Shadow removal

Rajesh Chitnis, Marek Cygan,
Mohammadtaghi Hajiaghayi, Dániel Marx

April 9, 2013
Warsaw, Poland

Randomized sampling of important separators

A recent technique used by several results:

- Multicut [Marx and Razgon STOC 2011]
- Clustering problems [Lokshtanov and Marx ICALP 2011]
- Directed Multiway Cut [Chitnis, Hajiaghayi, Marx SODA 2012]
- Directed Multicut in DAGs [Kratsch, Pilipczuk, Pilipczuk, Wahlström ICALP 2012]
- Directed Subset Feedback Vertex Set [Chitnis, C., Hajiaghayi, Marx ICALP 2012]
- Parity Multiway Cut [Lokshtanov, Ramanujan ICALP 2012]

Goals

After this talk I would like you to:

1. understand the statement of shadow removal theorem,
2. have a feeling how random sampling of important separators is used to prove it.

Goals

After this talk I would like you to:

1. understand the statement of shadow removal theorem,
2. have a feeling how random sampling of important separators is used to prove it.

Two toy problems: (p, q)-CLUSTERING, DIRECTED MULTIWAY CUT.

Clustering

We want to partition objects into clusters subject to certain requirements (typically: related objects are clustered together, bounds on the number or size of the clusters etc.)

(p, q)-CLUSTERING

Input: A graph G, integers p, q.

Find: A partition $(V_1, \ldots, V_m)$ of $V(G)$ such that for every i

- $|V_i| \leq p$ and
- $d(V_i) \leq q$.

$d(V_i)$: number of edges leaving V_i.

Theorem [Lokshtanov and Marx 2011]

(p, q)-CLUSTERING can be solved in time $2^{O(q)} \cdot n^{O(1)}$.

A sufficient and necessary condition

Good cluster: size at most p and at most q edges leaving it.

Necessary condition:

Every vertex is contained in a good cluster.

A sufficient and necessary condition

Good cluster: size at most p and at most q edges leaving it.

Necessary condition:

Every vertex is contained in a good cluster.

But surprisingly, this is also a **sufficient condition!**

Lemma

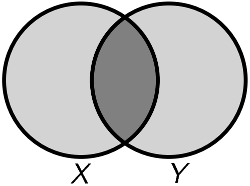
Graph G has a (p, q)-clustering if and only if every vertex is in a good cluster.

A sufficient and necessary condition

Lemma

Graph G has a (p, q)-clustering if and only if every vertex is in a good cluster.

Proof: Find a collection of good clusters covering every vertex and having minimum total size. Suppose two clusters intersect.

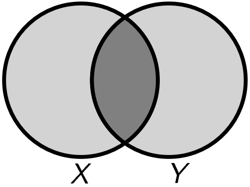


A sufficient and necessary condition

Lemma

Graph G has a (p, q)-clustering if and only if every vertex is in a good cluster.

Proof: Find a collection of good clusters covering every vertex and having minimum total size. Suppose two clusters intersect.



$$d(X) + d(Y) \geq d(X \setminus Y) + d(Y \setminus X)$$

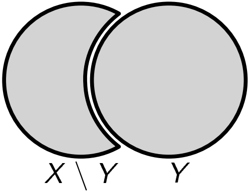
$\Rightarrow$ either $d(X) \geq d(X \setminus Y)$ or $d(Y) \geq d(Y \setminus X)$ holds.

A sufficient and necessary condition

Lemma

Graph G has a (p,q)-clustering if and only if every vertex is in a good cluster.

Proof: Find a collection of good clusters covering every vertex and having minimum total size. Suppose two clusters intersect.



$$d(X) + d(Y) \geq d(X \setminus Y) + d(Y \setminus X)$$

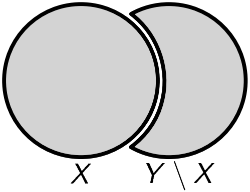
If $d(X) \geq d(X \setminus Y)$, replace X with $X \setminus Y$,
strictly decreasing the total size of the clusters.

A sufficient and necessary condition

Lemma

Graph G has a (p, q)-clustering if and only if every vertex is in a good cluster.

Proof: Find a collection of good clusters covering every vertex and having minimum total size. Suppose two clusters intersect.



$$d(X) + d(Y) \geq d(X \setminus Y) + d(Y \setminus X)$$

If $d(Y) \geq d(Y \setminus X)$, replace Y with $Y \setminus X$,
strictly decreasing the total size of the clusters. QED ■

Finding a good cluster

We have seen:

Lemma

Graph G has a (p, q)-clustering if and only if every vertex is in a good cluster.

All we have to do is to check if a given vertex v is in a good cluster. Trivial to do in time $n^{O(q)}$.

Finding a good cluster

We have seen:

Lemma

Graph G has a (p, q)-clustering if and only if every vertex is in a good cluster.

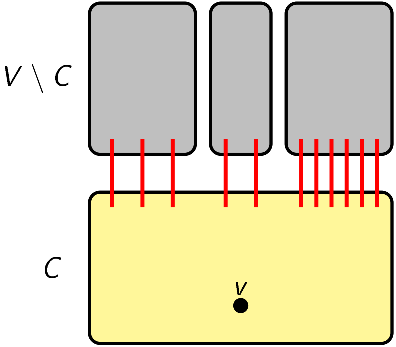
All we have to do is to check if a given vertex v is in a good cluster. Trivial to do in time $n^{O(q)}$.

It is enough to prove:

Lemma

We can check in time $2^{O(q)} \cdot n^{O(1)}$ if v is in a good cluster.

Shadow



Let $C \subseteq V$, $v \in C$ be a a good cluster, i.e. $|C| \leq p$, $|E(C, V \setminus C)| \leq q$.

The shadow of $S = E(C, V \setminus C)$ is $V \setminus C$ (gray vertices, v as a light source).

Random sampling

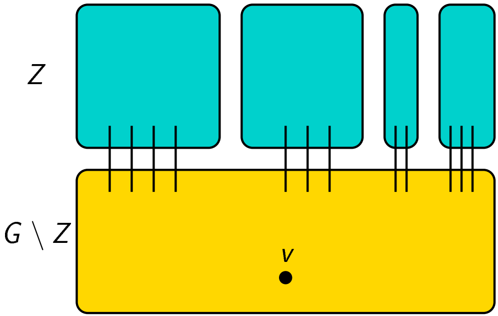
Good cluster: size at most p and at most q edges leaving it.

Theorem

In $2^{O(q)} \cdot n^{O(1)}$ time, we can compute a set $Z \subseteq V$ with the following property. If $v \in V$ is contained in a good cluster, then with probability at least $2^{-O(q)}$ there is good cluster $C \subseteq V$, $v \in C$, such that

- $V \setminus C \subseteq Z$ (the shadow is covered by Z) and
- no edge of $E(C, V \setminus C)$ is contained in Z.

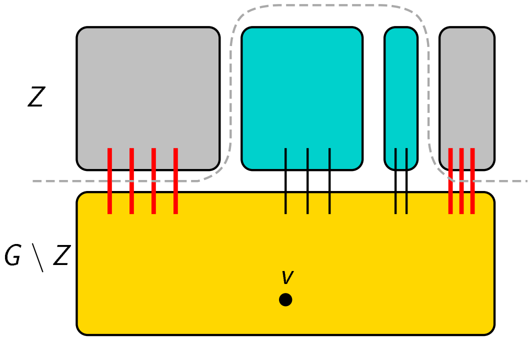
Finding good clusters



- $V \setminus C \subseteq Z$ (the shadow is covered by Z) and
- no edge of $E(C, V \setminus C)$ is contained in Z.

Where are the edges of $E(C, V \setminus C)$? Where is the good cluster?

Finding good clusters

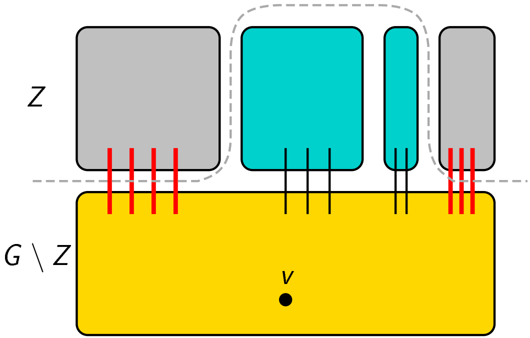


- $V \setminus C \subseteq Z$ (the shadow is covered by Z) and
- no edge of $E(C, V \setminus C)$ is contained in Z.

Where are the edges of $E(C, V \setminus C)$? Where is the good cluster?

Observe: Components of Z are either fully in the cluster or fully outside the cluster. What is this problem?

Finding good clusters



- $V \setminus C \subseteq Z$ (the shadow is covered by Z) and
- no edge of $E(C, V \setminus C)$ is contained in Z.

Where are the edges of $E(C, V \setminus C)$? Where is the good cluster?

Observe: Components of Z are either fully in the cluster or fully outside the cluster. What is this problem?

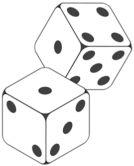
KNAPSACK!

(p, q)-CLUSTERING

(p, q)-CLUSTERING

Random set Z

success probability: $2^{-O(q)}$



KNAPSACK

Polynomial time

Transversal problems

Let G be a graph and let $\mathcal{F}$ be a set of subgraphs in G.

Definition

$\mathcal{F}$-transversal: a set of edges or vertices intersecting each subgraph in $\mathcal{F}$ (i.e., "hitting" or "killing" every object in $\mathcal{F}$).

Classical problems formulated as finding a minimum transversal:

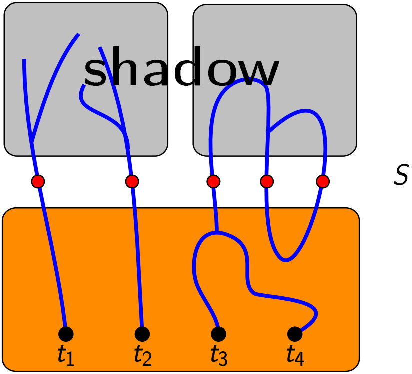
- $s - t$ CUT:
  $\mathcal{F}$ is the set of $s - t$ paths.
- MULTIWAY CUT:
  $\mathcal{F}$ is the set of paths between terminals.
- (DIRECTED) FEEDBACK VERTEX SET:
  $\mathcal{F}$ is the set of (directed) cycles.
- Delete edges/vertices to make the graph bipartite:
  $\mathcal{F}$ is the set of odd cycles.

Randomized sampling of important separators

- MULTICUT [Marx and Razgon STOC 2011]
- Clustering problems [Lokshtanov and Marx ICALP 2011]
- DIRECTED MULTIWAY CUT [Chitnis, Hajiaghayi, Marx SODA 2012]
- DIRECTED MULTICUT in DAGs [Kratsch, Pilipczuk, Pilipczuk, Wahlström ICALP 2012]
- DIRECTED SUBSET FEEDBACK VERTEX SET [Chitnis, C., Hajiaghayi, Marx ICALP 2012]
- PARITY MULTIWAY CUT [Lokshtanov, Ramanujan ICALP 2012]

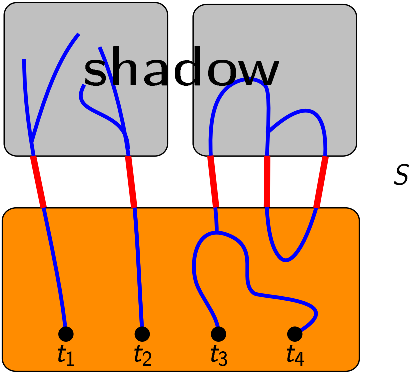
The setting

Let $\mathcal{F}$ be a set of connected (not necessarily disjoint!) subgraphs, each intersecting a set T of vertices.



The **shadow** of an $\mathcal{F}$-transversal S is the set of vertices not reachable from T in $G \setminus S$.

Let $\mathcal{F}$ be a set of connected (not necessarily disjoint!) subgraphs, each intersecting a set T of vertices.



The **shadow** of an $\mathcal{F}$-transversal S is the set of vertices not reachable from T in $G \setminus S$.

The random sampling (undirected edge version)

Shadow: Set of vertices not reachable in $G \setminus S$.

Condition: every $F \in \mathcal{F}$ is connected and intersects T.

Theorem

In $2^{O(k)} \cdot n^{O(1)}$ time, we can compute a set Z with the following property. If there exists an $\mathcal{F}$-transversal of at most k edges, then with probability $2^{-O(k)}$ there is a minimum $\mathcal{F}$-transversal S with

- the shadow of S is covered by Z and
- no edge of S is contained in Z.

Note: The algorithm does not have to know $\mathcal{F}$!

How do we apply it to (p, q)-CLUSTERING?

The random sampling (undirected edge version)

Shadow: Set of vertices not reachable in $G \setminus S$.

Condition: every $F \in \mathcal{F}$ is connected and intersects T.

Theorem

In $2^{O(k)} \cdot n^{O(1)}$ time, we can compute a set Z with the following property. If there exists an $\mathcal{F}$-transversal of at most k edges, then with probability at least $2^{-O(k)}$ there is a minimum $\mathcal{F}$-transversal S with

- the shadow of S is covered by Z and
- no edge of S is contained in Z.

Now:

- $T = \{v\}$, $k = q$,
- $\mathcal{F}$ contains every tree going through v having $> p$ vertices

(Directed) Multiway Cut

Input: Graph G, set of vertices T, integer k
Find: A set S of at most k vertices such that $G \setminus S$ has no (directed) $t_1 - t_2$ path for any $t_1, t_2 \in T$

The undirected version is fairly well understood: best known algorithm solves it in time $2^k \cdot n^{O(1)}$

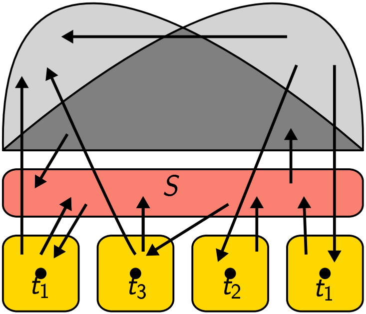
Theorem [Chitnis, Hajiaghayi, Marx 2012]

Directed Multiway Cut is FPT.

Can be formulated as minimum $\mathcal{F}$-transversal, where $\mathcal{F}$ is the set of directed paths between vertices of T.

Directed Multiway Cut

Shadow: those vertices of $G \setminus S$ that cannot be reached from T AND those vertices of $G \setminus S$ from which T cannot be reached.



Shadow removal - Chitnis, C., Hajiaghayi, Marx

Shadow: those vertices of $G \setminus S$ that cannot be reached from T AND those vertices of $G \setminus S$ from which T cannot be reached.

Condition: for every $F \in \mathcal{F}$ and every vertex $v \in F$, there is a $T \to v$ and a $v \to T$ path in F.

Theorem

In $f(k) \cdot n^{O(1)}$ time, we can compute a set Z with the following property. If there exists an $\mathcal{F}$-transversal of at most k vertices, then with probability $2^{-O(k^2)}$ there is a minimum $\mathcal{F}$-transversal S with

- the shadow of S is covered by Z and
- $S \cap Z = \emptyset$.

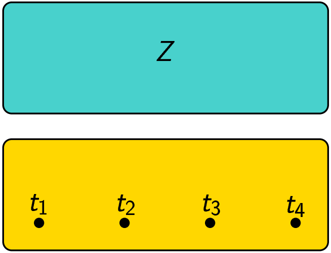
Now:

- T: terminals
- $\mathcal{F}$ contains every directed path between two distinct terminals

Shadow removal

We can assume that Z is disjoint from the solution, so we want to get rid of Z.

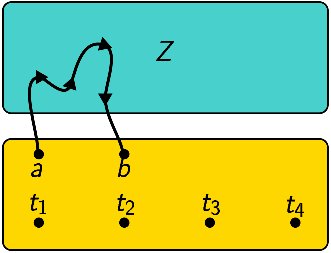
- Deleting Z is not a good idea: can make the problem easier.
- To compensate deleting Z, if there is an $a \to b$ path with internal vertices in Z, add a direct $a \to b$ edge.



Shadow removal

We can assume that Z is disjoint from the solution, so we want to get rid of Z.

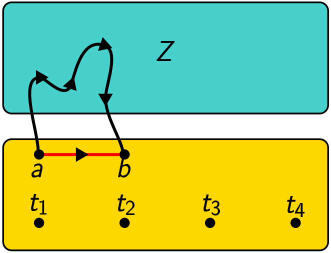
- Deleting Z is not a good idea: can make the problem easier.
- To compensate deleting Z, if there is an $a \to b$ path with internal vertices in Z, add a direct $a \to b$ edge.



Shadow removal

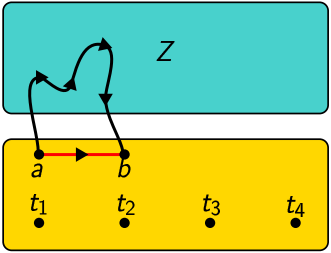
We can assume that Z is disjoint from the solution, so we want to get rid of Z.

- Deleting Z is not a good idea: can make the problem easier.
- To compensate deleting Z, if there is an $a \to b$ path with internal vertices in Z, add a direct $a \to b$ edge.



We can assume that Z is disjoint from the solution, so we want to get rid of Z.

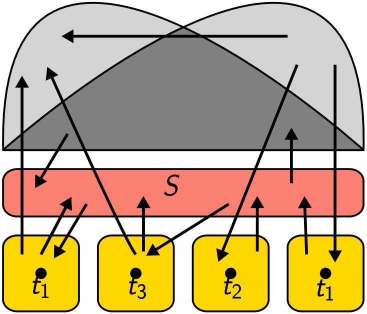
- Deleting Z is not a good idea: can make the problem easier.
- To compensate deleting Z, if there is an $a \to b$ path with internal vertices in Z, add a direct $a \to b$ edge.



Crucial observation:

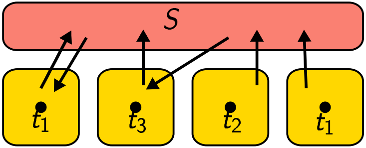
S remains a solution (since Z is disjoint from S) and
S is a **shadowless solution** (since Z covers the shadow of S).

How does a shadowless solution look like?



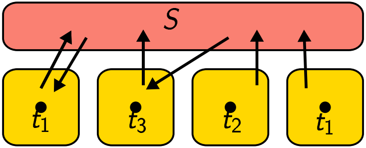
Shadowless solutions

How does a shadowless solution look like?



Shadowless solutions

How does a shadowless solution look like?



It is an undirected multiway cut in the underlying undirected graph!
⇒ **Problem can be reduced to undirected multiway cut.**

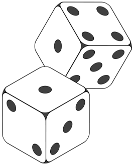
Directed Multiway Cut

DIRECTED MULTIWAY CUT

Random set Z

success probability:

$2^{-O(k^2)}$



UNDIRECTED MULTIWAY CUT

$2^k \cdot n^{O(1)}$ time

Conclusion

Conclusion

The shadow removal lemma for $\mathcal{F}$-transversal unifies the random sampling of important separators step in the known algorithms for: MULTICUT, (p, q)-CLUSTERING, DIRECTED MULTIWAY CUT, DIRECTED MULTICUT in DAGs, DIRECTED SUBSET FEEDBACK VERTEX SET.

Conclusion

Conclusion

The shadow removal lemma for $\mathcal{F}$-transversal unifies the random sampling of important separators step in the known algorithms for: MULTICUT, (p, q)-CLUSTERING, DIRECTED MULTIWAY CUT, DIRECTED MULTICUT in DAGs, DIRECTED SUBSET FEEDBACK VERTEX SET.

Thank you for your attention.
Questions?