APMF < APSP?

Gomory-Hu Tree in Subcubic Time

Which are easier to compute: distances or connectivities?

Shortest-Path\((s, t) = \) ?
Max-Flow\((s, t) = \) ?

With Robert Krauthgamer (Weizmann) and Ohad Trabelsi (Michigan)
[SODA'20, FOCS'20, STOC'21, FOCS'21, SODA'22]

+ [new paper] also with Jason Li (Simons), Debmalya Panigrahi (Duke), and Thatchaphol Saranurak (Michigan)

Amir Abboud (Weizmann)
Max-Flow vs. Shortest Path

Which is easier to compute?

Max-Flow $(s, t) = 11$

Shortest-Path $(s, t) = 16$

[Dinitz’70] $mn$

[Goldberg-Rao’98] $mn^{2/3}$

[Lee-Sidford’14] $mn^{1/2}$

via Continuous Optimization

[BLLSSSW’21] $\tilde{O}(m + n^{1.5})$

$1950’s$ Dijkstra’s $O(m + n \log n)$

Both $\tilde{O}(n^2)$ but shortest path is much simpler.
Max-Flow vs. Shortest Path

Which is easier to compute?

Max-Flow\( (s, t) = 3 \)

Shortest-Path\( (s, t) = 4 \)

via Randomized Contractions
[Karger-Levine’02] \( \tilde{O}(m + n^2) \)

via Continuous Optimization
[BLLSSSW’21] \( \tilde{O}(m + n^{1.5}) \)

Both \( \tilde{O}(n^2) \) but shortest path is much simpler.
Max-Flow vs. Shortest Path

Which is easier to compute?

Max-Flow \((s, t) = 3\)

Unweighted (simple graphs)

Shortest-Path \((s, t) = 4\)

Both work for directed graphs too.

via Randomized Contractions
[Karger-Levine’02] \(\tilde{O}(m + n^2)\)

via Continuous Optimization
[BLLSSSW’21] \(\tilde{O}(m + n^{1.5})\)

1950’s BFS
\(O(m + n)\)

Also solves the Single-Source version.

Both \(\tilde{\Theta}(n^2)\) but shortest path is much simpler.
All-Pairs MF vs. All-Pairs SP

**APMF:** \( \forall s, t \in V : \text{Max-Flow}(s, t) = ? \)

**APSP:** \( \forall s, t \in V : \text{Shortest-Path}(s, t) = ? \)

Which is easier to compute?
All-Pairs MF vs. All-Pairs SP

**APMF:** \( \forall s, t \in V : \text{Max-Flow}(s, t) = ? \)

**APSP:** \( \forall s, t \in V : \text{Shortest-Path}(s, t) = ? \)

**Trivial:** \( n^2 \cdot MF(n) = \tilde{O}(n^4) \)

**Trivial 2:** \( n^2 \cdot SP(n) = \tilde{O}(n^4) \)

**Gomory-Hu 1961:**
\[
(n - 1) \cdot MF(n) = \tilde{O}(n^3)
\]

Which is easier to compute?
Thm [GH 1961]:
Every undirected graph has a (weighted) cut-equivalent tree. Moreover, it can be computed in $(n - 1) \cdot MF(n)$ time.

∀s, t ∈ V : Min-Cut$_G$(s, t) = Min-Cut$_T$(s, t)

APMF on a tree is in $\tilde{O}(n^2)$ time. The challenge is to compute a GH tree...
Gomory-Hu Tree

Thm [GH 1961]:
Every undirected graph has a (weighted) cut-equivalent tree. Moreover, it can be computed in \((n - 1) \cdot MF(n)\) time.

\(\forall s, t \in V : \text{Min-Cut}_G(s, t) = \text{Min-Cut}_T(s, t)\)

- \(\Rightarrow\) APMF has only \(n-1\) answers.
- Space-optimal min-cut oracle.
- First graph sparsification result?

Impossible for APSP.
All-Pairs MF vs. All-Pairs SP

Gomory-Hu 1961:
\[(n - 1) \cdot MF(n) = \tilde{O}(n^3)\]

Open: \(o(n) \cdot MF(n)\)?

<table>
<thead>
<tr>
<th>Author</th>
<th>Runtime</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fredman</td>
<td>(n^3 \log \frac{\log n}{\log n^{1/3}})</td>
<td>1976</td>
</tr>
<tr>
<td>Takaoka</td>
<td>(n^3 \log \frac{\log n}{\log n^{1/2}})</td>
<td>1992</td>
</tr>
<tr>
<td>Dobosiewicz</td>
<td>(n^3 / \log n^{1/2})</td>
<td>1992</td>
</tr>
<tr>
<td>Han</td>
<td>(n^3 \log \frac{\log n^{5/7}}{\log n^{5/7}})</td>
<td>2004</td>
</tr>
<tr>
<td>Takaoka</td>
<td>(n^3 \log \frac{\log n^2}{\log n})</td>
<td>2004</td>
</tr>
<tr>
<td>Zwick</td>
<td>(n^3 \log \frac{\log n^{1/2}}{\log n})</td>
<td>2004</td>
</tr>
<tr>
<td>Chan</td>
<td>(n^3 / \log n)</td>
<td>2005</td>
</tr>
<tr>
<td>Han</td>
<td>(n^3 \log \frac{\log n^{5/4}}{\log n^{5/4}})</td>
<td>2006</td>
</tr>
<tr>
<td>Chan</td>
<td>(n^3 \log \frac{\log n^3}{\log n^2})</td>
<td>2007</td>
</tr>
<tr>
<td>Han, Takaoka</td>
<td>(n^3 \log \frac{\log n}{\log n^2})</td>
<td>2012</td>
</tr>
<tr>
<td>Williams</td>
<td>(n^3 / 2^{\Omega(\sqrt{\log n})})</td>
<td>2014</td>
</tr>
</tbody>
</table>

Meanwhile...
APSP in “mildly sub-cubic time”

Both \(\tilde{O}(n^3)\) but APSP seems easier.
All-Pairs MF vs. All-Pairs SP

Which is easier to compute?

**APMF:** \( \forall s, t \in V : \text{Max-Flow}(s, t) = ? \)

**APSP:** \( \forall s, t \in V : \text{Shortest-Path}(s, t) = ? \)

Trivial:

\[
n^2 \cdot MF(n) = \tilde{O}(n^4)
\]

Trivial 2:

Gomory-Hu 1961:

\[
(n - 1) \cdot MF(n) = \tilde{O}(n^3)
\]

Both \( \tilde{O}(n^3) \) but APSP seems easier.
All-Pairs MF vs. All-Pairs SP

Unweighted (simple graphs)

Gomory-Hu 1961:
\[(n - 1) \cdot MF(n) = \tilde{O}(n^3)\]

[Bhalgat-Hariharan-Kavitha-Panigrahi '07]
\[\tilde{O}(mn) \ (without \ MF)\]

Siedel 1995:
\[O(n^\omega) = O(n^{2.37})\]

APSP is subcubic, APMF is not: a separation, finally?
<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^3$</td>
<td>$n^{2.37}$</td>
</tr>
</tbody>
</table>

It feels like $\text{APMF} \geq \text{APSP}$...

Is it so?
Enter:
Fine-Grained Complexity

A small set of “conjectures”
(APSP, 3SUM, SETH, ...)

fine-grained reductions

Tight lower bounds for lots of natural and important problems

\( \Omega(n^3) \), \( \Omega(n^{2.5}) \), \( \Omega(n^2) \), ...
The Class P (before)

- k-clique
- Radius
- RNA folding
- LCS
- Diameter
- 3SUM
- Orthogonal Vectors
- Edit-Distance
- Maximum Matching
- Dynamic reachability
- Local Alignment
- CFG Parsing
- Polygon containment
- All Pairs Max Flow
- Frechet distance
- All Pairs Shortest Paths
- Linear Programming
- 3SUM

...
The Class P (after)

- **k-SAT**
  - Diameter
  - Closest Pair
  - Local Alignment
  - Dynamic Reachability
  - Single-Source Max-Flow
  - Subtree Isomorphism
  - Stable Matching
  - Edit-Distance
  - Frechet
  - LCS
  - ...

- **3SUM**
  - Colinearity
  - Polygon Containment
  - Strips Cover Rectangle
  - Triangle Enumeration
  - Compressed Inner Product
  - Dynamic Max Matching
  - Set Intersection
  - ...

- **APSP**
  - Radius
  - Dynamic Max Matching
  - Stochastic Context-Free Grammar Parsing
  - Negative Triangle
  - Dynamic Max Flow
  - Replacement Paths
  - Median
  - ...

...
Conjecture:
APSP in undirected weighted graphs cannot be solved in $O(n^{3-\varepsilon})$ time.

[Vassilevska Williams - Williams ’10]

It feels like $\text{APMF} \geq \text{APSP}$...
Is it so?
**APMF > APSP in directed graphs**

Assuming **SETH**:

- \( \Omega(n^{2-\varepsilon}) \) for Single-Source MF in sparse graphs \[ A.- Vassilevska W. - Yu ’15 \]
  vs. \( \tilde{O}(n) \) for Single-Source SP in sparse graphs

- \( \Omega(n^{3-\varepsilon}) \) for **APMF** in sparse graphs \[ Krauthgamer-Trablesi ’17 \]
  vs. \( \tilde{O}(n^2) \) for APSP in sparse graphs

Assuming the **4-Clique** Conjecture:

- \( \Omega(n^{\omega+1-\varepsilon}) = \Omega(n^{3.37}) \) for **APMF** in dense graphs \[ AGIKPTUW ’19 \]
  vs. \( \tilde{O}(n^3) \) for APSP in dense graphs

*APMF is indeed harder than APSP... in directed graphs.*
**APMF > APSP in directed graphs**

**Assuming SETH:**

\[ \Omega(n^{2-\varepsilon}) \] for Single-Source MF in sparse graphs  \[ \text{[A.- Vassilevska W. - Yu ’15]} \]

vs. \( \tilde{O}(n) \) for Single-Source SP in sparse graphs

\[ \Omega(n^{3-\varepsilon}) \] for \textbf{APMF} in sparse graphs  \[ \text{[Krauthgamer-Trablesi ’17]} \]

vs. \( \tilde{O}(n^2) \) for APSP in sparse graphs

**Assuming the 4-Clique Conjecture:**

\[ \Omega(n^{\omega+1-\varepsilon}) = \Omega(n^{3.37}) \] for \textbf{APMF} in dense graphs  \[ \text{[AGIKPTUW ’19]} \]

vs. \( \tilde{O}(n^3) \) for APSP in dense graphs

\[ \text{[A.- Krauthgamer - Trabelsi SODA’20]:} \]
Same lower bounds for undirected graphs with \textit{node capacities}.  

**APMF > APSP?**

*APMF is indeed harder than APSP...*

<table>
<thead>
<tr>
<th>Undirected Graphs</th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^3$</td>
<td>$n^{2.37}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other settings</th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directed</td>
<td>$\Omega(n^{3.37}), O(n^4)$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Node-capacities</td>
<td>$\Omega(n^{3.37}), O(n^4)$</td>
<td>$n^3$</td>
</tr>
</tbody>
</table>

**[Mayeda’62, Jelinek’63, Hassin-Levine’07]**

GH Trees are impossible in these settings.

*(Because there are $\Omega(n^2)$ answers.)*
**APMF > APSP?**

<table>
<thead>
<tr>
<th>Undirected Graphs</th>
<th><strong>APMF</strong></th>
<th><strong>APSP</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^3$</td>
<td>$n^{2.37}$</td>
</tr>
</tbody>
</table>

**Open Question:**
Prove any “fine-grained complexity” lower bounds for APMF.
Me 10-years ago: “why work on APMF vs. any other problem?”
Fine-Grained Complexity: “APMF is really worth studying...”
Non-reducibility?

Open Question:
Prove any “fine-grained complexity” lower bounds for APMF.

[Carmosino - Gao - Impagliazzo - Mikhailin - Paturi - Schneider ’16]
“No reduction between problems with different non-deterministic complexity (assuming NSETH)”

Are we failing to prove hardness because APMF is easy nondeterministically?

[A.- Krauthgamer - Trabelsi SODA’20]:
There is a prover-verifier protocol for constructing a GH-Tree of an unweighted graph, where the verifier only takes $\tilde{O}(m)$ time.
## Nondeterministic Algorithms

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Real algorithms</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^3$</td>
<td>$n^{2.37}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Nondeterministic algorithms</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^{2.94}$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^2$</td>
<td>$n^{2.37}$</td>
</tr>
</tbody>
</table>

[CGIMPS ’16]

[AKT ’20]

*Did we really need the nondeterminism?*
Towards Faster APMF

Did we really need the nondeterminism?

[A.- Krauthgamer - Trabelsi FOCS’20]:


2. $(1 + \varepsilon)$-**APMF** in $\tilde{O}(n^2)$ time. * Also $(1 + \varepsilon)$-**GHT** in $\tilde{O}(MF(n))$.

3. “Cut-Equivalent Trees are Optimal for Min-Cut Queries”

* [Li-Panigrahi STOC’21]
### APMF vs. APSP

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exact</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^3$</td>
<td>$n^{2.37}$</td>
</tr>
<tr>
<td><strong>Approximation</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>$n^2$</td>
</tr>
<tr>
<td>$poly \log n$</td>
<td>$n^2$</td>
<td></td>
</tr>
<tr>
<td>$(1 + \varepsilon)$</td>
<td>$n^2$</td>
<td>$n^{2.37}$</td>
</tr>
<tr>
<td></td>
<td>[AKT ’20,LP’21]</td>
<td>[Zwick’02]</td>
</tr>
</tbody>
</table>
Towards Faster APMF

All we have to do is solve Single-Source Max-Flow...

[A.- Krauthgamer - Trabelsi FOCS’20]:


2. $(1 + \epsilon)$-APMF in $\tilde{O}(n^2)$ time. * Also $(1 + \epsilon)$-GHT in $\tilde{O}(MF(n))$.

3. “Cut-Equivalent Trees are Optimal for Min-Cut Queries”

* [Li-Panigrahi STOC’21]
Breaking the Cubic Barrier

[A.- Krauthgamer - Trabelsi STOC’21]:

**GH Tree** in $\tilde{O}(n^{2.5})$ time for simple graphs.

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^{2.5}$</td>
<td>$n^{2.37}$</td>
</tr>
</tbody>
</table>

Still APMF > APSP but we are not so sure anymore...

Main tools:
- Expander-decomposition
- “Isolating Cuts”
APMF < APSP for simple graphs?

[Main tool:
Expander-decomposition with vertex demands]

GH Tree in $n^{2+o(1)}$ time for simple graphs.

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>$n^3$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^2$</td>
<td>$n^{2.37}$ [Siedel ’95]</td>
</tr>
</tbody>
</table>

✓ Optimal for APMF.
✓ All-Pairs in single-pair time!
✓ Can be derandomized [AKT’21]
Still open: GH Tree in $\tilde{O}(\text{Max-Flow-Time})$?

$n^{2+o(1)} \rightarrow m^{1+o(1)}$

So far only known for bounded genus, and bounded tree width graphs.

[A.- Krauthgamer - Trabelsi FOCS’21]

$\Omega(m + n^{1.5})$ for GH Tree in simple graphs,
assuming $\Omega(n^3)$ for multigraphs.

[A.- Krauthgamer - Trabelsi SODA’22]

$(m + n^{1.9})^{1+o(1)}$ for GH Tree in simple graphs,
via new “Friendly Cut Sparsifiers”.

Main Open Question:
APMF in subcubic time for general graphs?
Main Open Question:
APMF in subcubic time for general graphs?

Would imply a true separation under the APSP Conjecture.

Gomory-Hu’s 1961 algorithm remains the only solution for the general case…

Even nondeterministically.
APMF in Subcubic Time

Assuming the APSP Conjecture: APMF < APSP!

GH Tree in $\tilde{O}(n^{2.875})$ time.

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>$n^{2+7/8}$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>General</td>
<td>$n^{2+7/8}$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^2$</td>
<td>$n^\omega$</td>
</tr>
</tbody>
</table>

So maybe the APSP Conjecture is false...?
APMF in Quadratic Time!

21 days later...

[A.- Krauthgamer - Li - Panigrahi - Saranurak - Trabelsi (v2), and independently Zhang’21]:

**GH Tree** in $\tilde{O}(n^2)$ time.

<table>
<thead>
<tr>
<th></th>
<th>APMF</th>
<th>APSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>$n^2$</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Unweighted</td>
<td>$n^2$</td>
<td>$n^{\omega}$</td>
</tr>
</tbody>
</table>

**Assuming (a very weak) APSP Conjecture:** APMF < APSP!
Gomory-Hu Tree in Max-Flow Time?

[A.- Krauthgamer - Li - Panigrahi - Saranurak - Trabelsi (v2), and independently Zhang’21]:

GH Tree in time:
- $\tilde{O}(n^2) + \tilde{O}(\text{Max-Flow}(n, m))$ for weighted graphs,
- $m^{1+o(1)} + \tilde{O}(\text{Max-Flow}(n, m))$ for unweighted graphs.

We can compute a succinct all-pairs max-flow oracle in the time to compute a single-pair max-flow!
Technical Overview

How to construct a GH Tree in subcubic time

**GH Tree** Construction

- **Simple graphs**
  - Expander Decomposition
    - $\tilde{O}(1)$ calls “Isolating Cuts”
      - $\tilde{O}(1)$ calls to Max-Flow

- **General graphs**
  - Steiner Tree Packing
    - $\tilde{O}(1)$ calls to “Isolating Cuts”
      - $\tilde{O}(1)$ calls to Max-Flow
Let’s start with the GH algorithm and optimize it with single-source cuts.
The Gomory-Hu Algorithm

Max-Flow(s, t) = 11
Lemma [Gomory-Hu]:
The min s,t-cut in the contracted graph is a min s,t-cut in the original graph.
The Gomory-Hu Algorithm

Always makes \( n-1 \) calls to Max-Flow, but the graphs get smaller...

Worst case: Recursion depth \( \Omega(n) \).

\[ \Omega(n) \cdot MF(n) \]

Speedup idea: Use single-source cuts, from a random pivot.
Technical Overview

**GH Tree Construction**

- Single-Source Min Cuts
  - Simple graphs
    - Expander Decomposition
      - $\tilde{O}(1)$ calls “Isolating Cuts”
        - $\tilde{O}(1)$ calls to Max-Flow
  - General graphs
    - Steiner Tree Packing
      - $\tilde{O}(1)$ calls to “Isolating Cuts”
        - $\tilde{O}(1)$ calls to Max-Flow

Structural analysis of the graph
The Isolating Cuts Procedure

Discovered independently by [LP’20, AKT’21]

**GH Tree** Construction

![Diagram with steps and annotations]

- **Simple graphs**
  - Expander Decomposition
    - $\tilde{O}(1)$ calls "Isolating Cuts"
    - $\tilde{O}(1)$ calls to Max-Flow

- **General graphs**
  - Structural analysis of the graph
  - Steiner Tree Packing
    - $\tilde{O}(1)$ calls to "Isolating Cuts"
    - $\tilde{O}(1)$ calls to Max-Flow
The Isolating Cuts Procedure

Discovered independently by [LP’20, AKT’21]
and quickly found many applications
[LP21, CQ21, MN21, LNPSY21, AKT21a, LPS21, Zha21, AKT22, CLP22]

Given a set of terminals $U = \{u_1, \ldots, u_k\}$
return $\forall u_i \in U : \min (u_i, U\setminus\{u_i\})$-cut
The Isolating Cuts Procedure

Discovered independently by [LP’20, AKT’21] and quickly found many applications [LP21, CQ21, MN21, LNPSY21, AKT21a, LPS21, Zha21, AKT22, CLP22]

Given a set of terminals $U = \{u_1, \ldots, u_k\}$, return $\forall u_i \in U : \min (u_i, U \setminus \{u_i\})$-cut “Isolating cuts”

Only takes $\tilde{O}(MF(n, m))$ time.

Not $k \cdot MF(n, m)$...
SS Min-Cuts via Isolating Cuts

A hard case: suppose that this is the GHT.

Challenge: Find the triples (well-connected triangles).

Reminds of the max-triangle problem (APSP-equivalent)...

Solution: Use Isolating Cuts with random terminals.
SS Min-Cuts via Isolating Cuts

A hard case: suppose that this is the GHT.

Challenge: Find the triples (well-connected triangles).

Solution: Use Isolating Cuts with random terminals.
**SS Min-Cuts via Isolating Cuts**

A hard case: suppose that this is the GHT.

**Challenge:** Find the triples (well-connected triangles).

*Works well for finding unbalanced cuts...*

**Solution:** Use Isolating Cuts with random terminals.
SS Min-Cuts via Isolating Cuts

The hardest case: suppose that this is the GHT.

One non-trivial large cut; how to find it?

Challenge: The min \((c_r, v)\)-cuts tell us nothing about \(c_l\).

Solution: Use a structural analysis of the graph as a guide.

Figure by Ohad Trabelsi.
Technical Overview

GH Tree Construction

Single-Source Min Cuts

Simple graphs

Expander Decomposition

\( \tilde{O}(1) \) calls “Isolating Cuts”

\( \tilde{O}(1) \) calls to Max-Flow

General graphs

Structural analysis of the graph

Steiner Tree Packing

\( \tilde{O}(1) \) calls to “Isolating Cuts”

\( \tilde{O}(1) \) calls to Max-Flow

Steiner Tree

Packing

Simple graphs

Structural analysis of the graph

GH Tree

Construction

Single-Source Min Cuts

Simple graphs

Expander Decomposition

\( \tilde{O}(1) \) calls “Isolating Cuts”

\( \tilde{O}(1) \) calls to Max-Flow

General graphs

Steiner Tree Packing

\( \tilde{O}(1) \) calls to “Isolating Cuts”

\( \tilde{O}(1) \) calls to Max-Flow

Technical Overview
Expanders-guided querying

**GH Tree Construction**

Single-Source Min Cuts

- **Simple graphs**
  - Expander Decomposition
    - \( \tilde{O}(1) \) calls "Isolating Cuts"
      - \( \tilde{O}(1) \) calls to Max-Flow

- **General graphs**
  - Steiner Tree Packing
    - \( \tilde{O}(1) \) calls to "Isolating Cuts"
      - \( \tilde{O}(1) \) calls to Max-Flow

**Structural analysis of the graph**
Expander-guided querying

An expander decomposition of G:
Expanders-guided querying

An expander decomposition of $G$:

$\tilde{O}(n^{2.5})$ for simple graphs: [AKT STOC’21]

Set: $\phi = 1/\sqrt{n}$

1. In any expander:
   
   $\leq \sqrt{n}$ nodes from one side

2. $\leq n^{1.5}$ edges outside expanders.

$O(1)$ expanders with $\Omega(n)$ nodes.

*Simple graphs only:*

$\leq \sqrt{n}$ nodes in small expanders.

Figure by Ohad Trabelsi.
An expander decomposition of $G$:

\[ \tilde{O}(n^{2.5}) \text{ for simple graphs: } [AKT \text{ STOC'21}] \]

Set: $\phi = 1/\sqrt{n}$

1. In any expander:
   \[ \leq \sqrt{n} \text{ nodes from one side} \]

2. \[ \leq n^{1.5} \text{ edges outside expanders.} \]

\[ O(1) \text{ expanders with } \Omega(n) \text{ nodes.} \]

*Simple graphs only:*

\[ \leq \sqrt{n} \text{ nodes in small expanders.} \]

For each large expander:
Use Isolating Cuts with random terminals.

Figure by Ohad Trabelsi.
An expander decomposition of $G$:

$\tilde{O}(n^{2.5})$ for simple graphs: [AKT STOC'21]

Set: $\phi = 1/\sqrt{n}$

1. In any expander:
   $\leq \sqrt{n}$ nodes from one side

2. $\leq n^{1.5}$ edges outside expanders.

$O(1)$ expanders with $\Omega(n)$ nodes.

Simple graphs only:

$\leq \sqrt{n}$ nodes in small expanders.

For each large expander:
Use Isolating Cuts with random terminals.

Repeat $\sqrt{n}$ times.
Technical Overview

**GH Tree Construction**

- **Simple graphs**
  - Expander Decomposition
    - $\tilde{O}(1)$ calls “Isolating Cuts”
      - $\tilde{O}(1)$ calls to Max-Flow

- General graphs
  - Structural analysis of the graph
  - Steiner Tree Packing
    - $\tilde{O}(1)$ calls to “Isolating Cuts”
      - $\tilde{O}(1)$ calls to Max-Flow
Technical Overview

GH Tree Construction

Single-Source Min Cuts

Simple graphs

Expander Decomposition

\(\tilde{O}(1)\) calls “Isolating Cuts”

\(\tilde{O}(1)\) calls to Max-Flow

General graphs

Structural analysis of the graph

\(\tilde{O}(1)\) calls to “Isolating Cuts”

Steiner Tree Packing

\(\tilde{O}(1)\) calls to Max-Flow

Simple graphs

General graphs

Steiner Tree Packing

\(\tilde{O}(1)\) calls to Max-Flow
We generalize both techniques to single-source min-cuts...

Main challenge: existence of low degree nodes.
Main challenge: existence of low degree nodes.
Thm [Nash-Williams and Tutte 1961]: There are $\lambda_{\text{min}}/2$ disjoint spanning trees.

Karger’s Algorithm:
1. Pack $\sim \lambda_{\text{min}}/2$ spanning trees.
2. Pick one at random, it is 2-respecting.
3. Solve min-cut with 2-respecting tree.
From Tree Packings to k-Respecting Trees

Thm [Nash-Williams and Tutte 1961]: There are $\lambda_{min}/2$ disjoint spanning trees.

Karger’s Algorithm:
1. Pack $\sim \lambda_{min}/2$ spanning trees.
2. Pick one at random, it is 2-respecting.
3. Solve min-cut with 2-respecting tree.

2-respecting spanning tree:
Up to 2 tree edges cut by min-cut.

Low degree nodes prevent us from packing so many spanning trees, but Steiner trees are OK.
Spanning Trees vs. Steiner Trees

2-respecting spanning tree [Karger’96]:
≤ 2 tree edges cut by global min-cut.

2-respecting Steiner tree [AKLPST’21]:
≤ 2 tree edges cut by SS min-cuts.

Actually, 4-respecing…
Spanning Trees vs. Steiner Trees

2-respecting spanning tree [Karger’96]:
≤ 2 tree edges cut by global min-cut.

Cut fully determined by the 2 tree edges

2-respecting Steiner tree [AKLPST’21]:
≤ 2 tree edges cut by SS min-cuts.

Exponentially many options given the tree edges
SS Cuts via 4-respecting Steiner Trees

Our Algorithm:

1. There are $\lambda_{\text{Steiner}}/2$ disjoint Steiner trees.
2. Pack $\sim \lambda_{\text{Steiner}}/4$ Steiner trees, via MWU and Mehlhorn’s 2-approximate Min-Steiner Tree Alg.
3. Sample a 4-respecting Steiner Tree.
4. Solve SS-Min-Cuts using 4-respecting Steiner tree, via Isolating Cuts.

Thm [Nash-Williams and Tutte 1961]:
There are $\lambda_{\text{min}}/2$ disjoint spanning trees.

Karger’s Algorithm:
1. Pack $\sim \lambda_{\text{min}}/2$ spanning trees.
2. Pick one at random, it is 2-respecting.
3. Solve min-cut with 2-respecting tree.
Open Questions

1. Refute the APSP Conjecture, or prove it.  \textit{e.g. under SETH/3SUM.}
(min,+) - convolution

Coin Change
Knapsack (Small Weights)
...

Planar graph problems
...

Dynamic graph problems
...

APSP

Radius
Negative Triangle
Stochastic Context-Free Grammar Parsing
Replacement Paths
Median
...

Max-Weight-k-Clique

Viterbi
Tree Edit Distance
Max Rectangle
Min Weight Cycle
...

Max-Weight-k-Clique

Viterbi
Tree Edit Distance
Max Rectangle
Min Weight Cycle
...
Open Questions

1. Refute the APSP Conjecture, or prove it. \textit{e.g. under SETH/3SUM.}

2. More applications of the techniques (Steiner Tree Packing, Isolating Cuts, Expander Decompositions with vertex demands)
   \textit{e.g. for }k\text{-cuts, vertex cuts, directed cuts, hypergraphs, etc.}

3. \textbf{GH Tree} in $\tilde{O}(\text{Max-Flow-Time})$ for general graphs too?

4. Subcubic time deterministically for general graphs?

\textbf{Thanks for your attention!}