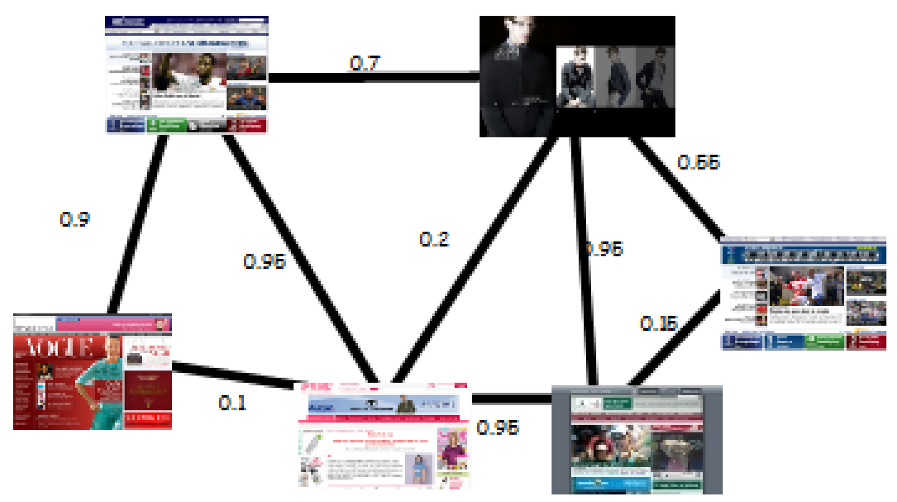
Clustering Perturbation Resilient k-Median Instances

Maria-Florina Balcan and Yingyu Liang

Georgia Tech College of Computing

Problem Setup

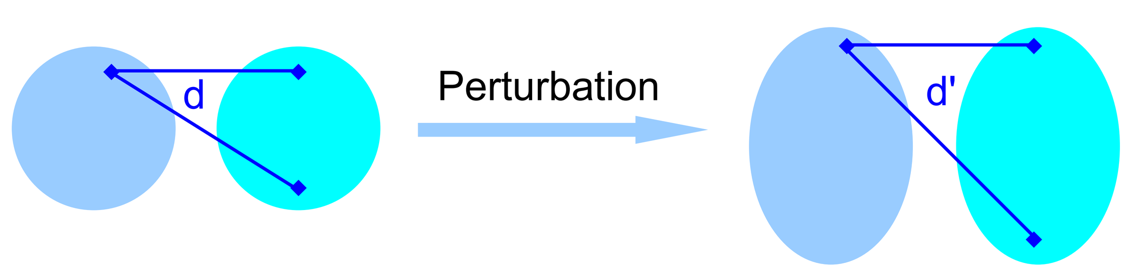
▷ **k-Median Clustering:** Given a set of n points in metric space, find centers $\mathbf{x} = \{x_1, \ldots, x_k\}$ to minimize $\sum_{p \in P} \min_i d(p, c_i)$.



▷ **New Direction:** exploit additional stability properties of the data

α-perturbation of d: a function d' s.t. $\forall p, q \in S, d'(p,q) \in [1, \alpha] d(p,q)$

Definition. [Bilu-Linial, ICS10; Awasthi-Blum-Sheffet, IPL12] *An instance is α-perturbation resilient if the optimal clustering under α-perturbation of the distance is unique and equal to the original optimal clustering.*



Definition. [Balcan-Liang, ICALP12] *An instance is (α, ϵ)-perturbation resilient, if the optimal clustering under any α-perturbation of distance can be obtained by moving at most ϵ fraction of the points in the original optimal clustering.*

Our Results

Efficient algorithm for (α, ϵ)-PR k-median instances

- produces $(1 + O(\epsilon/\rho))$-approx for $\alpha > 4$, where $\rho = \min_i |C_i|/n$
- improve over the bound $\alpha > 2 + \sqrt{7}$ in [Balcan-Liang, ICALP12]

Sublinear time algorithm for constructing implicit clustering

- produces $2(1 + O(\epsilon/\rho))$-approx for $\alpha > 4$
- running time logarithmic in #points

Algorithms

▷ **Algorithm 1** ($(1 + O(\epsilon/\rho))$-approx algorithm)

1. Generate a list of blobs as described above
2. Use existing robust linkage algo to link them into a tree
3. Use dynamic programming to get the lowest cost pruning

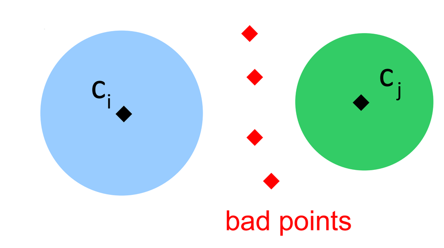
▷ **Algorithm 2** (Sublinear time algorithm)

1. Sample points from the original data
2. Run Algorithm 1 on the sample

Structure Property of (α, ϵ)-PR k-Median

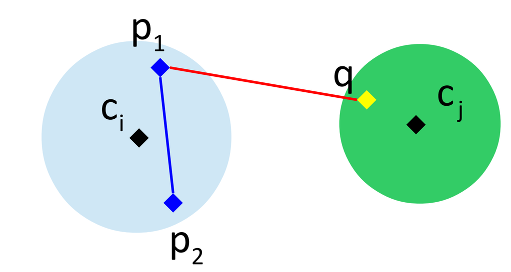
▷ Bounds on #bad points [Balcan-Liang, ICALP12]

Theorem. *Assume $\min_i |C_i| = \Omega(\epsilon n)$. Except for $\le \epsilon n$ bad points, any other good point is α times closer to its own center than to other centers.*



▷ Neighbors of good points

Lemma. *When $\alpha > 4$, for any good points $p_1, p_2 \in G_i, q \in G_j (j \neq i)$, we have $d(p_1, p_2) < d(p_1, q)$.*



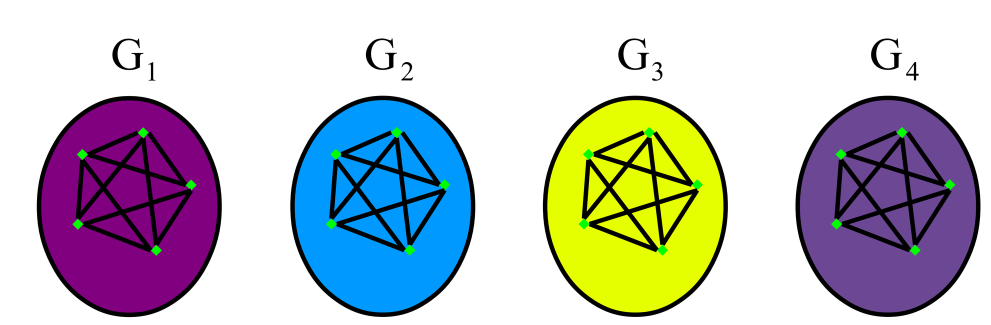
Generating Blobs

▷ **Algorithm**

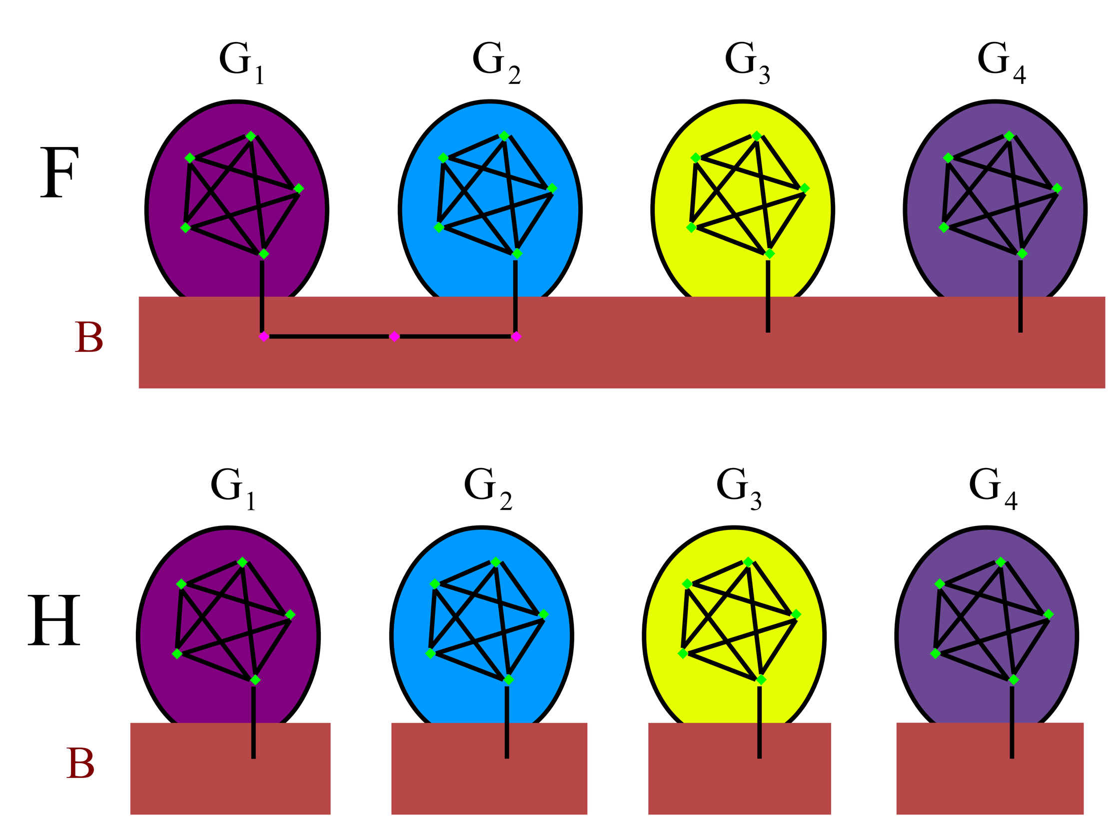
1. Initialize $t = \min_i |C_i|$
2. Build F_t: connect points with enough common neighbors in their t nearest neighbors
3. Build H_t: connect points with enough neighbors in F_t
4. Pull out sufficiently large blobs
5. If a point has sufficiently many points in the blobs, then insert it to the one with smallest median distance
6. While $\exists$ points left, $t = t + 1$ and Goto Step 2.

▷ **Intuition**

1. Simple case: no bad points, all clusters have size n_C
Algo: Build graph F by connecting points to the n_C nearest neighbors

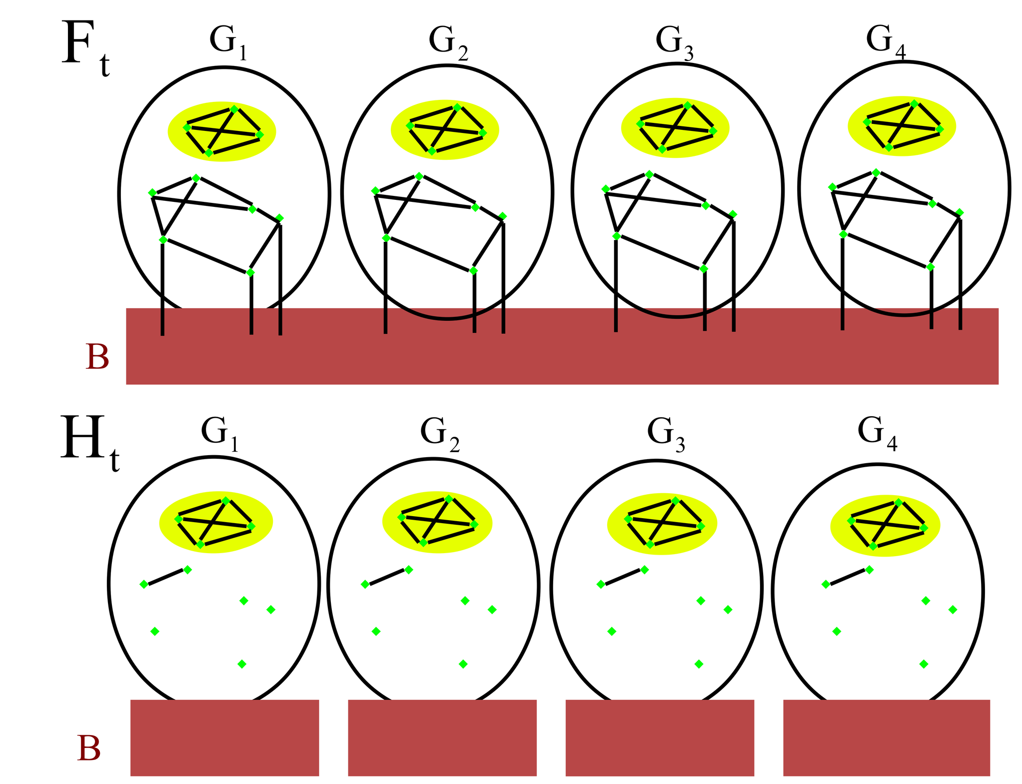


2. Simple case 2: with bad points, all clusters have size n_C
Algo: Build graph H based on F

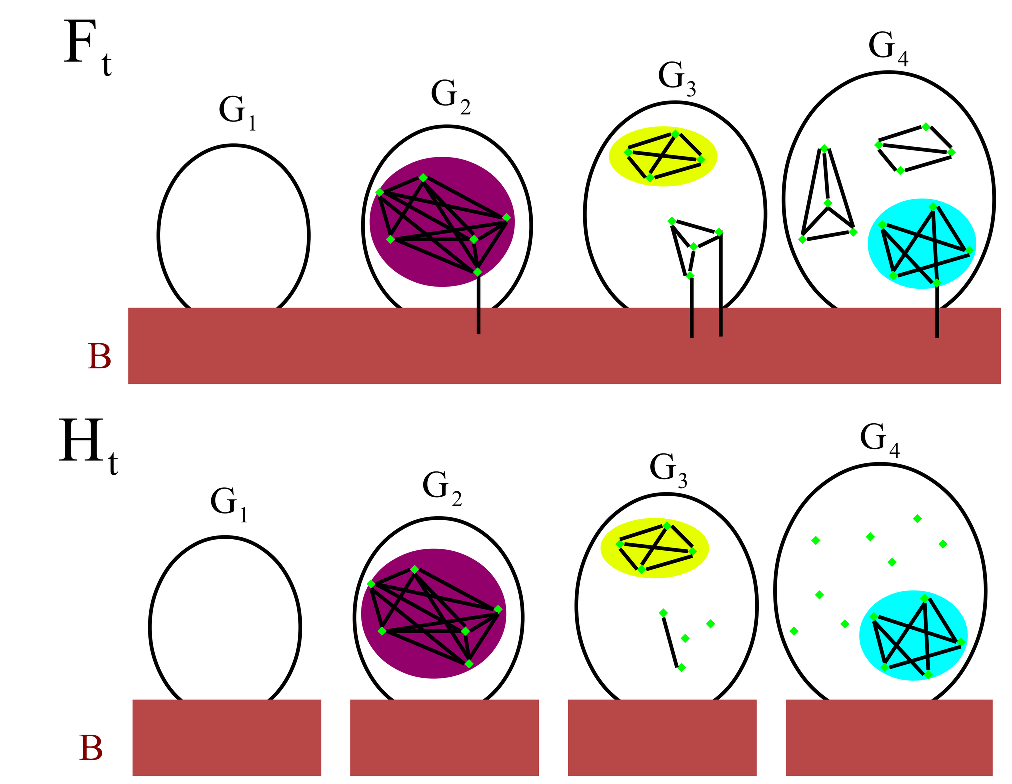


Generating Blobs: continue

3. More difficult case: with bad points, n_C unknown
Algo: Start with a threshold t, build F_t H_t, pull out blobs



4. General case: with bad points, clusters have different sizes
Algo: Pull out blobs as before
Keypoint: when $t = |C_i|$, all good points in C_i are pulled out



▷ **Analysis**

- Each blob only contains good points from one cluster
- Blobs from the same cluster are first linked by robust linkage
- There is a pruning that assigns all good points correctly

Sublinear Time Algorithm

Let $\tilde{\mathbf{c}}$ be the output of Algorithm 1 on a sample S' of size $\tilde{\Theta}(\frac{k}{\epsilon^2})$. To show $\mathrm{cost}(S, \tilde{\mathbf{c}}) \approx \mathrm{cost}(S, \mathbf{c})$

- Average costs of S' and S are close on any set of centers, so it suffices to show $\mathrm{cost}(S', \tilde{\mathbf{c}}) \approx \mathrm{cost}(S', \mathbf{c})$.
- Algorithm 1 builds a tree with a pruning $\mathcal{P}$ assigning all good points correctly. Let $\mathbf{c}'$ be the best centers for it.
- Use $\mathrm{cost}(\mathcal{P}, \mathbf{c}')$ as a bridge:
  - $\mathrm{cost}(S', \tilde{\mathbf{c}}) \approx \mathrm{cost}(\tilde{\mathcal{C}}, \tilde{\mathbf{c}}) \approx \mathrm{cost}(\mathcal{P}, \mathbf{c}')$
  - $\mathrm{cost}(\mathcal{P}, \mathbf{c}') \le 2\mathrm{cost}(\mathcal{P}, \mathbf{c}) \approx 2\mathrm{cost}(S', \mathbf{c})$