A Polynomial-time Bicriteria Approximation Scheme for Planar Bisection

Kyle Fox  
Duke University

Philip N. Klein  
Brown University

Shay Mozes  
IDC Herzliya
Minimum Graph Bisection
Minimum Graph Bisection

Evenly partition vertices
Minimum Graph Bisection

- Minimize # crossing edges
- Evenly partition vertices
• NP-hard for general graphs [Garey, Johnson, Stockmeyer ’76]
• **NP-hard for general graphs** [Garey, Johnson, Stockmeyer ’76]

• **O(\log n)**-approximation [Räcke ’08]
Planar Graphs

• Many problems that are hard in general have approximation schemes
Planar Graphs

• Many problems that are hard in general have approximation schemes

• Cut (bipartition) problems in particular become much easier
Planar Graphs

• Many problems that are hard in general have approximation schemes

• Cut (bipartition) problems in particular become much easier

• May even exists a poly-time algorithm for minimum bisection
Our Results

• A *bicriteria approximation scheme* for minimum bisection in planar graphs
Our Results

• A bicriteria approximation scheme for minimum bisection in planar graphs
  ◦ Let $\text{OPT}$ be the cost of the minimum bisection.
Our Results

- A bicriteria approximation scheme for minimum bisection in planar graphs
  - Let $OPT$ be the cost of the minimum bisection.
    
    Our algorithm returns a bipartition with at least $(1/2 - \varepsilon) \ n$ vertices on each side
Our Results

• A *bicriteria approximation scheme* for minimum bisection in planar graphs

  ◦ Let $\text{OPT}$ be the cost of the minimum bisection.

    Our algorithm returns a bipartition with at least $(1/2 - \varepsilon) n$ vertices on each side

    and total cost at most $(1 + \varepsilon) \text{OPT}$
Our Results

- A *bicriteria approximation scheme* for minimum bisection in planar graphs

  - Let $OPT$ be the cost of the minimum bisection.

  Our algorithm returns a bipartition with at least $(1/2 - \varepsilon) n$ vertices on each side and total cost at most $(1 + \varepsilon) OPT$ in polynomial time, for any constant $\varepsilon > 0$. 


Our Results

- A *bicriteria approximation scheme* for minimum bisection in planar graphs
- Works with arbitrary non-negative *edge costs* and non-negative *vertex weights*
Our Results

- A *bicriteria approximation scheme* for minimum bisection in planar graphs

- Works with arbitrary non-negative *edge costs* and non-negative *vertex weights*

- Also gives a bicriteria approximation scheme for *minimum b-balanced* cut in planar graphs
Our Results

• A bicriteria approximation scheme for minimum bisection in planar graphs

• Works with arbitrary non-negative edge costs and non-negative vertex weights

• Also gives a bicriteria approximation scheme for minimum $b$-balanced cut in planar graphs

- Previously known: a 2-approximation algorithm that requires $b \leq 1/3$ [Garg, Saran, Vazirani ’99]
Approximations

- Approximation schemes known for many problems in planar graphs
Approximations

• Approximation schemes known for many problems in planar graphs

• Ours is the first problem involving balance
The Framework

• Planar graph approximations often follow a single framework [Klein ’08]
The Framework

• Planar graph approximations often follow a single framework [Klein ’08]

• To use the framework for bisection, all we had to do was provide a construction for the sparsifier
• Contracting an edge merges its endpoints
Sparsifier
Sparsifier

• Contraction with $\tilde{O}(f(\epsilon) \cdot \text{OPT})$ edges
Sparsifier

- Contraction with $\tilde{O}(f(\epsilon) \cdot \text{OPT})$ edges
- Cost of optimal (near-)bisection goes up by at most $1+\epsilon$ factor
Duality

- Vertices ⇔ faces
- Edges ⇔ edges
- Faces ⇔ vertices
- Edge contraction ⇔ edge deletion
Cut-Cycle Duality
Cut-Cycle Duality
Cut-Cycle Duality
Cut-Cycle Duality
Cut-Cycle Duality
Cut-Cycle Duality
• In dual, optimal bisection is a collection of cycles enclosing half the faces
• In dual, optimal bisection is a collection of cycles enclosing half the faces

• Contractions in the input graph are deletions in its dual
• In dual, optimal bisection is a collection of cycles enclosing half the faces

• Contractions in the input graph are deletions in its dual

• So in the dual, the sparsifier is a subgraph
• Goal: Given a planar graph $G$ (the dual), find a subgraph $H$ (the sparsifier) such that
• Goal: Given a planar graph \( G \) (the dual), find a subgraph \( H \) (the sparsifier) such that
  
  ☐ Cost of \( H \) is \( \tilde{O}(f(\epsilon) \cdot \text{OPT}) \)
• Goal: Given a planar graph $G$ (the dual), find a subgraph $H$ (the sparsifier) such that

- Cost of $H$ is $\tilde{O}(f(\epsilon) \cdot \text{OPT})$

- $H$ contains a cheap collection of cycles enclosing (roughly) half the faces
• Goal: Given a planar graph $G$ (the dual), find a subgraph $H$ (the sparsifier) such that
  
  ⬤ Cost of $H$ is $\tilde{O}(f(\epsilon) \cdot \text{OPT})$

  ⬤ $H$ contains a cheap collection of cycles enclosing (roughly) half the faces

  ‣ “Cheap” = $(1 + \epsilon) \cdot \text{OPT}$
• Goal: Given a planar graph $G$ (the dual), find a subgraph $H$ (the sparsifier) such that

- Cost of $H$ is $\tilde{O}(f(\epsilon) \cdot \text{OPT})$

- $H$ contains a cheap collection of cycles enclosing (roughly) half the faces
  
  - “Cheap” = $(1 + \epsilon) \cdot \text{OPT}$
  - “Roughly” = to within $(1 + \epsilon)$ factor
$: \text{optimal bissection cycle}$
○ : optimal **bisection cycle**
图画：optimal bisection cycle
○ : optimal bisection cycle
〇 : optimal bisection cycle
○ : optimal bisection cycle
○: optimal bisection cycle
Boundary-to-boundary Spanner

[Klein '06]
Boundary-to-boundary Spanner

[Klein '06]
Boundary-to-boundary Spanner

Shortest path
Boundary-to-boundary Spanner

Shortest path

Spanner path
○ : optimal bisection cycle
O : optimal \textit{bisection cycle}
○ : optimal bisection cycle
○ : skeleton cycle
○ : optimal **bisection cycle**
○ : **skeleton cycle**
○ : optimal bisection cycle
○ : skeleton cycle
**: optimal bisection cycle**

**: skeleton cycle**

**: sparsifier cycle**
○ : optimal bisection cycle
○ : skeleton cycle
● : sparsifier cycle
Bisection Algorithm

1. Find a sparsifier with $\tilde{O}(f(\epsilon) \cdot \text{OPT})$ edges

2. Apply remaining steps of sparsifier framework
Bisection Algorithm

1. Find a sparsifier with $\tilde{O}(f(\epsilon) \cdot \text{OPT})$ edges
   
   a. Find many skeleton cycles in dual graph

2. Apply remaining steps of sparsifier framework
Bisection Algorithm

1. Find a sparsifier with $\tilde{O}(f(\epsilon) \cdot \text{OPT})$ edges
   a. Find many skeleton cycles in dual graph
   b. Add edges for boundary-to-boundary spanner paths

2. Apply remaining steps of sparsifier framework
○ : optimal **bisection cycle**
○ : **skeleton cycle**
● : **sparsifier cycle**
○: optimal **bisection cycle**
○: **skeleton cycle**
○○: **sparsifier cycle**
Perturbation cycle:

- optimal bisection cycle
- skeleton cycle
- sparsifier cycle
Understanding Cycles

• Need to understand the cost and weight of cycles
Understanding Cycles

• Need to understand the cost and weight of cycles
  \[ \text{cost}(C) := \text{number of edges along the cycle } C \]
Understanding Cycles

• Need to understand the *cost* and *weight* of cycles

  - \(\text{cost}(C) := \text{number of edges along the cycle } C\)
  - \(\text{weight}(C) := \text{number of faces enclosed by } C\)
Understanding Cycles

- Need to understand the *cost* and *weight* of cycles
  - \( \text{cost}(C) := \text{number of edges along the cycle } C \)
  - \( \text{weight}(C) := \text{number of faces enclosed by } C \)
  - \( \text{ratio}(C) := \text{cost}(C) / \text{weight}(C) \)
Ratio is the Key

**Claim:** Removing all cycles of high ratio (at least \( \frac{OPT}{\varepsilon n} \)) from the optimal solution causes at most \( \varepsilon n \) faces to switch sides.
Ratio is the Key

**Claim:** Removing all cycles of *high ratio* (at least \( \frac{\text{OPT}}{\varepsilon n} \)) from the optimal solution causes at most \( \varepsilon n \) faces to switch sides.

**Proof:**

- Faces outside these cycles don’t change sides.
Ratio is the Key

Claim: Removing all cycles of high ratio (at least OPT / (εn)) from the optimal solution causes at most εn faces to switch sides.

Proof:

• Faces outside these cycles don’t change sides

• Optimal bisection cycles have total cost OPT
Ratio is the Key

**Claim:** Removing all cycles of *high ratio* (at least $\text{OPT} / (\varepsilon n)$) from the optimal solution causes at most $\varepsilon n$ faces to switch sides.

**Proof:**

- Faces outside these cycles don’t change sides.
- Optimal bisection cycles have total cost $\text{OPT}$.
- So total weight inside high-ratio cycles is at most $\text{OPT} * (\varepsilon n) / \text{OPT} = \varepsilon n$. 
• Can show high-ratio perturbation cycles have total weight $O(\varepsilon n)$
• Can show high-ratio perturbation cycles have total weight $O(\epsilon n)$

• Main technical idea: Guarantee all perturbation cycles have high ratio
• Can show high-ratio perturbation cycles have total weight $O(\varepsilon n)$

• Main technical idea: Guarantee all perturbation cycles have high ratio

by exhaustively adding cycles with low ratio to the skeleton.
○ : optimal *bisection cycle*
○ : *skeleton cycle*
● : *spanner cycle*
- optimal bisction cycle
- skeleton cycle
- spanner cycle
○ : optimal bisection cycle
○ : skeleton cycle
○ ○ : spanner cycle
○ : optimal **bisection cycle**
○ : **skeleton cycle**
● : **spanner cycle**
- optimal **bisection cycle**
- **skeleton cycle**
- **spanner cycle**
- optimal bisection cycle
- skeleton cycle
- spanner cycle
- optimal bisection cycle
- skeleton cycle
- spanner cycle
- optimal *bisection cycle*
- skeleton cycle
- spanner cycle
○: optimal *bisection cycle*
○: *skeleton cycle*
●: *spanner cycle*
○ : optimal **bisection cycle**
○ : **skeleton cycle**
■ : **spanner cycle**
Perturbation cycle

- optimal *bisection cycle*
- skeleton cycle
- sparsifier cycle
• : optimal *bisection cycle*
• : *skeleton cycle*
• : *sparsifier cycle* Not in the skeleton ⇒ high ratio!
Spanner Paths Between Different Boundary
Spanner Paths Between Different Boundary
Spanner Paths Between Different Boundary
Spanner Paths Between Different Boundary

Use PC-clustering [Bateni, Hajiaghayi, Marx ’11]
Claim: If skeleton tree has depth $d$, then the total cost of the skeleton is $O(d/\varepsilon \text{ OPT})$
Keeping the Skeleton Cheap

Claim: If skeleton tree has depth $d$, then the total cost of the skeleton is $O(d/\varepsilon \OPT)$

Proof:

• Skeleton cycles have ratio at most $\OPT / (\varepsilon n)$
Keeping the Skeleton Cheap

**Claim:** If skeleton tree has depth $d$, then the total cost of the skeleton is $O(d/\varepsilon \text{ OPT})$

**Proof:**

- Skeleton cycles have ratio at most $\text{OPT} / (\varepsilon n)$
- Total weight inside skeleton cycles is $O(dn)$
Keeping the Skeleton Cheap

**Claim:** If skeleton tree has depth $d$, then the total cost of the skeleton is $O(d/\varepsilon \text{ OPT})$

**Proof:**

- Skeleton cycles have ratio at most $OPT / (\varepsilon n)$
- Total weight inside skeleton cycles is $O(dn)$
- So total cost is at most $O(dn) \times \text{OPT} / (\varepsilon n)$
○ : optimal **bisection cycle**
○ : **skeleton cycle**
● : **spanner cycle**
○ : optimal **bisection cycle**
○ : **skeleton cycle**
★ : **spanner cycle**
○ : optimal *bisection cycle*
○ : skeleton cycle
● : spanner cycle
Conclusions and Open Problems

- We found a *bicriteria approximation scheme* for minimum bisection in planar graphs
Conclusions and Open Problems

- We found a *bicriteria approximation scheme* for minimum bisection in planar graphs
  - Runs in time $n^{\text{poly}(1/\varepsilon)}$
Conclusions and Open Problems

• We found a *bicriteria approximation scheme* for minimum bisection in planar graphs
  - Runs in time $n^{\text{poly}(1/\varepsilon)}$

• Is there an *efficient* PTAS, i.e. one with running time $f(\varepsilon) \cdot n^{O(1)}$?
Conclusions and Open Problems

- We found a *bicriteria approximation scheme* for minimum bisection in planar graphs
  - Runs in time $n^{\text{poly}(1/\epsilon)}$
- Is there an *efficient* PTAS, i.e. one with running time $f(\epsilon) \cdot n^{O(1)}$?
- Is there a PTAS that returns a perfectly balanced solution?
Conclusions and Open Problems

• We found a *bicriteria approximation scheme* for minimum bisection in planar graphs
  - Runs in time $n^{\text{poly}(1/\varepsilon)}$

• Is there an *efficient* PTAS, i.e. one with running time $f(\varepsilon) \cdot n^{O(1)}$?

• Is there a PTAS that returns a perfectly balanced solution?

• Is minimum bisection actually poly-time solvable in planar graphs?
Thank you!