Algorithms for instance-stable and perturbation-resilient problems

Haris Angelidakis, TTIC
Konstantin Makarychev, Northwestern
Yury Makarychev, TTIC

Midwest Theory Day, Indiana U., April 16, 2017

Motivation

- Practice: Need to solve clustering and combinatorial optimization problems.
- Theory:
  - Many problems are NP-hard. Cannot solve them exactly.
  - Design approximation algorithms for worst case.

Can we get better algorithms for real-world instances than for worst-case instances?

Motivation

- Answer: Yes!

  When we solve problems that arise in practice, we often get much better approximation than it is theoretically possible for worst case instances.

- Want to design algorithms with provable performance guarantees for solving real-world instances.

Motivation

- Need a model for real-world instances.
- Many different models have been proposed.
- It’s unrealistic that one model will capture all instances that arise in different applications.

This work

- Assumption: instances are stable/perturbation-resilient
- Consider several problems:
  - k-means
  - k-median
  - Multiway Cut
- Get exact polynomial-time algorithms

k-means and k-median

Given a set of points X, distance $d(\cdot,\cdot)$ on X, and k

Partition X into k clusters $C_1, \ldots, C_k$ and find a "center" c_i in each C_i so as to minimize

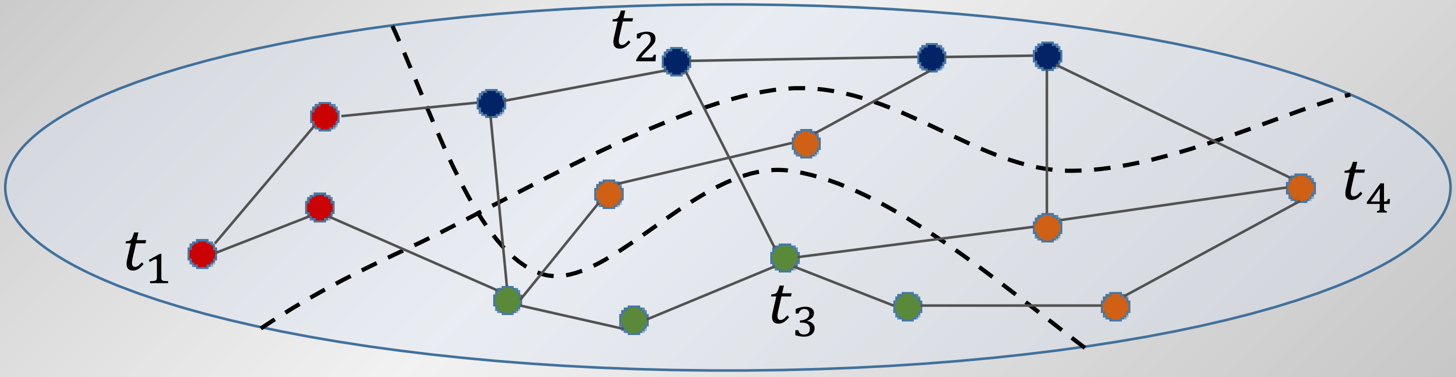
$$\sum_{i=1}^{k} \sum_{u \in C_i} d(u, c_i) \qquad (k\text{-median})$$

$$\sum_{i=1}^{k} \sum_{u \in C_i} d(u, c_i)^2 \qquad (k\text{-means})$$

Multiway Cut

Given

- a graph $G = (V, E, w)$
- a set of terminals $t_1, \ldots, t_k$



Find a partition of V into sets $S_1, \ldots, S_k$ that minimizes the weight of cut edges s.t. $t_i \in S_i$.

Instance-stability & perturbation-resilience

- Consider an instance $\mathcal{I}$ of an optimization or clustering problem.

- $\mathcal{I}'$ is a *γ-perturbation* of $\mathcal{I}$ if it can be obtained from $\mathcal{I}$ by "perturbing the parameters" — multiplying each parameter by a number from 1 to γ.
  - $w(e) \leq w'(e) \leq \gamma \cdot w(e)$
  - $d(u, v) \leq d'(u, v) \leq \gamma \cdot d(u, v)$

Instance-stability & perturbation-resilience

An instance $\mathcal{I}$ of an optimization or clustering problem is *perturbation-resilient/instance-stable* if the optimal solution remains the same when we perturb the instance:

every γ-perturbation $\mathcal{I}'$ has the same optimal solution as $\mathcal{I}$

Instance-stability & perturbation-resilience

Every γ-perturbation $\mathcal{I}'$ has the same optimal solution as $\mathcal{I}$

- In practice, we are interested in solving instances where the optimal solution "stands out" among all solutions [Bilu, Linial]
- Objective function is an approximation to the "true" objective function.
- "Practically interesting instance" $\Rightarrow$ the solution is stable

Results

History

Instance-stability & perturbation-resilience was introduced

- in discrete optimization: by Bilu and Linial ‘10
- in clustering: by Awasthi, Blum, and Sheffet ‘12

Results (clustering)

$\gamma \geq 3$	k-center, k-means, k-median	[Awasthi, Blum, Sheffet '12]
$\gamma \geq 1 + \sqrt{2}$	k-center, k-means, k-median	[Balcan, Liang '13]
$\gamma \geq 2$	sym. /asym. k-center	[Balcan, Haghtalab, White '16]
$\gamma \geq 2$	k-means, k-median	[AMM '17]

Results (optimization)

$\gamma \geq cn$	**Max Cut**	[Bilu, Linial '09]
$\gamma \geq c\sqrt{n}$	**Max Cut**	[Bilu, Daniely, Linial, Saks '13]
$\gamma \geq c\sqrt{\log n}\log\log n$	**Max Cut**	[MMV '13]
$\gamma \geq 4$	**Multiway**	[MMV '13]
$\gamma \geq 2 - 2/k$	**Multiway**	[AMM '17]

Results (optimization)

Our algorithm for Multiway Cut is robust.

- Finds the optimal solution, if the instance is stable.
- Finds an optimal solution or detects that the instance is not stable, otherwise.
- Never outputs an incorrect answer.

Algorithm also solves weakly stable instances.

- Assume that the optimal solution may slightly change when we perturb the instance.

Results for Other Problems

Set Cover, Vertex Cover, Min 2-Horn Deletion

There is no *robust* algorithm for $O(n^{1-\varepsilon})$-stable instances unless $\mathrm{P} = \mathrm{NP}$ [AMM '17].

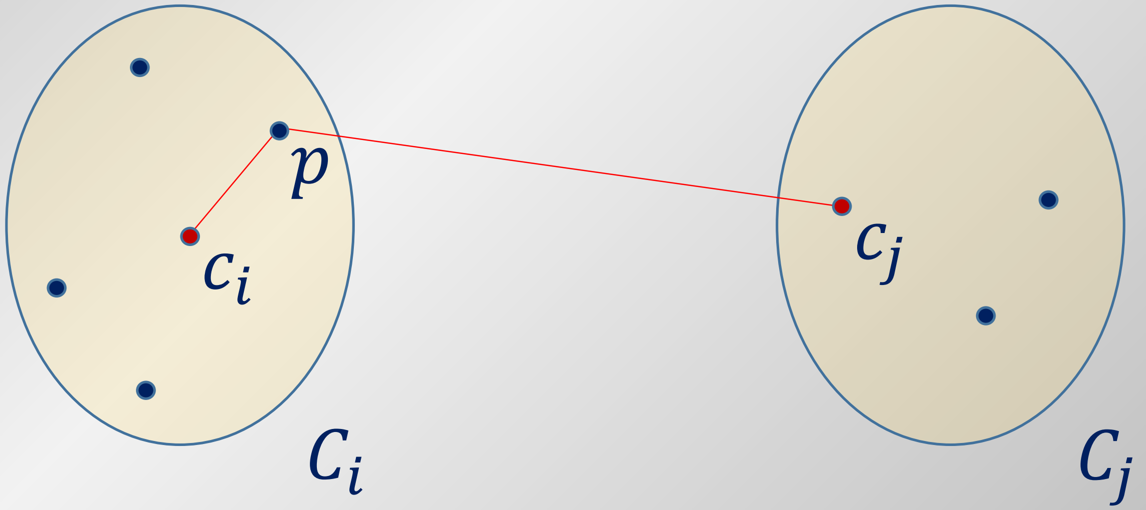
Provide evidence that Multiway cut is hard when $\gamma < \frac{4}{3} - O\left(\frac{1}{k}\right)$.

Algorithm for Clustering Problems

Center Proximity Property

[Awasthi, Blum, Sheffet '12] A clustering $C_1, \ldots, C_k$ with centers $c_1, \ldots, c_k$ satisfies the center proximity property if for every $p \in C_i$:

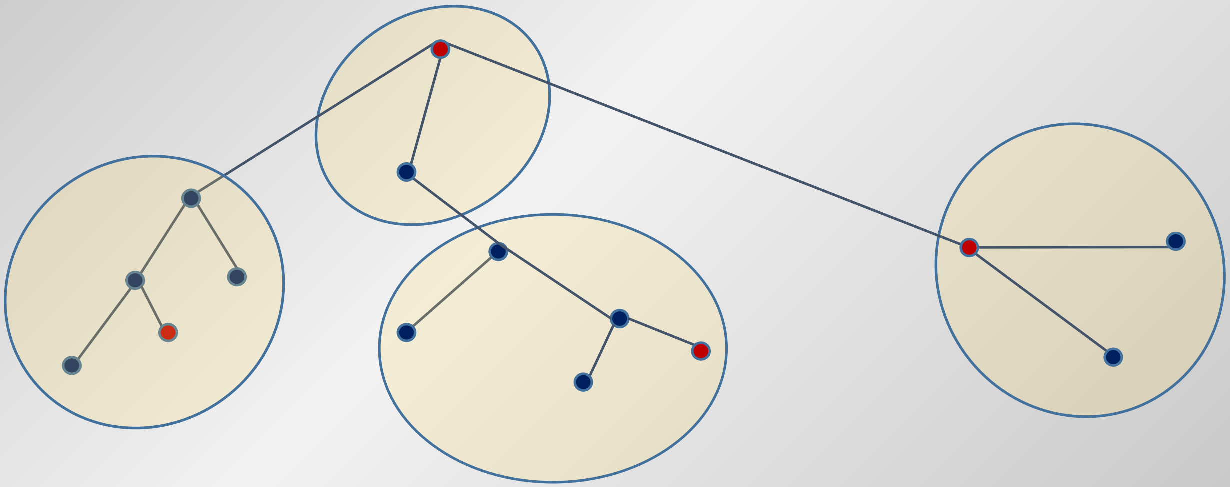
$$d(p, c_j) > \gamma\, d(p, c_i)$$



Plan

i. γ-perturbation resilience ⇒ γ-center proximity

ii. 2-center proximity ⇒ each cluster is a subtree of the MST



iii. use single-linkage + DP to find $C_1, \ldots, C_k$

Perturbation resilience ⇒ center proximity

Perturbation resilience: the optimal clustering doesn't change when we perturb the distances.

$$d(u,v)/\gamma \leq d'(u,v) \leq d(u,v)$$

[ABS '12] $d'(\cdot,\cdot)$ doesn't have to be a metric

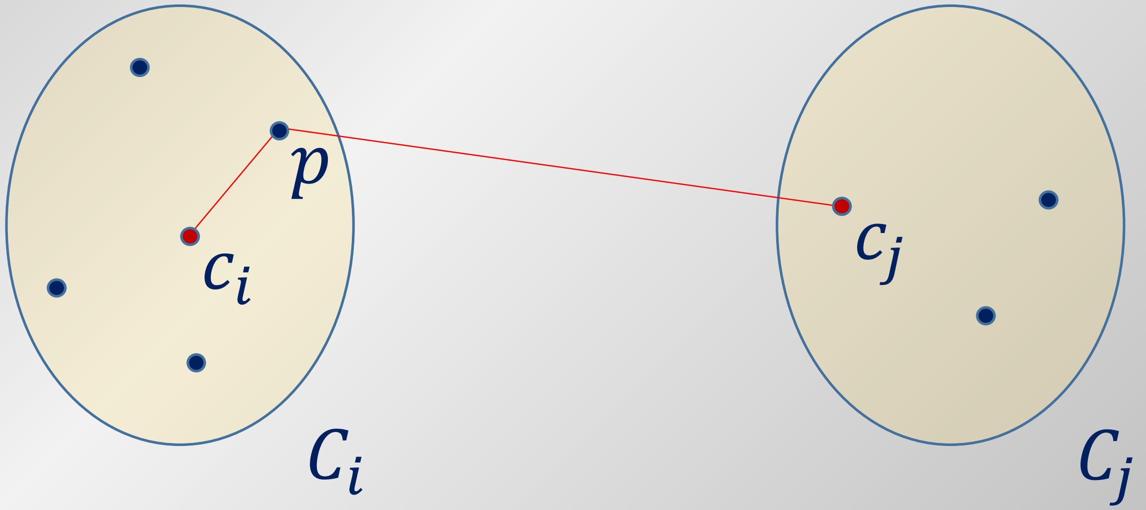
[AMM '17] $d'(\cdot,\cdot)$ is a metric

Metric perturbation resilience is a more natural notion.

Perturbation resilience ⇒ center proximity [ABS '12, AMM '17]

Assume center proximity doesn't hold.

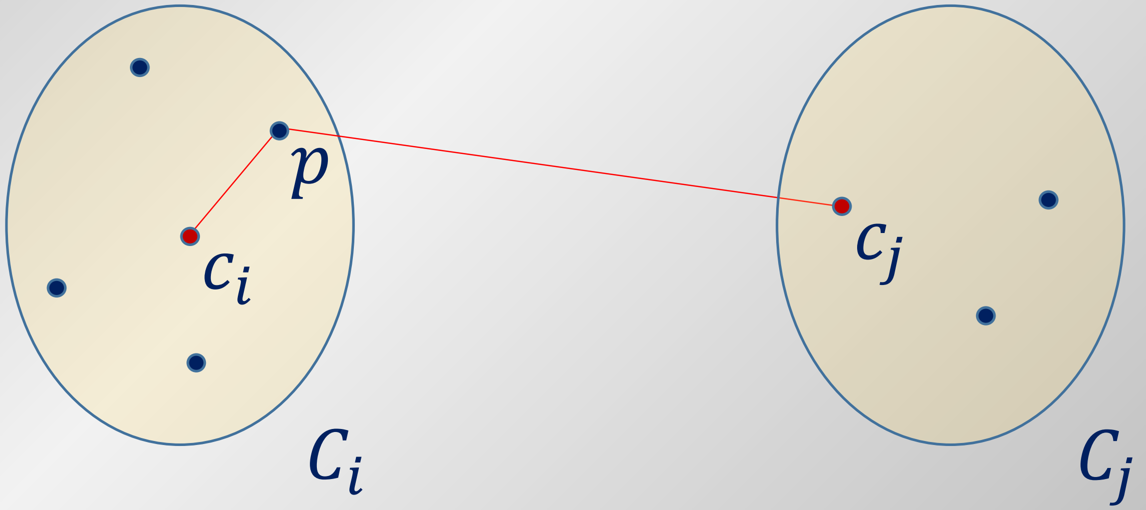
Then $d(p, c_j) \leq \gamma\, d(p, c_i)$.



Perturbation resilience ⇒ center proximity [ABS '12, AMM '17]

Assume center proximity doesn't hold.

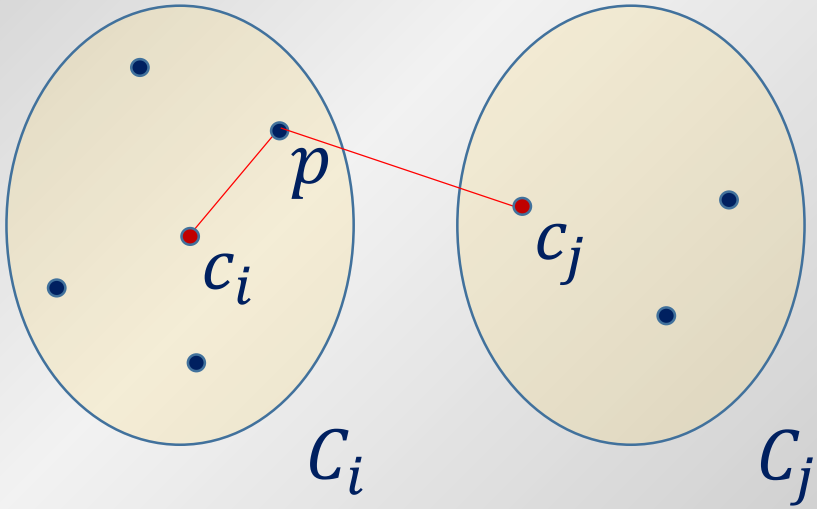
- Let $d'(p, c_j) = d(p, c_i) \geq \gamma^{-1} d(p, c_j)$.
- Don't change other distances.
- Consider the shortest-path closure.



Perturbation resilience ⇒ center proximity [ABS ’12, AMM ‘17]

Assume center proximity doesn’t hold.

- Let $d'(p, c_j) = d(p, c_i) \geq \gamma^{-1} d(p, c_j)$.
- Don’t change other distances.
- Consider the shortest-path closure.

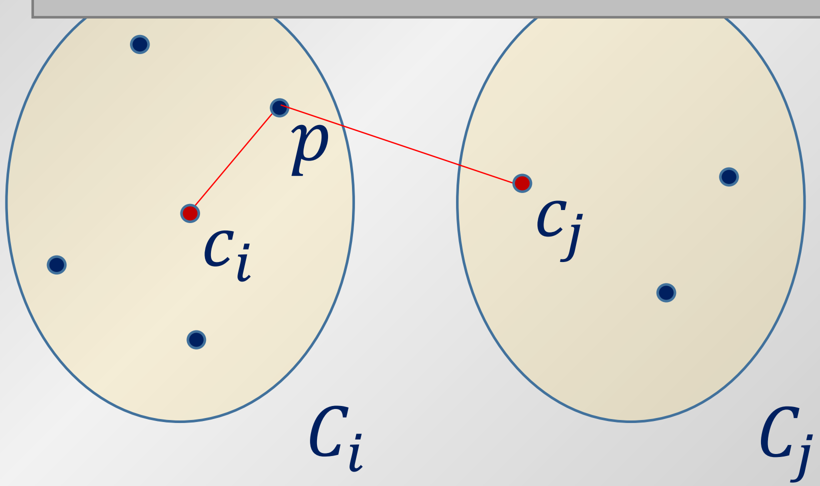


Perturbation resilience ⇒ center proximity [ABS ’12, AMM ‘17]

Assume center proximity doesn’t hold.

- Let $d'(p, c_j) = d(p, c_i) \geq \gamma^{-1} d(p, c_j)$.
- Don’t
- Consi

This is a γ-perturbation.

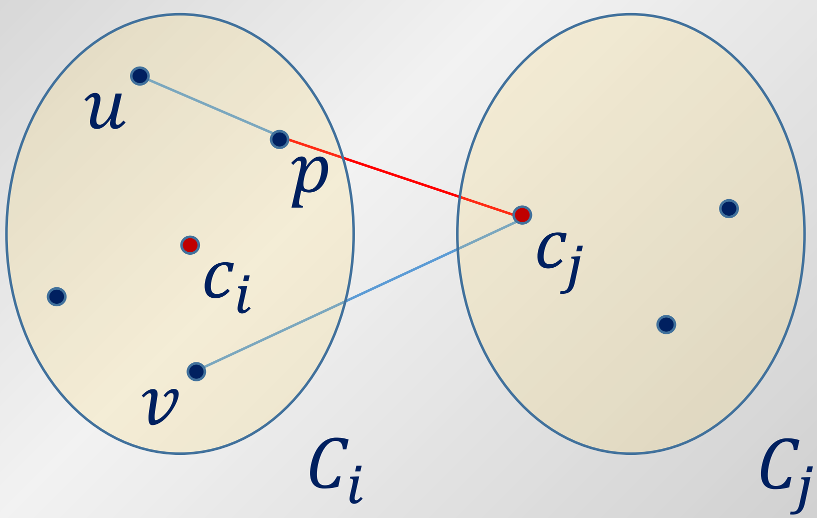


Perturbation resilience ⇒ center proximity [ABS '12, AMM '17]

Distances inside clusters C_i and C_j don't change.

Consider $u, v \in C_i$.

$$d'(u,v) = \min\begin{pmatrix} d(u,v), \\ d(u,p) + d'(p,c_j) + d(c_j,v) \end{pmatrix}$$

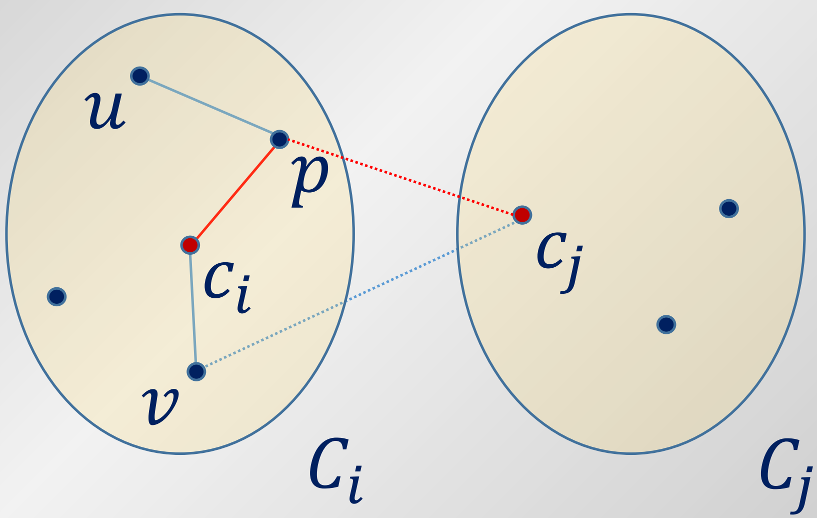


Perturbation resilience ⇒ center proximity [ABS '12, AMM '17]

Distances inside clusters C_i and C_j don't change.

Consider $u, v \in C_i$.

$$d'(u,v) = \min\begin{pmatrix} d(u,v), \\ d(u,p) + d'(p,c_j) + d(c_j,v) \end{pmatrix}$$

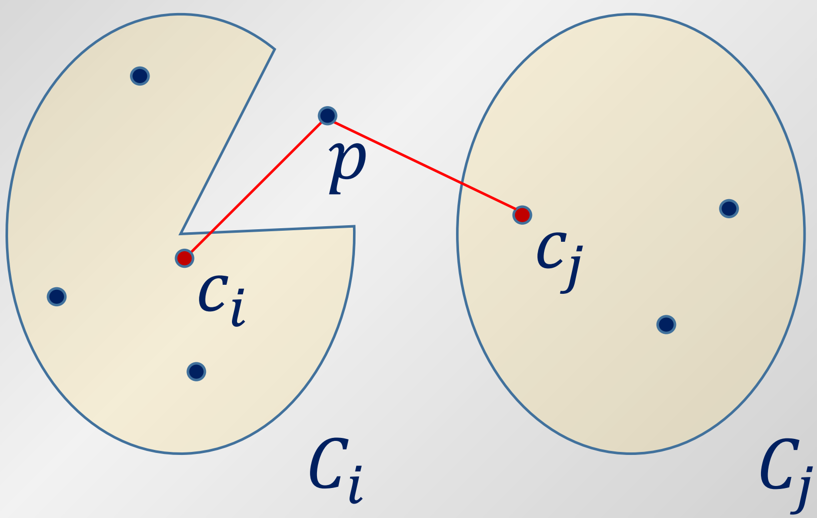


Perturbation resilience ⇒ center proximity [ABS ’12, AMM ‘17]

Since the instance is γ-stable, $C_1, \dots, C_k$ must be the unique optimal solution for distance d'.

Still, c_i and c_j are optimal centers for C_i and C_j.

$d'(p, c_i) = d'(p, c_j)$ ⇒ can move p from C_i to C_j

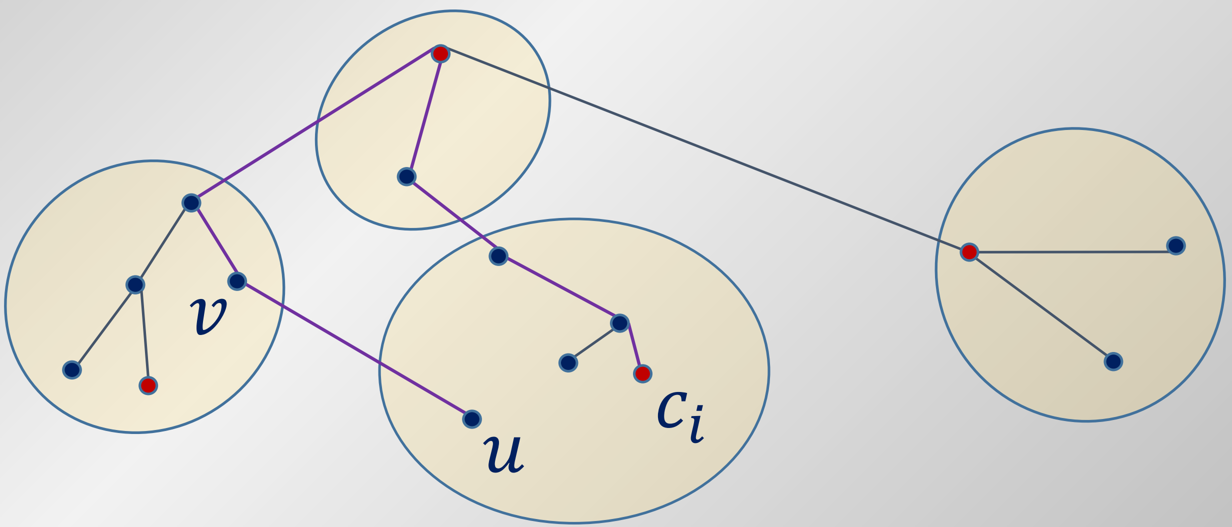


Each cluster is a subtree of MST

[ABS '12] 2-center proximity ⇒

every $u \in C_i$ is closer to c_i than to any $v \notin C_i$

Assume the path from $u \in C_i$ to c_i in MST, leaves C_i.

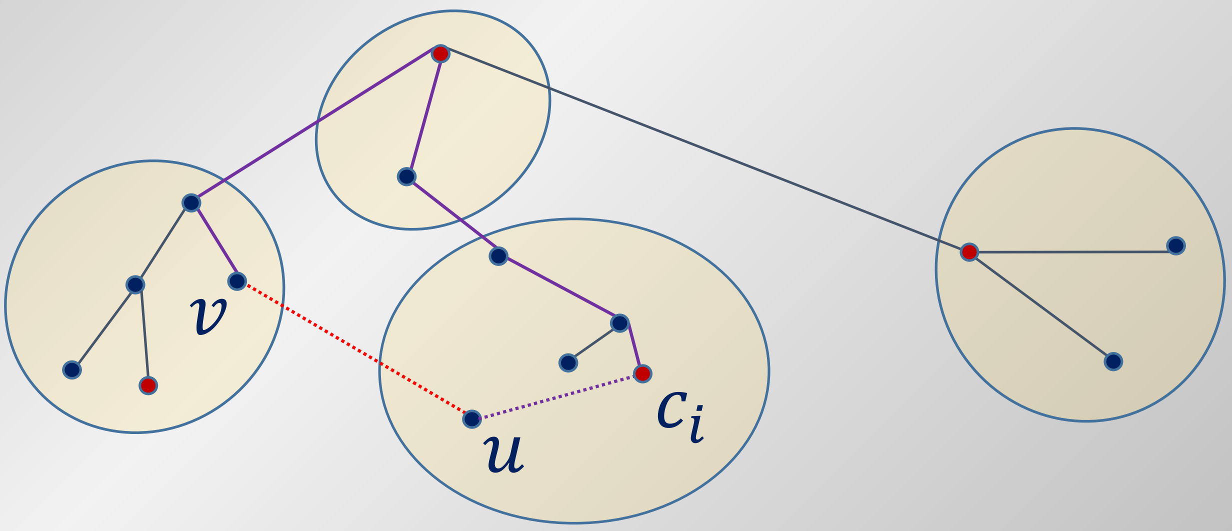


Each cluster is a subtree of MST

[ABS '12] 2-center proximity ⇒

every $u \in C_i$ is closer to c_i than to any $v \notin C_i$

Assume the path from $u \in C_i$ to c_i in MST, leaves C_i.

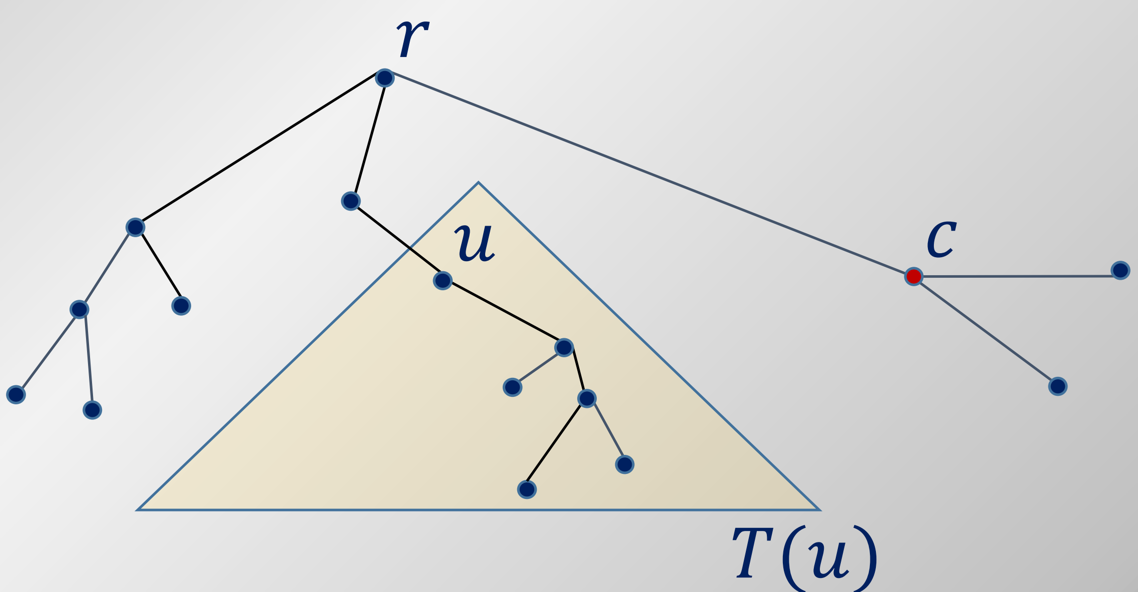


Dynamic programming algorithm

Root MST at some r. $T(u)$ is the subtree rooted at u.

$\text{cost}_u(j, c)$: the cost of the partitioning of $T(u)$

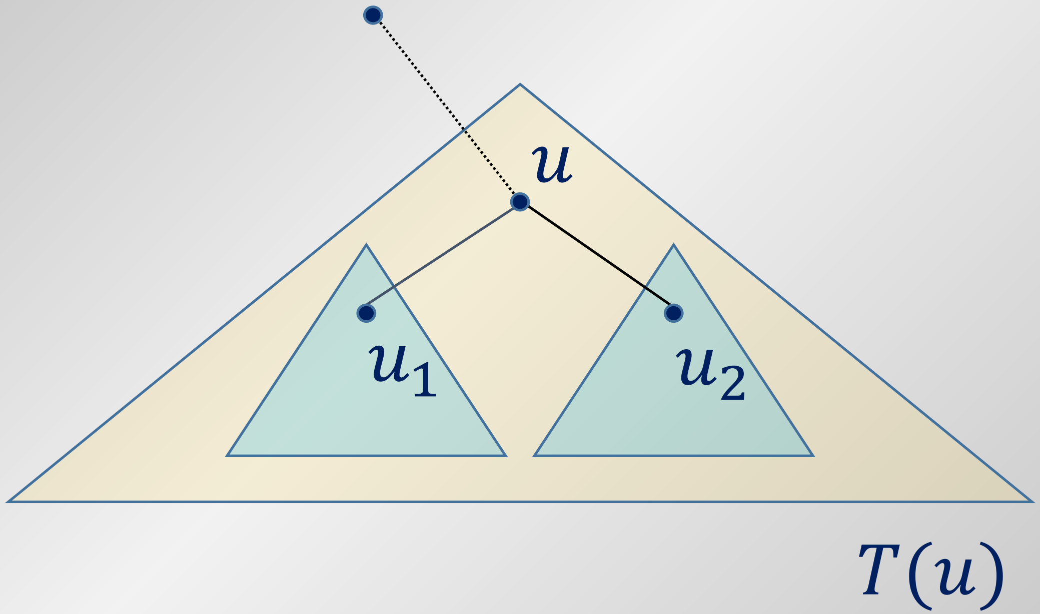
- into j clusters (subtrees)
- so that c is the center of the cluster containing u.



Dynamic programming algorithm

Fill out the DP table bottom-up.

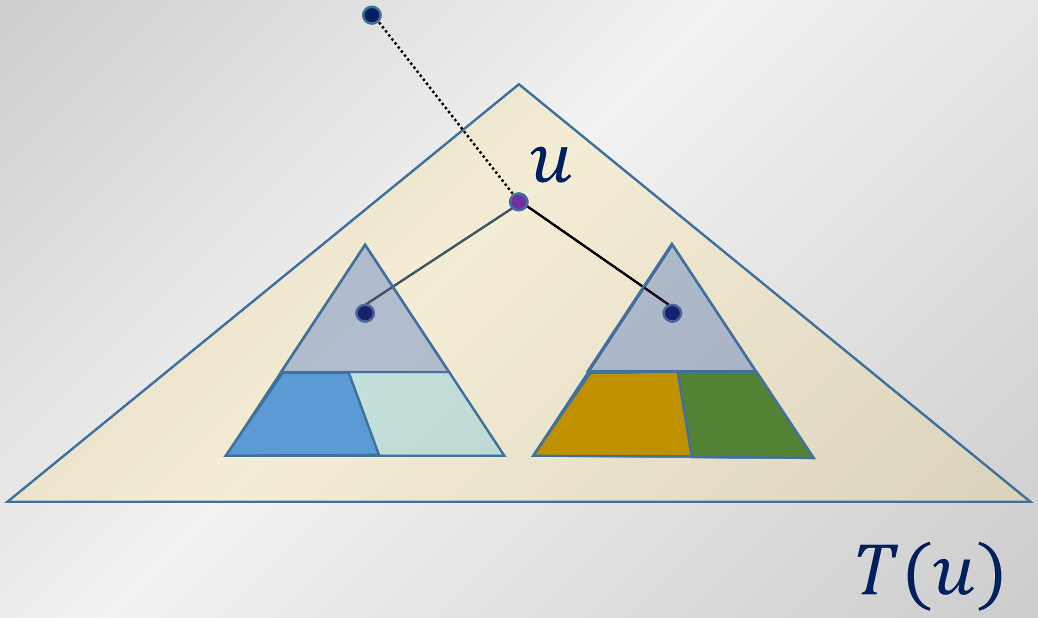
Example: k-median, u has 2 children u_1 and u_2.



Dynamic programming algorithm

Fill out the DP table bottom-up.

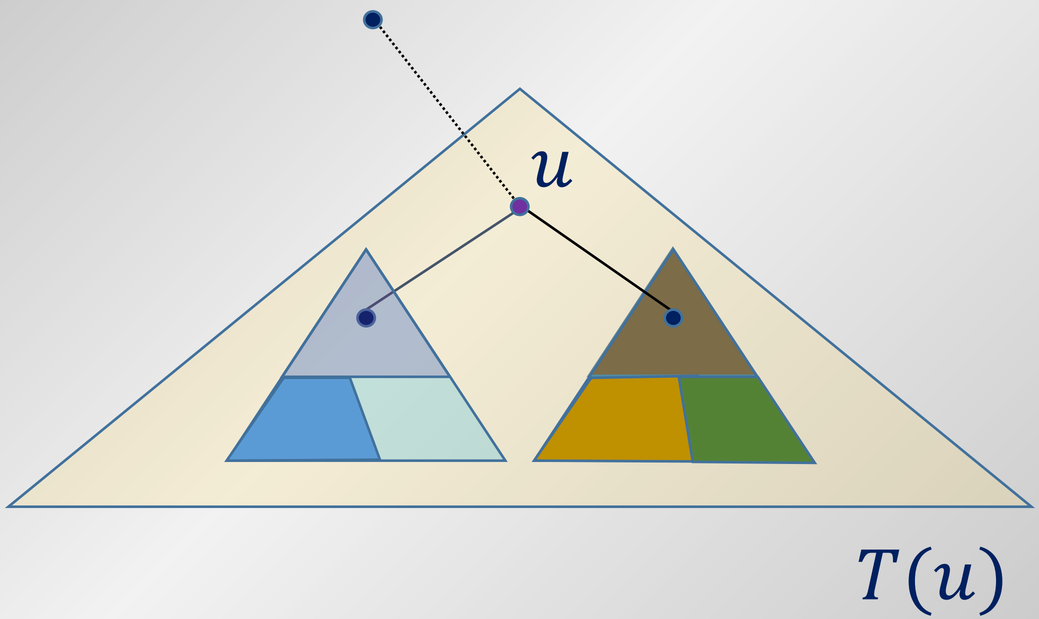
Example: k-median, u has 2 children u_1 and u_2.



Dynamic programming algorithm

Fill out the DP table bottom-up.

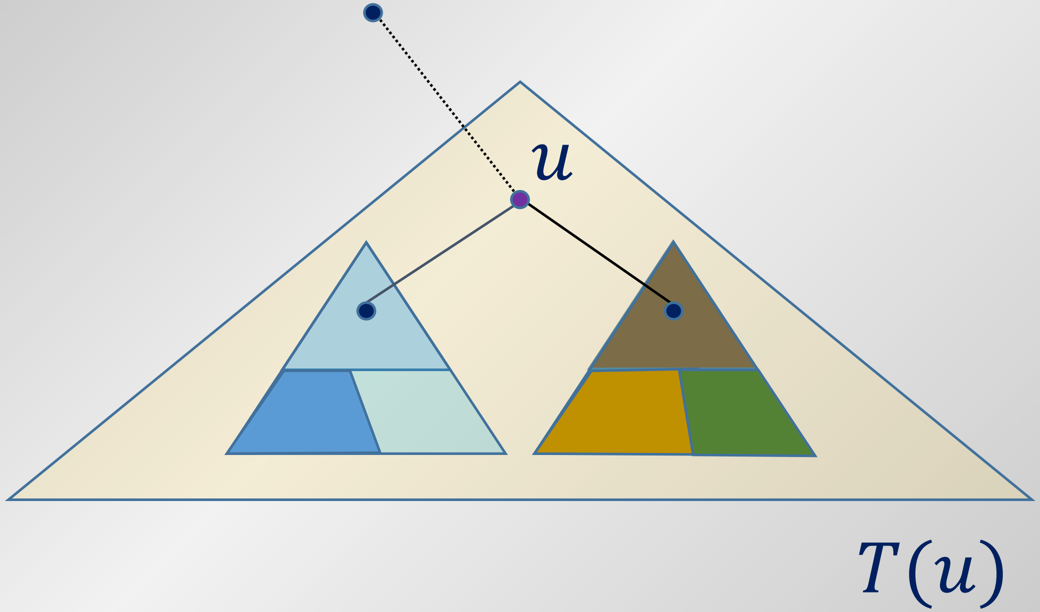
Example: k-median, u has 2 children u_1 and u_2.



Dynamic programming algorithm

Fill out the DP table bottom-up.

Example: k-median, u has 2 children u_1 and u_2.



Dynamic programming algorithm

u, u_1, u_2 lie in the same cluster

$$\text{cost}_u(j, c) = d(u, c) + \text{cost}_{u_1}(j_1, c) + \text{cost}_{u_2}(j_2, c)$$

where $j_1 + j_2 = j + 1$

u, u_1, u_2 lie in different clusters

$$\text{cost}_u(j, c) = d(u, c) + \text{cost}_{u_1}(j_1, c_1) + \text{cost}_{u_2}(j_2, c_2)$$

where $j_1 + j_2 = j - 1$, $c_1 \in T(u_1)$, $c_2 \in T(u_2)$

u, u_1 lie in the same clusters, u_2 in a different

$$\text{cost}_u(j, c) = d(u, c) + \text{cost}_{u_1}(j_1, c) + \text{cost}_{u_2}(j_1, c_2)$$

where $j_1 + j_2 = j$, $c_2 \in T(u_2)$

Hardness results for center-based obejctives

[Balcan, Haghtalab, White ’16] No polynomial-time algorithm for $(2-\varepsilon)$-perturbation-resilient instances of k-center $(NP \neq RP)$.

[Ben-David, Reyzin ‘14] No polynomial-time algorithm for instances of k-means, k-median, k-center satisfying $(2-\varepsilon)$-center proximity property $(P \neq NP)$.

Summary

- Algorithms for 2-perturbation-resilient instances of problems with a natural center based objective: k-means, k-median, facility location

- Algorithms for $\left(2 - \frac{2}{k}\right)$-stable instances of Multiway Cut

- Hardness results for stable instances of Set Cover, Vertex Cover, Min 2-Horn Deletion