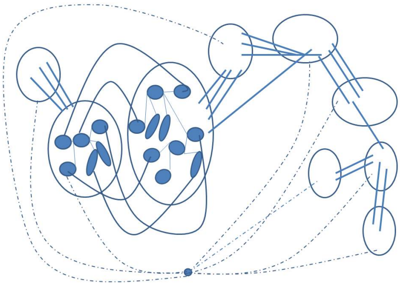
- recursively subdivide into balanced, low-diameter subproblems
- locally maintain and update biconnected component structure



Efficient **Distributed** Planar Embedding Algorithm

Algorithm:

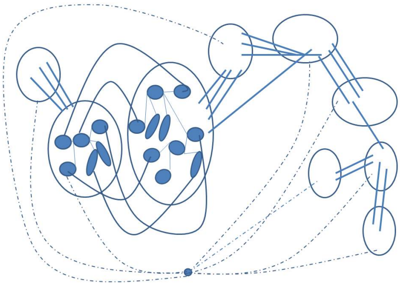
- recursively subdivide into balanced, low-diameter subproblems
- locally maintain and update biconnected component structure
- track and propagate all possible flippings



Efficient **Distributed** Planar Embedding Algorithm

Algorithm:

- recursively subdivide into balanced, low-diameter subproblems
- locally maintain and update biconnected component structure
- track and propagate all possible flippings



Claim 1:

There is a $\tilde{O}(D)$ distributed planar embedding algorithm.

New Tools II: Low-Congestion Shortcuts

Uniquely Distributed Problem:

- Need to preserve diameter of subproblems (e.g., Divide-and-Conquer)

New Tools II: Low-Congestion Shortcuts

Uniquely Distributed Problem:

- Need to preserve diameter of subproblems (e.g., Divide-and-Conquer)

Idea:

- Add extra edges to each subproblem to ensure low diameter.
- Avoid congestion, i.e., not use any edge too often

New Tools II: Low-Congestion Shortcuts

Uniquely Distributed Problem:

- Need to preserve diameter of subproblems (e.g., Divide-and-Conquer)

Definition: c-congestion d-dilation shortcuts

Given planar $G = (V, E)$ and partition $S_1, \ldots, S_N \subset V$ with $G[S_i]$ connected. $H_1, \ldots, H_N \subset G$ are (c, d)-shortcuts iff:

(1) $\forall i$: diameter of $G[S_i] + H_i$ is at most d.

(2) Each edge $e \in E$ is in at most c of the subgraphs H_i.

New Tools II: Low-Congestion Shortcuts

Uniquely Distributed Problem:

- Need to preserve diameter of subproblems (e.g., Divide-and-Conquer)

Definition: c-congestion d-dilation shortcuts

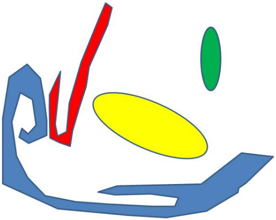
Given planar $G = (V, E)$ and partition $S_1, \ldots, S_N \subset V$ with $G[S_i]$ connected. $H_1, \ldots, H_N \subset G$ are (c, d)-shortcuts iff:

(1) $\forall i$: diameter of $G[S_i] + H_i$ is at most d.

(2) Each edge $e \in E$ is in at most c of the subgraphs H_i.

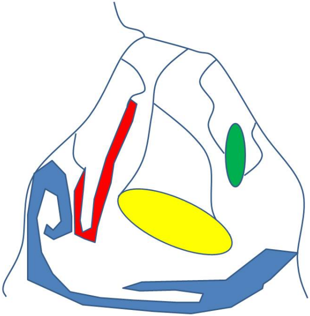
Claim 2:

$(D \log D, D \log D)$-Shortcuts exist, can be computed distributedly in $\tilde{O}(D)$ rounds, and are essentially best possible.



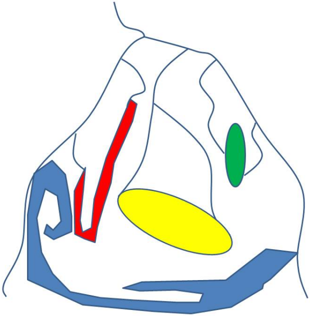
Simple Construction:

New Tools II: Low-Congestion Shortcuts



Simple Construction:

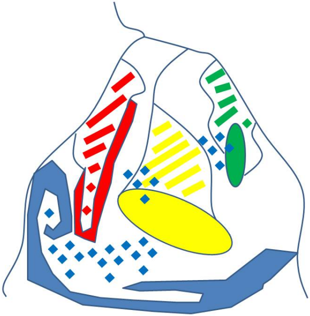
- Compute Planar Embedding
- Build a Left-First BFS-tree T



Simple Construction:

- Compute Planar Embedding
- Build a Left-First BFS-tree T
- For each S_i determine left-most and right-most node l_i and r_i
- (l_i, r_i)-path closes a (fundamental) cycle C in T

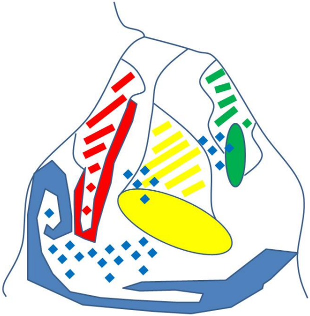
New Tools II: Low-Congestion Shortcuts



Simple Construction:

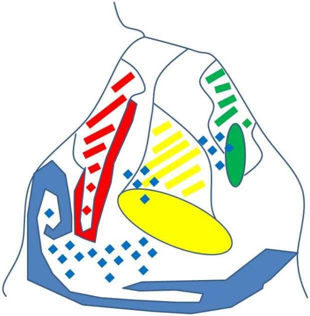
- Compute Planar Embedding
- Build a Left-First BFS-tree T
- For each S_i determine left-most and right-most node l_i and r_i
- (l_i, r_i)-path closes a (fundamental) cycle C in T
- $H_i =$ any edge above S_i if enclosed by C (or beneath S_i)

New Tools II: Low-Congestion Shortcuts



Simplified Analysis:

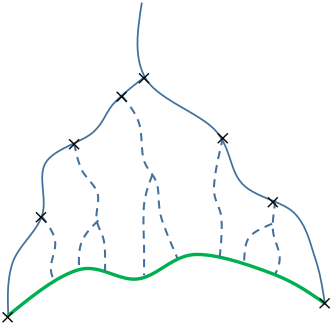
New Tools II: Low-Congestion Shortcuts



Simplified Analysis:

- Congestion $\leq D$:
  - embedded BFS-tree induces a left-right order
  - set to the left/right do not share edges
  - at most D subsets are above/below

New Tools II: Low-Congestion Shortcuts



Simplified Analysis:

- Congestion $\leq D$:
  - embedded BFS-tree induces a left-right order
  - set to the left/right do not share edges
  - at most D subsets are above/below
- Dilation D^2:
  - Project any S_i-path P_i onto its T-path
  - Shortcut as far as possible within the enclosed subtree of T
  - At most D shortcuts each of lenght D needed

An Application: MST in planar networks

Claim 3:

For planar networks there is a $\tilde{O}(D)$ distributed MST algorithm.

An Application: MST in planar networks

Claim 3:

For planar networks there is a $\tilde{O}(D)$ distributed MST algorithm.

Algorithm:

- Compute an embedding
- Boruvka's algorithm:

An Application: MST in planar networks

Claim 3:

For planar networks there is a $\tilde{O}(D)$ distributed MST algorithm.

Algorithm:

- Compute an embedding
- Boruvka's algorithm:

  Start with singleton components

  Repeat ($\log n$ times)

  - each component adds cheapest out-going edge
  - merge components

An Application: MST in planar networks

Claim 3:

For planar networks there is a $\tilde{O}(D)$ distributed MST algorithm.

Algorithm:

- Compute an embedding
- Boruvka's algorithm:

  Start with singleton components

  Repeat ($\log n$ times)

  - each component adds cheapest out-going edge
  - merge components
    - compute short-cuts
    - compute $O(c + d) = O(D \log D)$ scheme
  - $\implies$ find new cheapest edge in $\tilde{O}(D)$ rounds

Summary and Open Questions

Take-Home Message

- HUGE untapped potential for planar CONGEST algorithms

Summary and Open Questions

Take-Home Message

- HUGE untapped potential for planar CONGEST algorithms
- distributed perspective leads to interesting new questions, insights, problems, and solutions

Summary and Open Questions

Take-Home Message

- HUGE untapped potential for planar CONGEST algorithms
- distributed perspective leads to interesting new questions, insights, problems, and solutions

Goals:

- $\tilde{O}(D)$ or $O(D + \log^{O(1)} n)$ distributed algorithms for planar networks
- Distributed algorithmic toolbox

Summary and Open Questions

Take-Home Message

- HUGE untapped potential for planar CONGEST algorithms
- distributed perspective leads to interesting new questions, insights, problems, and solutions

Goals:

- $\tilde{O}(D)$ or $O(D + \log^{O(1)} n)$ distributed algorithms for planar networks
- Distributed algorithmic toolbox

Open Questions:

- Maximum Flow in $o(n)$
- Exact / Approximate Shortest-Paths in $\tilde{o}(\sqrt{n})$
- Depth-First-Search Trees in $o(n)$
- Separators (construction, useful definitions)
- Extensions, e.g., to bounded genus or excluded minor graph classes
- ...

Thank you!

Questions?