Harnessing implicit assumptions in problem formulations:

Approximation-stability and proxy objectives

Avrim Blum

Carnegie Mellon University

Based on work joint with Pranjal Awasthi, Nina Balcan, Anupam Gupta and Or Sheffet

Stanford Workshop on "Beyond Worst-Case Analysis", 2011

Theme of this talk

- Theory tells us many of the problems we most want to solve are (NP-)hard. Even hard to approximate well.

- But that doesn't make the problems go away. And in AI/ML/…, people often find strategies that do well in practice.

- One way to reconcile: distrib assumptions. This talk: make use of properties we often need to hold anyway.

Theme of this talk

- Theory tells us many of the problems we most want to solve are (NP-)hard. Even hard to approximate well.

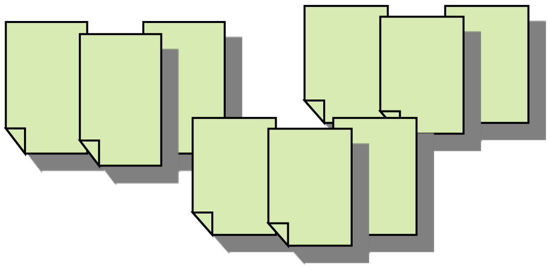
- In particular, often objective is a proxy for some other underlying goal. Implicitly assuming they are related.

- If make this explicit up front, can give alg more to work with, and potentially get around hardness barriers.

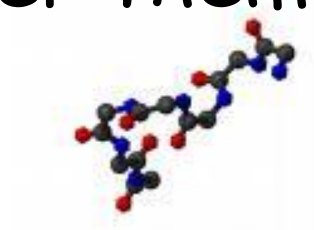
Main running example: Clustering

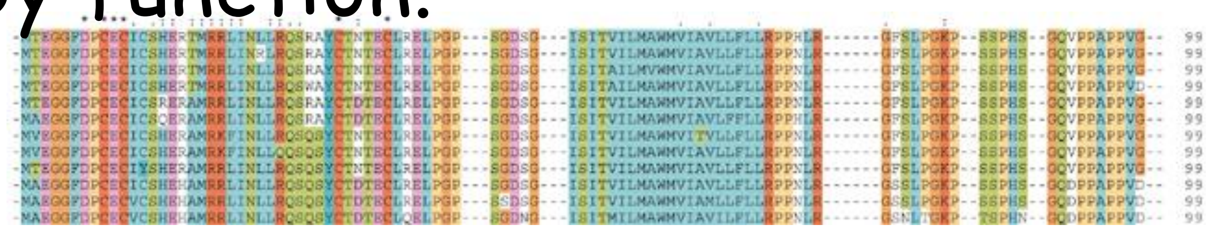
Clustering comes up in many places

- Given a set of documents or search results, cluster them by topic.



- Given a collection of protein sequences, cluster them by function.



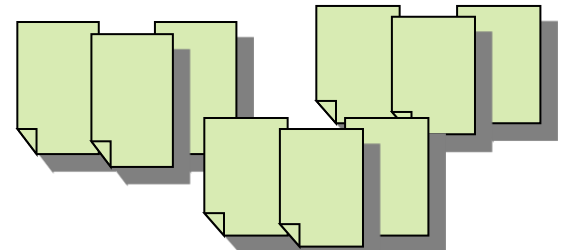


- ...

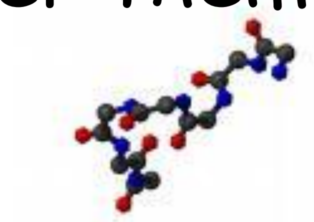
So, how do we solve it?

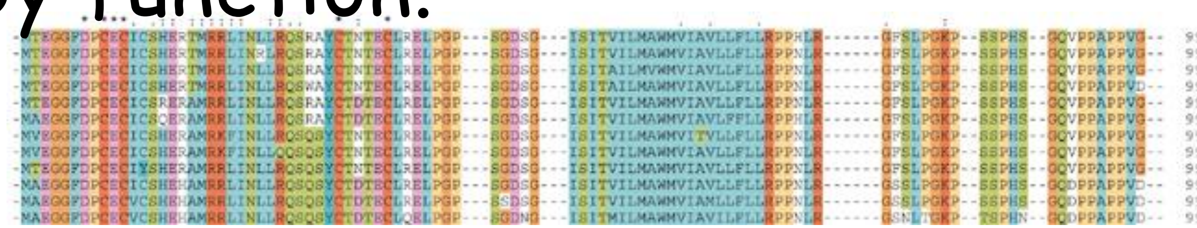
Standard approach

- Given a set of documents or search results, cluster them by topic.



- Given a collection of protein sequences, cluster them by function.



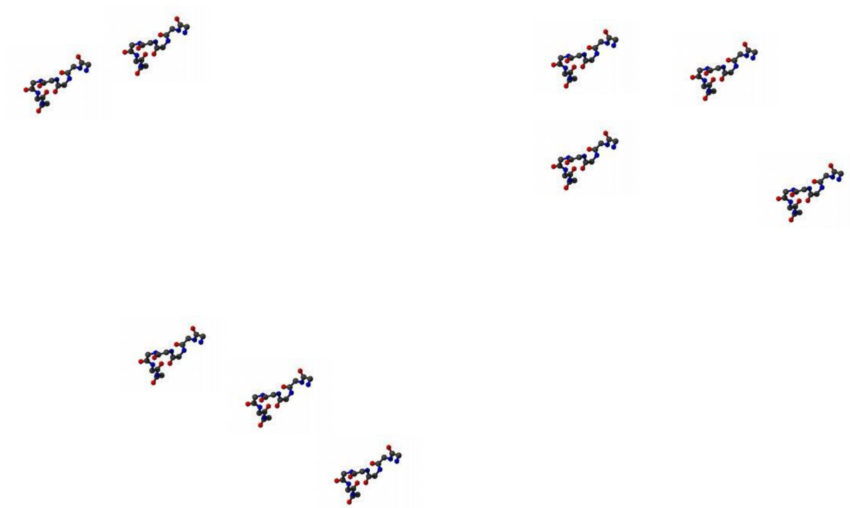


- …

So, how do we solve it?

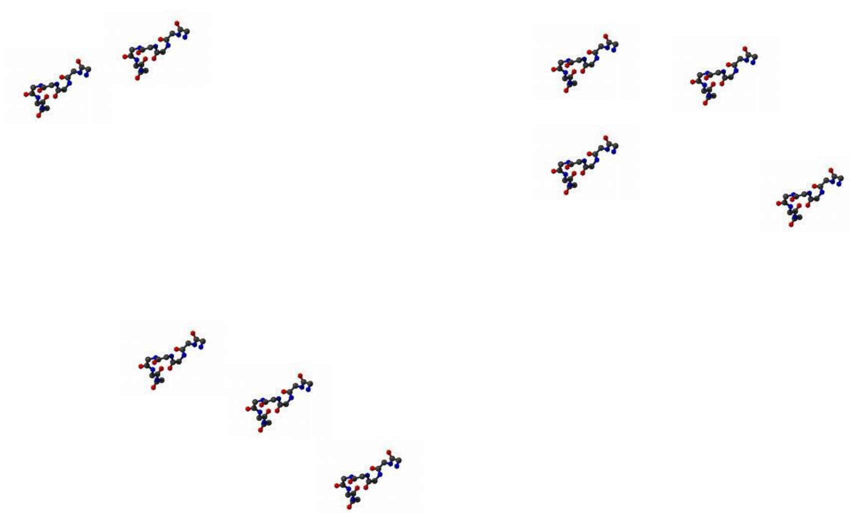
Standard approach

- Come up with some set of features (words in document) or distance measure (edit distance)
- Use to view data as points in metric space
- Run clustering algorithm on points. Hope it gives a good output.



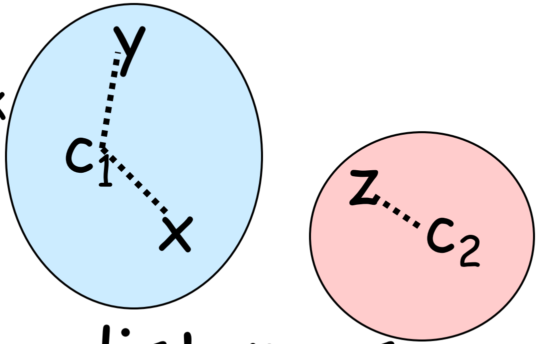
Standard theoretical approach

- Come up with some set of features (words in document) or distance measure (edit distance)
- Use to view data as points in metric space
- Pick some objective to optimize like k-median, k-means, min-sum,…



Standard theoretical approach

- Come up with some set of features (words in document) or distance measure (edit distance)
- Use to view data as points in metric space
- Pick some objective to optimize like k-median, k-means, min-sum,…
  - E.g., k-median asks: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d(x,c_i)$
  - k-means asks: find $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d^2(x,c_i)$
  - Min-sum asks: find k clusters minimizing sum of intra-cluster distances.

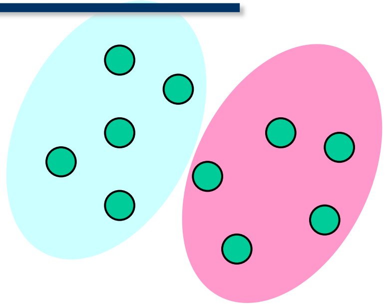


Standard theoretical approach

- Come up with some set of features (words in document) or distance measure (edit distance)
- Use to view data as points in metric space
- Pick some objective to optimize like k-median, k-means, min-sum,...
- Develop algorithm to (approx) optimize this objective. (E.g., best known for k-median is $3+\varepsilon$ approx [AGKMMP04]. k-means is $9+\varepsilon$, min-sum is $(\log n)^{1+\epsilon}$. Beating $1 + 1/e$ is NP-hard [JMS02].)

Can we do better… on the cases where doing better would matter?

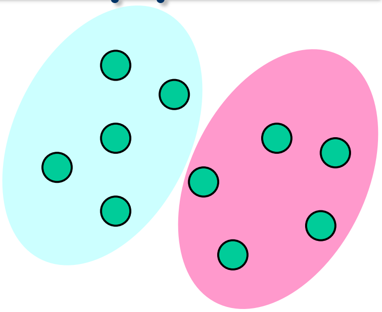
Standard theoretical approach



- Remember, what we really wanted was to cluster proteins by function, etc.
- Objectives like k-median etc. are only a proxy.

Can we do better… on the cases where doing better would matter?

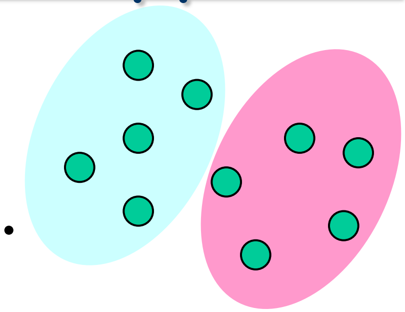
Why do we want to get a c=2 or c=1.1 approx?



- Remember, what we really wanted was to cluster proteins by function, etc.
- Objectives like k-median etc. are only a proxy.

Can we do better… on the cases where doing better would matter?

Why do we want to get a c=2 or c=1.1 approx?

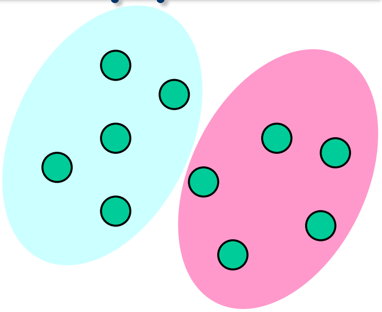


- Remember, what we really wanted was to cluster proteins by function, etc.
- Implicitly hoping that getting c-approx to our objective will allow us to get most points correct.
  - This is an assumption about how the distance measure and objective relate to the clustering we are looking for.
  - What happens if you make it explicit?

Can we do better... on the cases where doing better would matter?

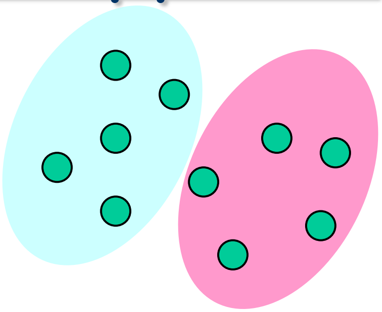
Why do we want to get a c=2 or c=1.1 approx?

- Remember, what we really wanted was to cluster proteins by function, etc.



- Assume: all c-approximations are ε-close (as clusterings) to desired target. I.e., getting c-approx to objective implies getting ε-error wrt real goal.
- Question: does this buy you anything?
- Answer: Yes (for clustering with k-median, k-means, or min-sum objectives)
  - For any constant c>1, can use to get $O(\epsilon)$-close to target. Even though getting a c-apx may be NP-hard (for min-sum, needed large clusters. Improved by [Balcan-Braverman])
  - For k-means, k-median, can actually get c-apx (and therefore, ϵ-close), if cluster sizes > ϵn.

Why do we want to get a c=2 or c=1.1 approx?

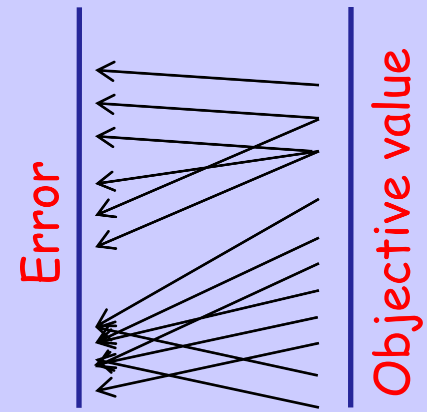


- Remember, what we really wanted was to cluster proteins by function, etc.
- Assume: all c-approximations are ε-close (as clusterings) to desired target. I.e., getting c-approx to objective implies getting ε-error wrt real goal.
- Question: does this buy you anything?
- Answer: Yes (for clustering with k-median, k-means, or min-sum objectives)

More generally: have one objective you can measure, and a different one you care about.

Implicitly assuming they are related.

Let's make it explicit. See if we can use properties it implies.



Approximation-stability

- Instance is (c,ε)-apx-stable for objective Φ: any c-approximation to Φ has error $\leq \varepsilon$.
  - "error" is in terms of distance in solution space. For clustering, we use the fraction of points you would have to reassign to match target.

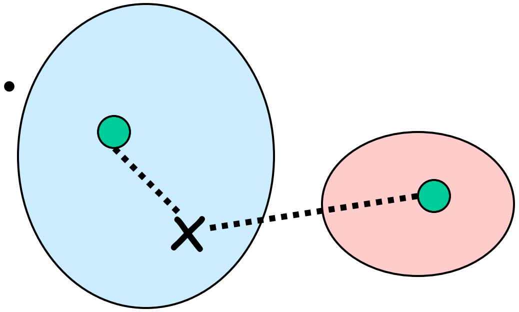
How are we going to use this to cluster well if we don't know how to get a c-approximation?

Will show one result from [Balcan-Blum-Gupta'09] for getting error $O(\epsilon/(c-1))$ under stability to k-median

Clustering from (c,ε) k-median stability

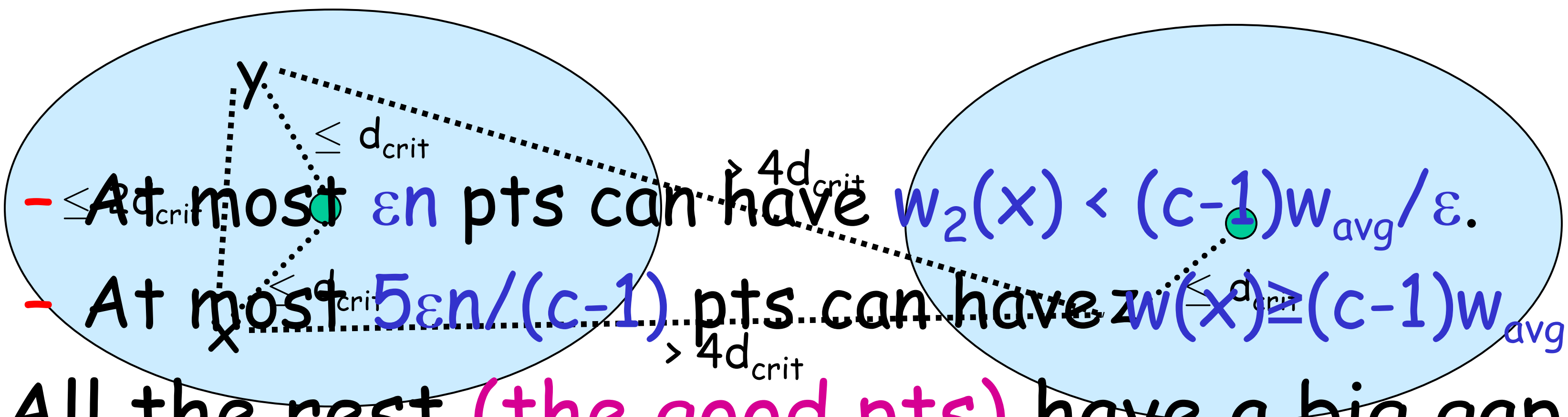
- For simplicity, say target *is* k-median opt, and for now, that all clusters of size $> 2\varepsilon n$.

- For any x, let $w(x)$=dist to own center, $w_2(x)$=dist to 2nd-closest center.
- Let $w_{avg}=avg_x\, w(x)$. $[OPT = n \cdot w_{avg}]$
- Then:
  - At most εn pts can have $w_2(x) < (c-1)w_{avg}/\varepsilon$.
  - At most $5\varepsilon n/(c-1)$ pts can have $w(x) \geq (c-1)w_{avg}/5\varepsilon$.
- All the rest (the good pts) have a big gap.



Clustering from (c,ε) k-median stability

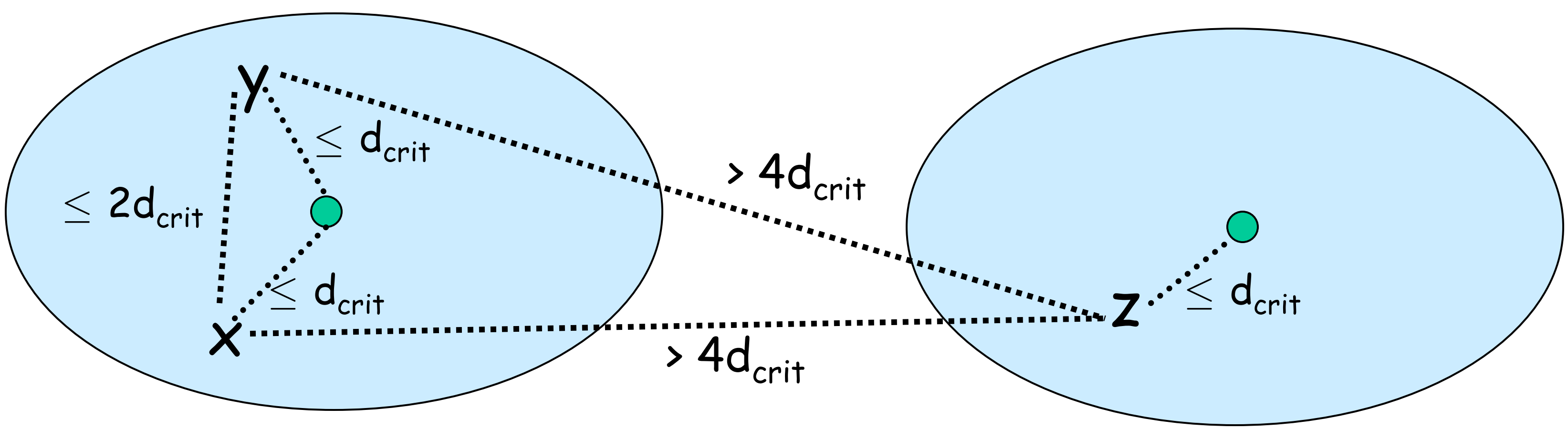
- Define critical distance $d_{crit}=(c-1)w_{avg}/5\varepsilon$.
- So, a $1-O(\varepsilon)$ fraction of pts look like:



  – At most εn pts can have $w_2(x) < (c-1)w_{avg}/\varepsilon$.
  – At most $5\varepsilon n/(c-1)$ pts can have $w(x) \geq (c-1)w_{avg}/5\varepsilon$.
- All the rest (the good pts) have a big gap.

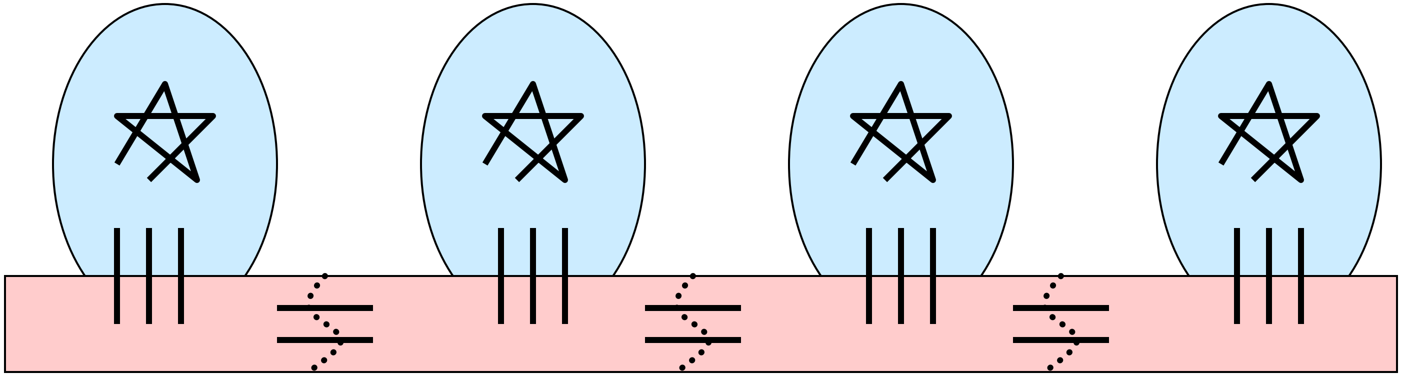
Clustering from (c,ε) k-median stability

- So if we define a graph G connecting any two pts within distance $\leq 2d_{crit}$, then:
  - Good pts within cluster form a clique
  - Good pts in different clusters have no common nbrs
- So, a $1\text{-}O(\varepsilon)$ fraction of pts look like:



Clustering from (c,ε) k-median stability

- So if we define a graph G connecting any two pts within distance $\leq 2d_{crit}$, then:
  - Good pts within cluster form a clique
  - Good pts in different clusters have no common nbrs
- So, the world now looks like:

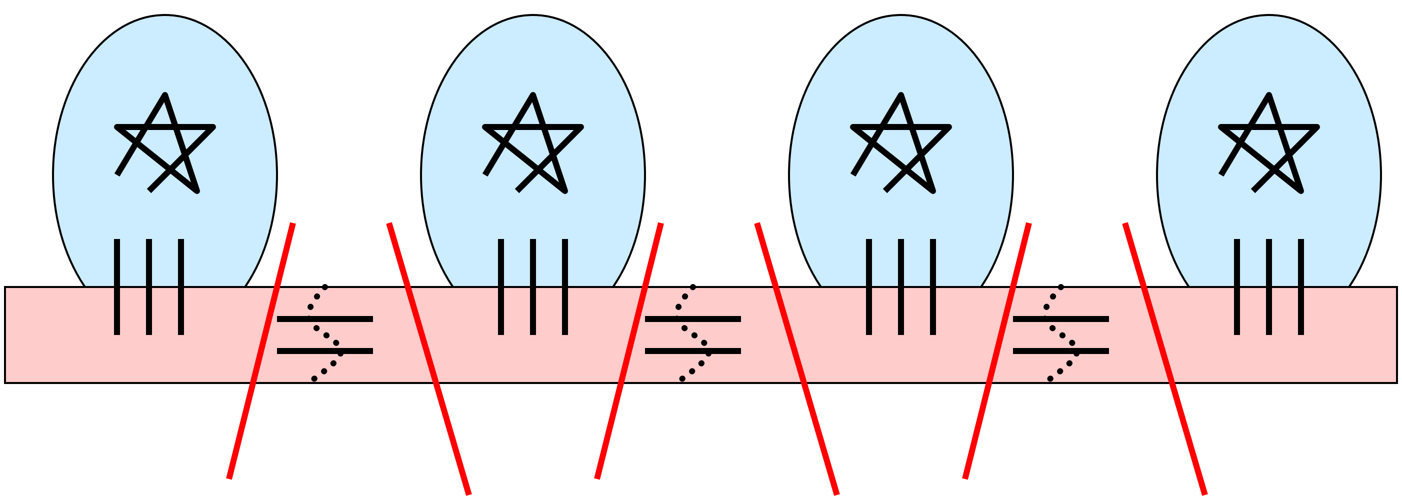


Clustering from (c,ε) k-median stability

- If furthermore all clusters have size $> 2b+1$, where b = # bad pts = $O(\varepsilon n/(c-1))$, then:
  - Create graph H where connect x,y if share > b nbrs in common in G.
  - Output k largest components in H.

(only makes mistakes on bad points)

- So, the world now looks like:

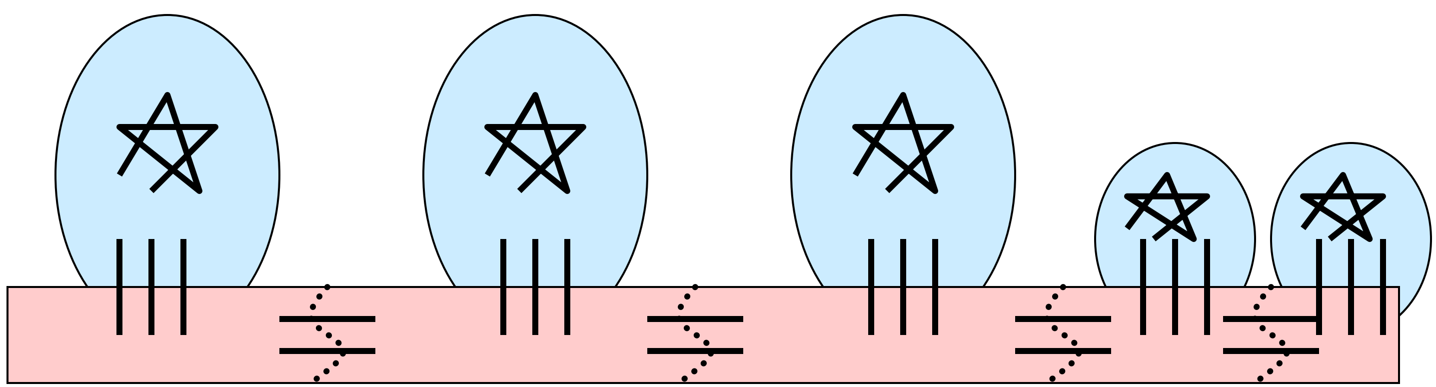


Clustering from (c,ε) k-median stability

If clusters not so large, then need to be more careful but can still get error $O(\varepsilon/(c-1))$.

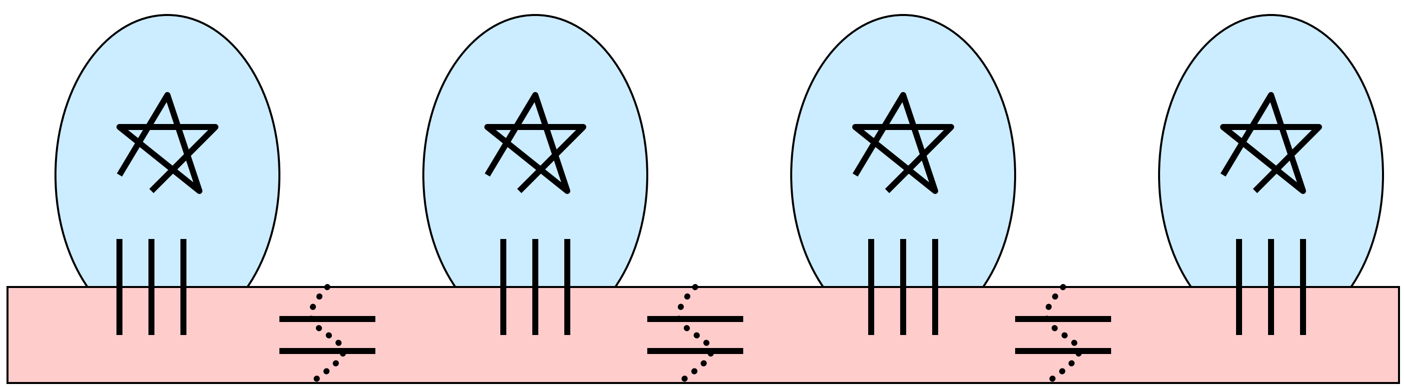
Could have some clusters dominated by bad pts…

Actually, algorithm is not too bad (but won't go into here).



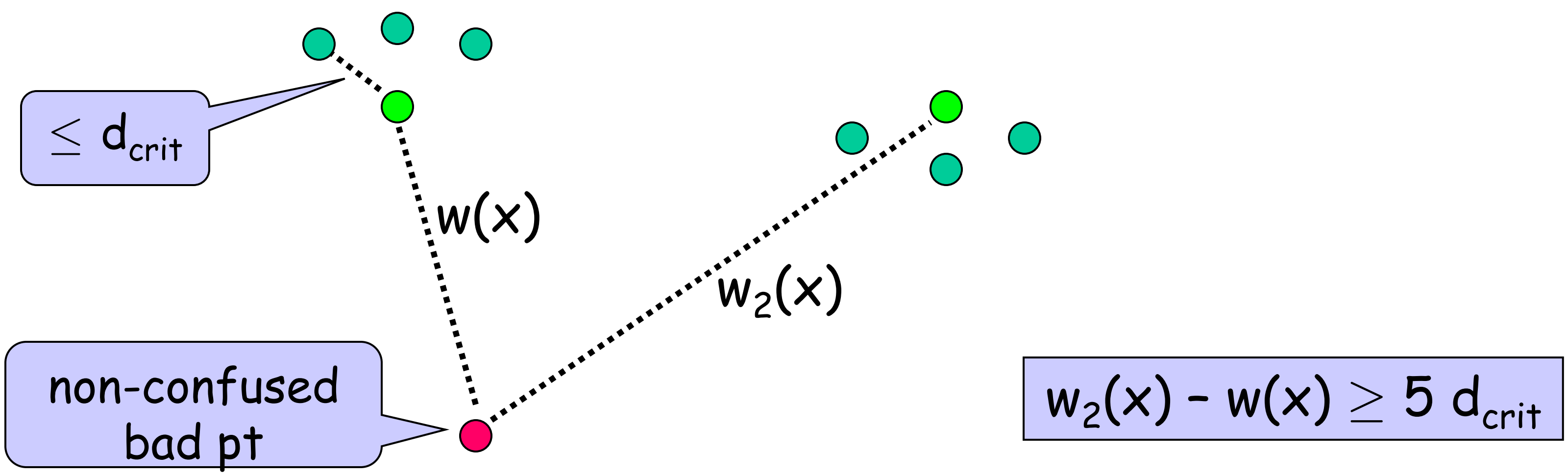
$O(\varepsilon)$-close $\Rightarrow$ ε-close

- Back to the large-cluster case: can improve to get ε-close. (for any $c>1$, but "large" depends on c).
- Idea: Really two kinds of bad pts.
  - At most εn "confused": $w_2(x)-w(x) < (c-1)w_{avg}/\varepsilon$.
  - Rest not confused, just far: $w(x) \geq (c-1)w_{avg}/5\varepsilon$.
- Can recover the non-confused ones…



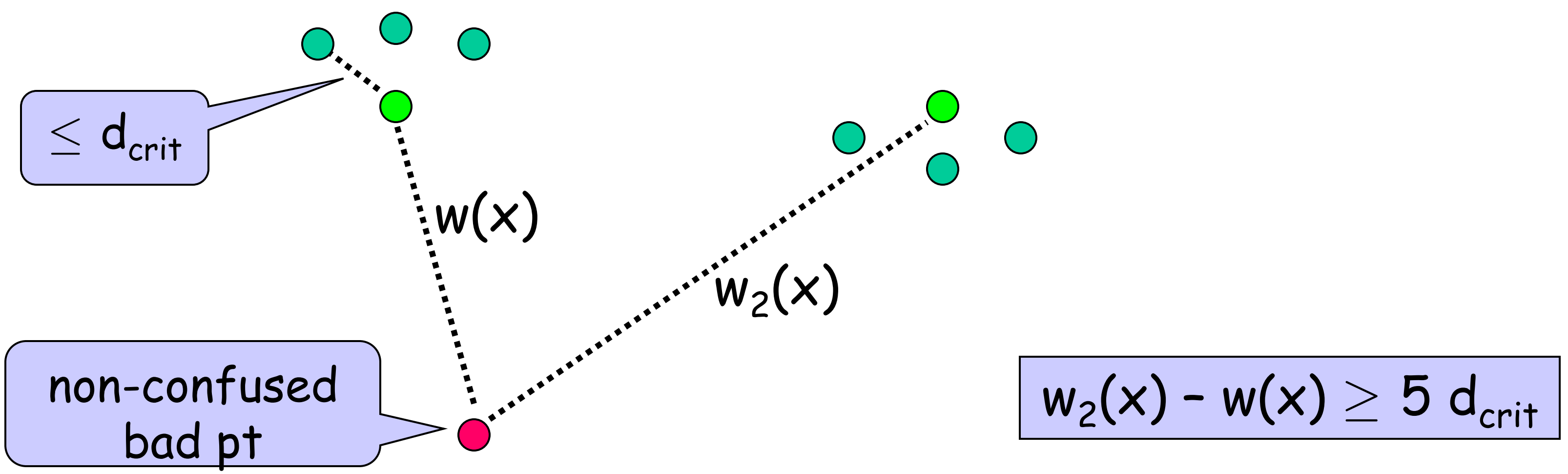
$O(\varepsilon)$-close $\Rightarrow$ ε-close

- Back to the large-cluster case: can improve to get ε-close. (for any c>1, but "large" depends on c).
- Idea: Really two kinds of bad pts.
  - At most εn "confused": $w_2(x)-w(x) < (c-1)w_{avg}/\varepsilon$.
  - Rest not confused, just far: $w(x) \geq (c-1)w_{avg}/5\varepsilon$.
- Can recover the non-confused ones…



$O(\varepsilon)$-close $\Rightarrow$ ε-close

- Back to the large-cluster case: can improve to get ε-close. (for any $c>1$, but "large" depends on c).
  - Given output C' from alg so far, reclassify each x into cluster of lowest median distance
  - Median is controlled by good pts, which will pull the non-confused points in the right direction.



$O(\varepsilon)$-close $\Rightarrow$ ε-close

- Back to the large-cluster case: can improve to get ε-close. (for any $c>1$, but "large" depends on c).
  - Given output C' from alg so far, reclassify each x into cluster of lowest median distance
  - Median is controlled by good pts, which will pull the non-confused points in the right direction.

A bit like 2-rounds of k-means/Lloyd's algorithm

Stepping back…

- Have shown that (c,ε) approx-stability for k-median allows us to get ε-close (for large clusters) or $O(\varepsilon)$-close (for general cluster sizes)

What about in practice?

- [Voevodski-Balcan-Roglin-Teng-Xia UAI'10]
  - Consider protein sequence clustering problem.
  - Even if property doesn't strictly hold, still provides a very useful guide to algorithm design.

Stepping back…

- Have shown that (c,ε) approx-stability for k-median allows us to get ε-close (for large clusters) or $O(\varepsilon)$-close (for general cluster sizes)

What about in practice?

- [Voevodski-Balcan-Roglin-Teng-Xia UAI'10]
  - In this setting, can only perform small number of one-versus-all distance queries.
  - Design algorithm with good performance under approx-stability. Apply to datasets with known correct solutions (Pfam, SCOP databases)
  - Fast and high accuracy.

Stepping back…

- [Voevodski-Balcan-Roglin-Teng-Xia UAI'10]
  - Design algorithm with good performance under approx-stability. Apply to datasets with known correct solutions (Pfam, SCOP databases)
  - Fast and high accuracy.

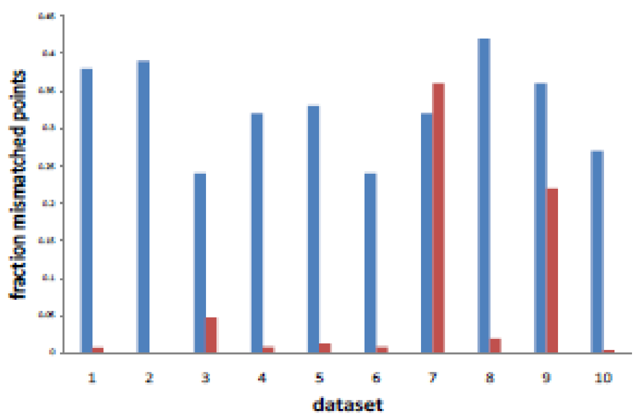


Figure 1: Comparing the performance of k-means in the embedded space (blue) and *Landmark-Clustering* (red) on 10 datasets from Pfam. Datasets **1-10** are created by randomly choosing 8 families from Pfam of size s, $1000 \leq s \leq 10000$.

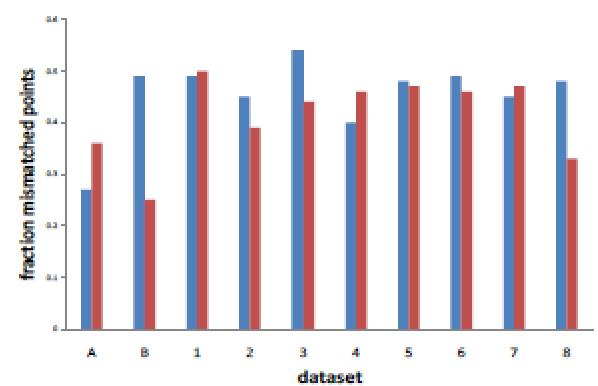


Figure 2: Comparing the performance of spectral clustering (blue) and *Landmark-Clustering* (red) on 10 datasets from SCOP. Datasets **A** and **B** are the two main examples from [10], the other datasets (**1-8**) are created by randomly choosing 8 superfamilies from SCOP of size s, $20 \leq s \leq 200$.

Stepping back…

- [Voevodski-Balcan-Roglin-Teng-Xia UAI'10]
  - Design algorithm with good performance under approx-stability. Apply to datasets with known correct solutions (Pfam, SCOP databases)
  - Fast and high accuracy.

Even if property doesn't strictly hold, gives a useful guide to algorithm design.

Extensions

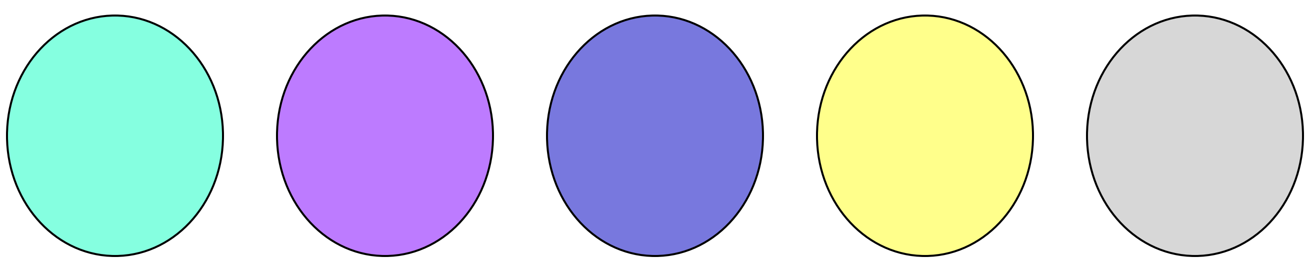
[Awasthi-B-Sheffet'10]

All c-approximations are ε-close

$\Downarrow$

All c-approximations use at least k clusters

(Strictly weaker condition if all target clusters of size $\geq \epsilon n$, since that implies a k-1 clustering can't be ϵ-close)



Extensions

[Awasthi-B-Sheffet'10]

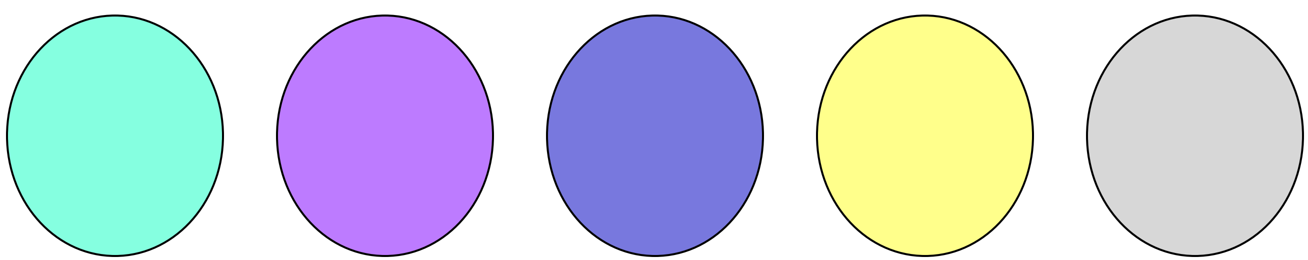
All c-approximations are ε-close

↓

All c-approximations use at least k clusters

↓

Deleting a center of OPT is not a c-approximation



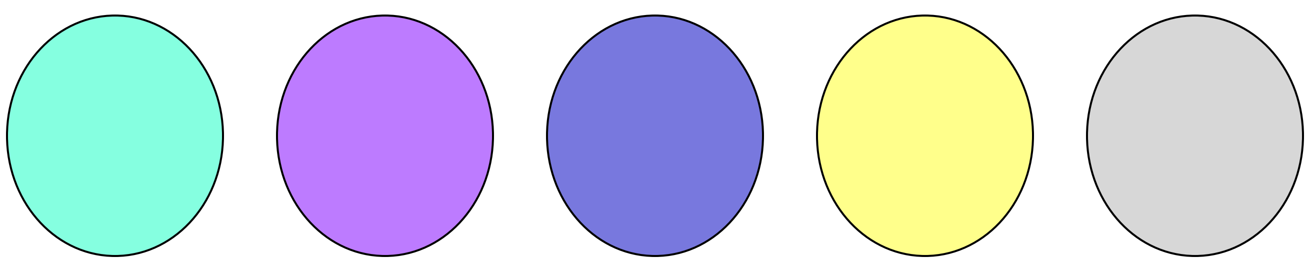
Extensions

[Awasthi-B-Sheffet'10]

Deleting a center of OPT is not a c-approximation

- Under this condition, for any constant c>1, get PTAS: 1+α apx in polynomial time for any constant α. (k-median/k-means)

- Implies getting ϵ-close solution under original condition (set 1+α = c).



What about other problems?

What about other problems?

Nash equilibria?

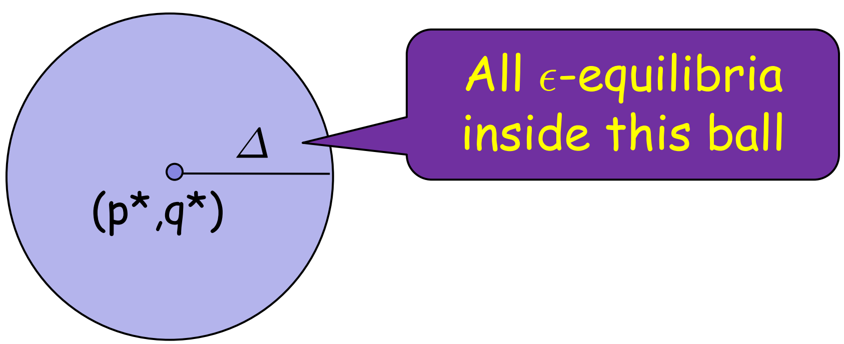
Sparsest cut?

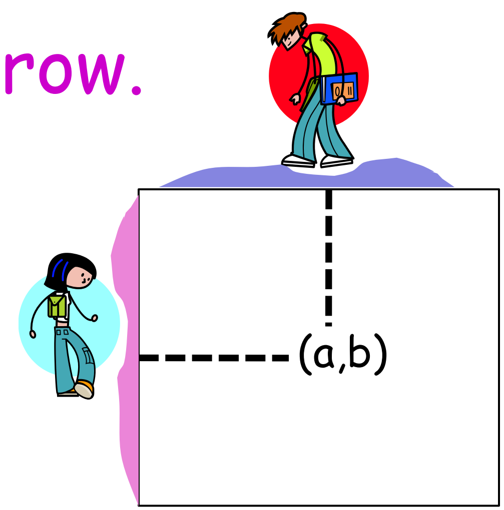
Phylogenetic Trees?

What about other problems?

Nash equilibria

- What if the reason we want to find an apx Nash equilibrium is to predict how people will play?
- Then it's natural to focus on games where all apx equilibria are close to each other.
- Does this make the problem easier to solve?
- Pranjal Awasthi will talk about tomorrow.

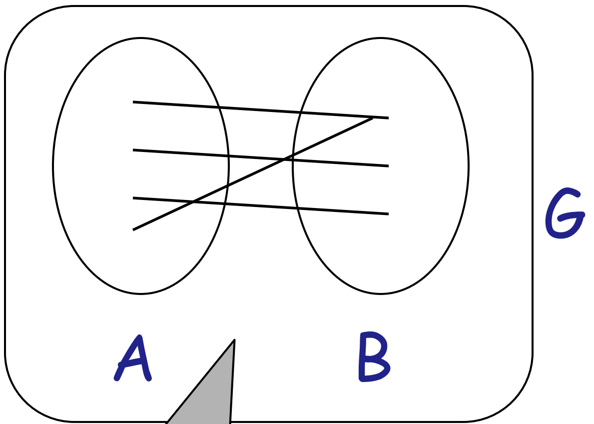




What about other problems?

Sparsest cut?

- Best apx is $O((\log n)^{1/2})$ [ARV]
- Often the reason you want a good cut is to segment an image, partition cats from dogs, etc. (edges represent similarity)
- Implicitly hoping good apx implies low error…
- What if assume any 10-apx has error $\leq \epsilon$?



Minimize
e(A,B)/(|A|*|B|)

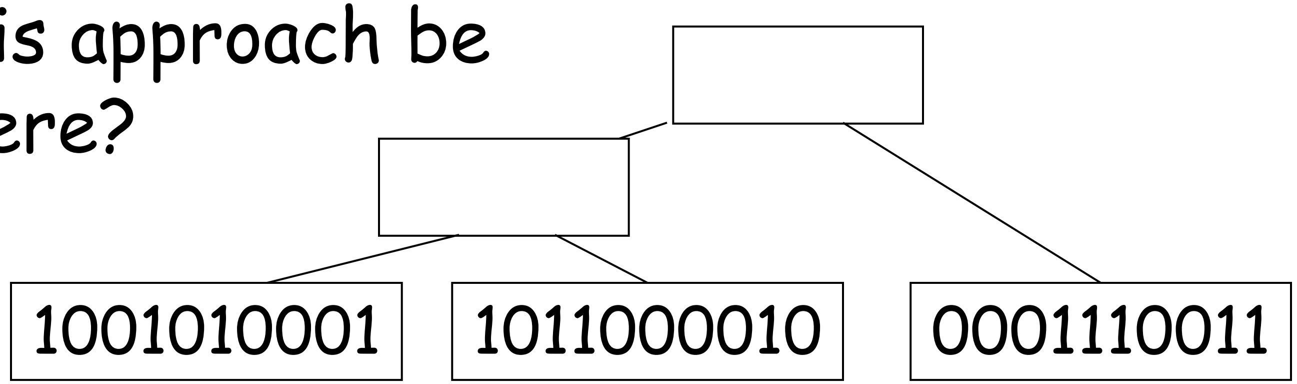
Good open question!

What about other problems?

Phylogenetic Trees?

Trying to reconstruct evolutionary trees

- Often posed as a Steiner-tree-like optimization problem.
- But really our goal is to get structure close to the correct answer.
- Could this approach be useful here?



Summary & Open Problems

For clustering, can say "if data has the property that a 1.1 apx to [pick one: k-median, k-means, min-sum] would be sufficient to have error ϵ then we can get error $O(\epsilon)$" …even though you might think NP-hardness results for approximating these objectives would preclude this.

Notion of Approx-Stability makes sense to examine for other optimization problems where objective function may be a proxy for something else.

Open question #1: other problems?

- Nash equilibria
- Sparsest cut?
- Evolutionary trees?

Summary & Open Problems

Open question #2: what if we only assume most c-approximations are close to target? Can we get positive results from that?

Open question #3: for k-median, general bound was $O(\epsilon/(c-1))$. What if only assume that $(1+\epsilon)$-apx is ϵ-close? [recall that best known apx is factor of 3, so would be impressive to be able to do this]

Open question #4: for "easy" problems: given arbitrary instance, find stable portions of solution.

Summary & Open Problems

Open question #5: connection to & combinations with Bilu-Linial perturbation-stability notion. [very nice clustering alg of Balcan and Liang for perturbation-stable instances that breaks factor-3 barrier]