Hierarchical Clustering: Objectives & Algorithms

Vincent Cohen-Addad
Paris Sorbonne & CNRS

Varun Kanade
University of Oxford

Frederik Mallmann-Trenn
MIT

Claire Mathieu
École normale supérieure & CNRS

Clustering

Flat Clustering

- Often data can be grouped together into subsets that are coherent, called clusters
- Data in the same cluster is typically more similar that data across different clusters

Clustering

Flat Clustering

- Often data can be grouped together into subsets that are coherent, called clusters
- Data in the same cluster is typically more similar that data across different clusters

Cluster the following news headlines in 3 categories

Will AI take over?
Black holes swallow stars whole according to new study
Wenger signs new two year deal
England will attack during Champions trophy

Example credit: Avrim Blum

Clustering

Flat Clustering

- Often data can be grouped together into subsets that are coherent, called clusters
- Data in the same cluster is typically more similar that data across different clusters

Cluster the following news headlines in 3 categories

CS	Will AI take over?
Physics	Black holes swallow stars whole according to new study
Sports	Wenger signs new two year deal
Sports	England will attack during Champions trophy

Example credit: Avrim Blum

Clustering

Flat Clustering

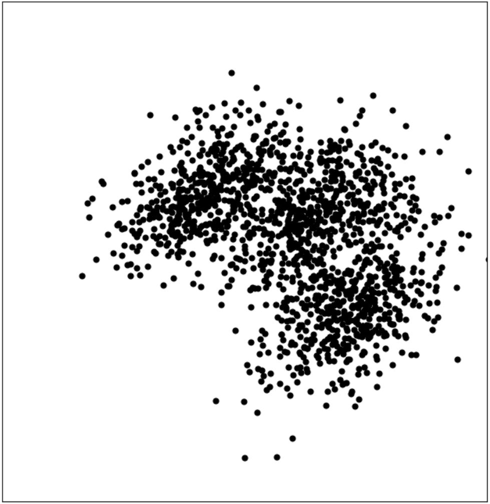
- Often data can be grouped together into subsets that are coherent, called clusters
- Data in the same cluster is typically more similar that data across different clusters

Cluster the following news headlines in 3 categories

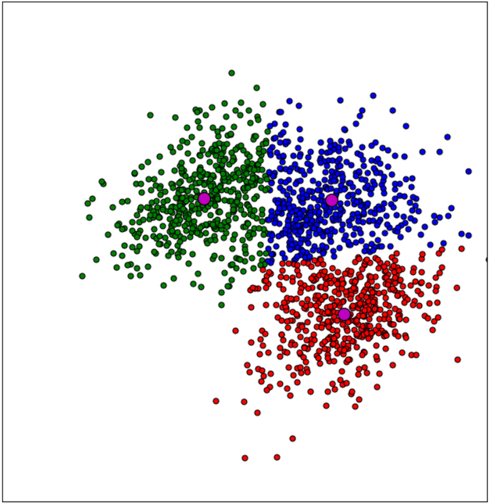
Science	Will AI take over?
Science	Black holes swallow stars whole according to new study
Football	Wenger signs new two year deal
Cricket	England will attack during Champions trophy

Example credit: Avrim Blum

(Flat) Clustering



(Flat) Clustering

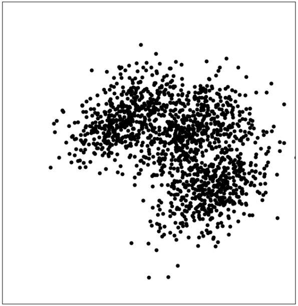


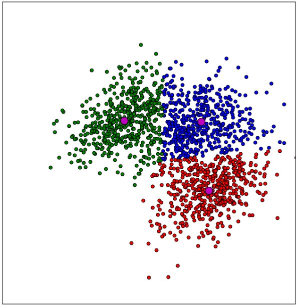
(Flat) Clustering: Objectives and Algorithms

Data lies in some metric space $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^D$

Find k points $\mu_1, \ldots, \mu_k$ that minimize, e.g.

1. k-median objective $\displaystyle\sum_{i=1}^{N} \left(\min_{j \in [k]} d(\mathbf{x}_i, \mu_j) \right)$
2. k-means objective $\displaystyle\sum_{i=1}^{N} \left(\min_{j \in [k]} d(\mathbf{x}_i, \mu_j) \right)^2$



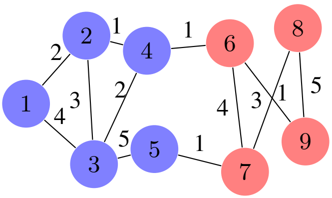


(Flat) Clustering: Objectives and Algorithms

1. k-medians objective $\sum_{i=1}^{N} \left(\min_{j \in [k]} d(\mathbf{x}_i, \mu_j) \right)$
2. k-means objective $\sum_{i=1}^{N} \left(\min_{j \in [k]} d(\mathbf{x}_i, \mu_j) \right)^2$

- Minimizing these objective functions is NP-hard
- Approximation algorithms are known

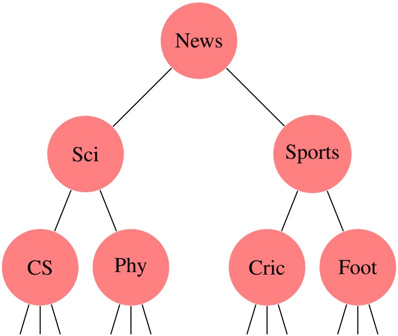
Clustering: Input as (Dis)-Similarity Graph



- Edge weights represent similarities
- Graph partitioning algorithms, e.g., mincut, sparsest cut, multi-way cut
- Many of these problems are NP-complete
- Approximation algorithms are widely studied
- Spectral partitioning algorithms can be highly efficient

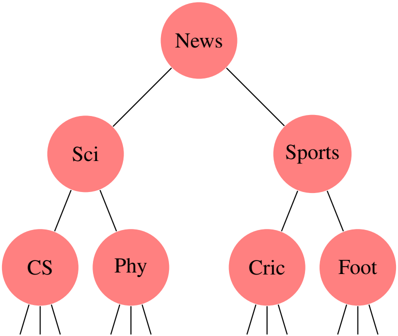
Hierarchical Clustering

- Recursive partitioning of data at an increasingly finer granularity represented as a tree
- The leaves of the hierarchical cluster tree represent data.



Hierarchical Clustering

- Recursive partitioning of data at an increasingly finer granularity represented as a tree
- The leaves of the hierarchical cluster tree represent data.



Will AI take over?

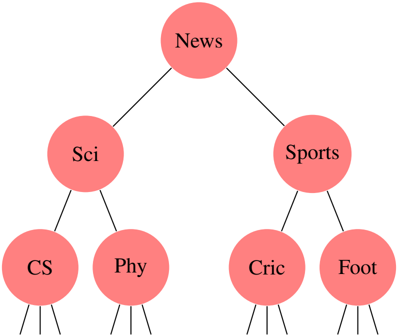
Someone finally figured out why neural nets work

Black holes swallow stars whole according to new study

Neymar breaks his leg and stops football
Someone finally figured out the rules of cricket

Hierarchical Clustering

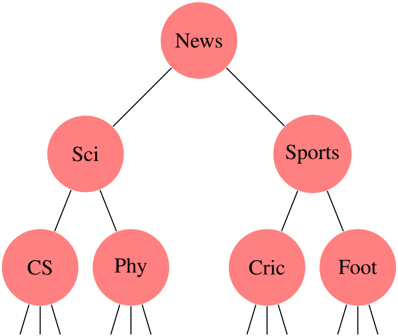
- Recursive partitioning of data at an increasingly finer granularity represented as a tree
- The leaves of the hierarchical cluster tree represent data.



Science	Will AI take over?
Science	Someone finally figured out why neural nets work
Science	Black holes swallow stars whole according to new study
Sports	Neymar breaks his leg and stops football
Sports	Someone finally figured out the rules of cricket

Hierarchical Clustering

- Recursive partitioning of data at an increasingly finer granularity represented as a tree
- The leaves of the hierarchical cluster tree represent data.



CS	Will AI take over?
CS	Someone finally figured out why neural nets work
Physics	Black holes swallow stars whole according to new study
Football	Neymar breaks his leg and stops football
Cricket	Someone finally figured out the rules of cricket

Hierarchical Clustering in Practice: Linkage Algorithms

- We are given pairwise similarities between (some) pairs of datapoints

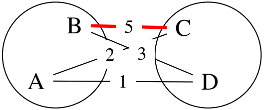
Hierarchical Clustering in Practice: Linkage Algorithms

- We are given pairwise similarities between (some) pairs of datapoints
- Initially each data point is its own clusters
- Repeatedly merge most similar clusters
- Builds up cluster tree bottom-up

Hierarchical Clustering in Practice: Linkage Algorithms

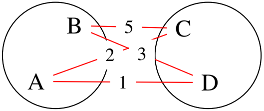
- We are given pairwise similarities between (some) pairs of datapoints
- Initially each data point is its own clusters
- Repeatedly merge most similar clusters
- Builds up cluster tree bottom-up

Single Linkage



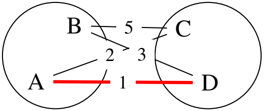
Similarity: 5

Average Linkage



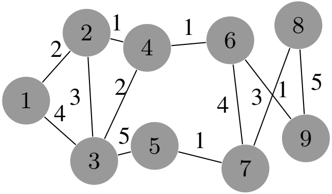
Similiarity: 2.75

Complete Linkage

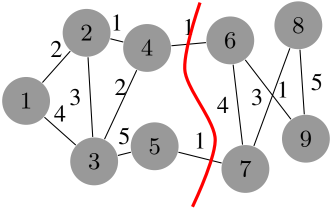


Similiarity: 1

Hierarchical Clustering: Divisive Heuristics

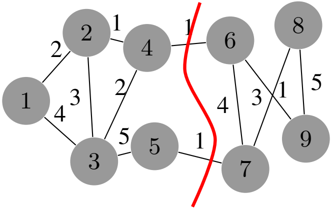


Hierarchical Clustering: Divisive Heuristics



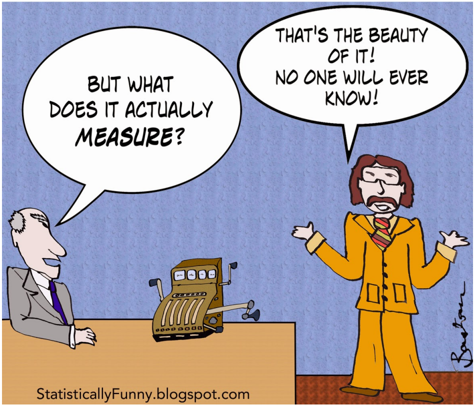
- Find a partition of the input similarity graph (or set of points)
  - Split using bisection k-means
  - Split using sparsest cut

Hierarchical Clustering: Divisive Heuristics



- Find a partition of the input similarity graph (or set of points)
  - Split using bisection k-means
  - Split using sparsest cut
- Recurse on each part
- Builds cluster-tree top-down

What are these algorithms actually doing?



What quantity are these algorithms optimizing?

- For flat clustering, algorithms designed to optimize some objective function
  - We can decide quantitatively which one is the best
- For hierarchical clustering, algorithms have been studied procedurally
  - Thus, comparisons between hierarchical clustering algorithms are only qualitative

What quantity are these algorithms optimizing?

- For flat clustering, algorithms designed to optimize some objective function
  - We can decide quantitatively which one is the best
- For hierarchical clustering, algorithms have been studied procedurally
  - Thus, comparisons between hierarchical clustering algorithms are only qualitative
- [Dasgupta '16]

  "The lack of an objective function has prevented a theoretical understanding"

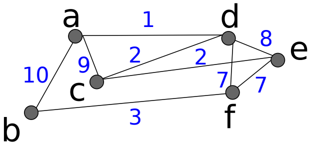
What quantity are these algorithms optimizing?

- For flat clustering, algorithms designed to optimize some objective function
  - We can decide quantitatively which one is the best
- For hierarchical clustering, algorithms have been studied procedurally
  - Thus, comparisons between hierarchical clustering algorithms are only qualitative
- [Dasgupta '16]

  > *"The lack of an objective function has prevented a theoretical understanding"*

- Dasgupta introduced an objective function to model the hierarchical clustering problem

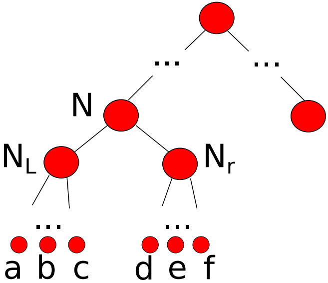
Dasgupta's Cost Function



Input: a weighted similarity graph G

- Edge weights represent similarities

Output: T a tree with leaves labelled by nodes of G



Cost of the output: Sum of the costs of the nodes of T

Cost of a node N of the tree:

$A = \{u \mid u \text{ is leaf of subtree rooted at } N_L\}$

$B = \{v \mid v \text{ is leaf of subtree rooted at } N_R\}$

$$\text{cost}(N) = (|A| + |B|) \cdot \sum_{\substack{u \in A \\ v \in B}} \text{similarity}(u, v)$$

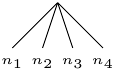
Intuition: Better to cut a high similarity edge at a lower level

Cost of N = $(3 + 3) \cdot (1 + 2 + 2 + 3)$

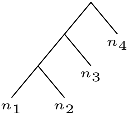
Dasgupta's Cost Function

Some Desirable Properties

- Using binary trees can always reduce cost



cost = $(n_1 + \cdots + n_4) \cdot$
$(w(A_1, A_2) + \cdots + w(A_3, A_4))$

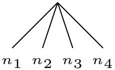


cost = $(n_1 + n_2) w(A_1, A_2) + \cdots +$
$+ (n_1 + n_2 + n_3 + n_4) w(A_1 \cup A_2 \cup A_3, A4)$

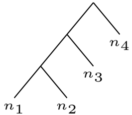
Dasgupta's Cost Function

Some Desirable Properties

- Using binary trees can always reduce cost



cost = $(n_1 + \cdots + n_4) \cdot$
$(w(A_1, A_2) + \cdots + w(A_3, A_4))$



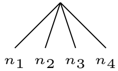
cost = $(n_1 + n_2)w(A_1, A_2) + \cdots +$
$+ (n_1 + n_2 + n_3 + n_4)w(A_1 \cup A_2 \cup A_3, A4)$

- Disconnected components must be separated first

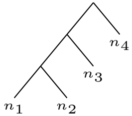
Dasgupta's Cost Function

Some Desirable Properties

- Using binary trees can always reduce cost



cost = $(n_1 + \cdots + n_4) \cdot (w(A_1, A_2) + \cdots + w(A_3, A_4))$



cost = $(n_1 + n_2) w(A_1, A_2) + \cdots + + (n_1 + n_2 + n_3 + n_4) w(A_1 \cup A_2 \cup A_3, A4)$

- Disconnected components must be separated first

- For unit-weight cliques, all binary trees have the same cost

- For planted partition random graphs, the optimal tree first separates according to the partition

Cost Functions: An Axiomatic Approach

- Are there other suitable cost functions?
- What properties should cost functions satisfy?

Cost Functions: An Axiomatic Approach

- Are there other suitable cost functions?
- What properties should cost functions satisfy?

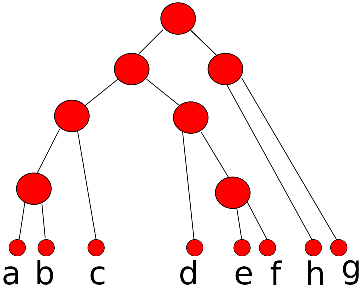
Admissible Cost Function

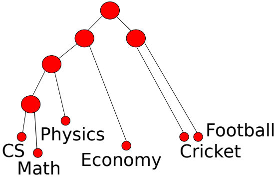
If the input has an underlying “ground-truth” hierarchical clustering tree,

then any tree should be optimal with respect to the cost function if and only if it is a “ground-truth” tree.

Inputs with an Underlying "Ground-Truth" Hierarchical Clustering

There exists a hierarchical clustering of the input,





such that:

- similarity$(a, b) >$ similarity$(a, c) >$ similarity(b, f),
- similarity(a, c) = similarity(b, c).

We want

> If the input graph has such an underlying structure then the above tree is the optimal one w.r.t. the cost function.

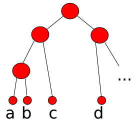
Inputs with an Underlying "Ground-Truth" Hierarchical Clustering

Ultrametrics to generate ground-truth inputs:

Assume that the data elements $x_1, \ldots, x_n$ lie in some ultrametric:

$$d(x_i, x_j) \leq \max(d(x_i, x_\ell), d(x_j, x_\ell)) \quad \forall i, j, \ell$$

can be represented as a weighted tree:



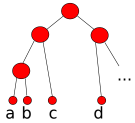
Inputs with an Underlying "Ground-Truth" Hierarchical Clustering

Ultrametrics to generate ground-truth inputs:

Assume that the data elements $x_1, \ldots, x_n$ lie in some ultrametric:

$$d(x_i, x_j) \leq \max(d(x_i, x_\ell), d(x_j, x_\ell)) \quad \forall i, j, \ell$$

can be represented as a weighted tree:



A weighted graph G is a ground-truth input if there exists an ultrametric and a non-increasing function f such that similarity$(u, v) = f(d(x_u, x_v)), \forall u, v$.

The tree represents the ground-truth hierarchical clustering.

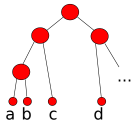
Inputs with an Underlying "Ground-Truth" Hierarchical Clustering

Ultrametrics to generate ground-truth inputs:

Assume that the data elements $x_1, \ldots, x_n$ lie in some ultrametric:

$$d(x_i, x_j) \le \max(d(x_i, x_\ell), d(x_j, x_\ell)) \quad \forall i, j, \ell$$

can be represented as a weighted tree:



A weighted graph G is a ground-truth input if there exists an ultrametric and a non-increasing function f such that similarity$(u, v) = f(d(x_u, x_v)), \forall u, v$.

The tree represents the ground-truth hierarchical clustering.

Theorem: All the algorithms used in practice output the ground-truth hierarchical clustering on a ground-truth input.

Admissible Cost Functions

Ground-Truth Input

A weighted graph G is a ground-truth input if there exists an ultrametric and a non-increasing function f such that similarity$(u, v) = f(d(x_u, x_v)), \forall u, v$.

Admissible Costs Functions

For any ground-truth input, a tree is optimal if and only if it is a ground-truth tree (i.e.: the ultrametric tree).

Theorem

A cost function of the form $\sum_{N \in T}(\text{Cut } N_L, N_R\ ) \cdot g(N_L, N_R)$ is admissible if and only if

(i) g is symmetric, *i.e.,* $g(|A|, |B|) = g(|B|, |A|)$

(ii) g is increasing, *i.e.,* $g(|A| + 1, |B|) \geq g(|A|, |B|)$

(iii) Every binary tree has same cost when the input is a unit weight clique

Theorem

A cost function of the form $\sum_{N \in T} (\text{Cut } N_L, N_R\ ) \cdot g(N_L, N_R)$ is admissible if and only if

(i) g is symmetric, *i.e.*, $g(|A|, |B|) = g(|B|, |A|)$

(ii) g is increasing, *i.e.*, $g(|A| + 1, |B|) \geq g(|A|, |B|)$

(iii) Every binary tree has same cost when the input is a unit weight clique

- Dasgupta's cost function is admissible

$$g(|A|, |B|) = |A| + |B|$$

- There is an entire family of cost functions that are admissible
- In some sense, Dasgupta's function is the most "natural"

Theorem

A cost function of the form $\sum_{N \in T} (\text{Cut } N_L, N_R\ ) \cdot g(N_L, N_R)$ is admissible if and only if

(i) g is symmetric, *i.e.*, $g(|A|, |B|) = g(|B|, |A|)$

(ii) g is increasing, *i.e.*, $g(|A| + 1, |B|) \geq g(|A|, |B|)$

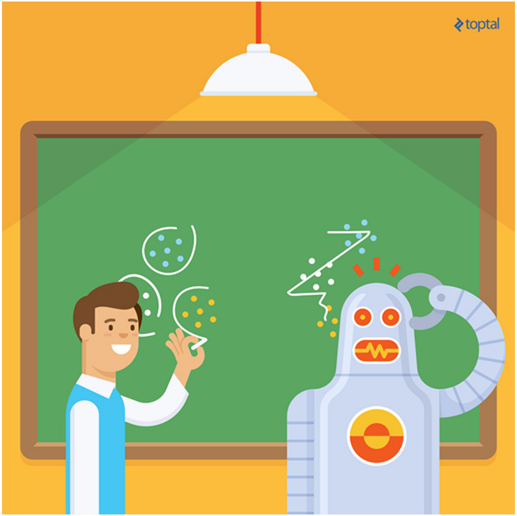
(iii) Every binary tree has same cost when the input is a unit weight clique

- Dasgupta's cost function is admissible

$$g(|A|, |B|) = |A| + |B|$$

- There is an entire family of cost functions that are admissible
- In some sense, Dasgupta's function is the most "natural"
- Rest of Talk: Focus on Dasgupta's cost function

Algorithms



Algorithms

In the worst case, most of the practical algorithms have bad approximation guarantees

In the worst case, most of the practical algorithms have bad approximation guarantees

Dasgupta showed his objective function is NP-hard

Algorithms

In the worst case, most of the practical algorithms have bad approximation guarantees

Dasgupta showed his objective function is NP-hard

Charikar and Chatziafratis and Roy and Pokutta showed that we cannot have constant approximation under the Small Set Expansion Hypothesis

Algorithms

In the worst case, most of the practical algorithms have bad approximation guarantees

Dasgupta showed his objective function is NP-hard

Charikar and Chatziafratis and Roy and Pokutta showed that we cannot have constant approximation under the Small Set Expansion Hypothesis

Solution 1: Find approximation algorithms
Solution 2: Beyond worst-case analysis

Hope: Recursive Sparsest Cut

Algorithm: Recursive Sparsest Cut

Input: Weighted graph $G = (V, E, w)$

$\{A, V \setminus A\} \leftarrow$ cut with sparsity $\leq \phi \cdot \min_{S \subseteq V} \dfrac{w(S, V \setminus S)}{|S| \cdot |V \setminus S|}$

Recurse on subgraphs $G[A]$, $G[V \setminus A]$ to obtain trees T_A, $T_{V \setminus A}$

Output: Return tree whose root has subtrees T_A, $T_{V \setminus A}$

Hope: Recursive Sparsest Cut

Algorithm: Recursive Sparsest Cut

Input: Weighted graph $G = (V, E, w)$

$\{A, V \setminus A\} \leftarrow$ cut with sparsity $\leq \phi \cdot \min_{S \subseteq V} \dfrac{w(S, V \setminus S)}{|S| \cdot |V \setminus S|}$

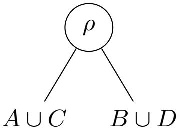
Recurse on subgraphs $G[A]$, $G[V \setminus A]$ to obtain trees T_A, $T_{V \setminus A}$

Output: Return tree whose root has subtrees T_A, $T_{V \setminus A}$

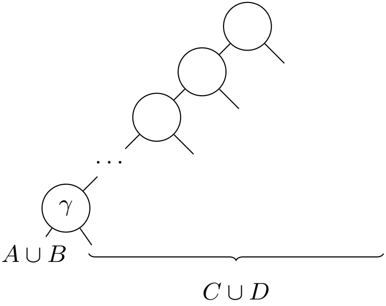
- For Dasgupta's cost function, $O(\log n \cdot \phi)$-approximation [Dasgupta '16]
- Current best known value for ϕ is $O(\sqrt{\log n})$ [ARV '09]
- We show $O(\phi)$-approximation (also independently [CC '17])

Proof Sketch

Tree T output by the algorithm



Optimal Tree T^*



$$\frac{w(A \cup C, B \cup D)}{|A \cup C| \cdot |B \cup D|} \leq \phi \frac{w(A \cup B, C \cup D)}{|A \cup B| \cdot |C \cup D|} = \Theta(\phi \cdot \frac{w(A \cup B, C \cup D)}{n^2})$$

$$\text{cost}(\rho) = (|A| + |B| + |C| + |D|) \cdot w(A \cup C, B \cup D) = n \cdot w(A \cup C, B \cup D)$$

$$\text{cost}(\gamma \text{ and ancestors}) \geq (|A| + |B|) \cdot w(A \cup B, C \cup D) \geq n/3 \cdot w(A \cup B, C \cup D)$$

Charge the cost of ρ to the edges of $(A \cup B, C \cup D)$

Proof Sketch

Lemma

The total charge (due to all nodes of T) for any edge (u, v) is at most $\frac{9}{2}\phi \min\{\frac{3}{2}|V(\mathrm{LCA}_{T^*}(u, v))|, n\}$

Proof by induction.

Proof Sketch

Lemma

The total charge (due to all nodes of T) for any edge (u, v) is at most $\frac{9}{2}\phi \min\{\frac{3}{2}|V(\mathrm{LCA}_{T^*}(u, v))|, n\}$

Proof by induction.

Lemma [Dasgupta '16]

For a tree T^*, $\mathrm{cost}(T^*) = \sum_{(u,v)\in E} w((u, v)) \cdot |V(\mathrm{LCA}_{T^*}(u, v))|$

Combining the two lemmas shows that the recursive sparsest cut gives an $O(\phi)$-approximation

- For worst case inputs, Recursive Sparsest Cut gives $O(\phi)$-approximation
- Assuming the "Small Set Expansion Hypothesis", no polytime $O(1)$-approx.

- For worst case inputs, Recursive Sparsest Cut gives $O(\phi)$-approximation
- Assuming the "Small Set Expansion Hypothesis", no polytime $O(1)$-approx.

Real-world graphs are often not worst-case

Hierarchical Clustering: Random Graph Models

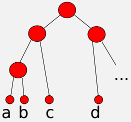
What is a reasonable model for real-world inputs?

Hierarchical Clustering: Random Graph Models

What is a reasonable model for real-world inputs?

In real world, inputs have some underlying, noisy ground-truth.

Generate graphs using ultrametrics:
Take an ultrametric,

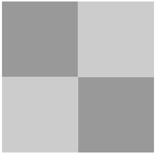


Generate an unweighted edge u, v with probability $f(\text{dist}(u, v))$ for some non-increasing function $f : \mathbb{R}_+ \mapsto [0, 1]$.

A generalization of the random graphs model for flat clustering

Flat Clustering

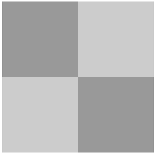
- Planted partition/block models
- Higher probability of edge between same part
- Lower probability of edge across different parts
- Adjacency matrix for graphs with 2 parts



A generalization of the random graphs model for flat clustering

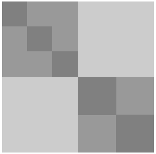
Flat Clustering

- Planted partition/block models
- Higher probability of edge between same part
- Lower probability of edge across different parts
- Adjacency matrix for graphs with 2 parts



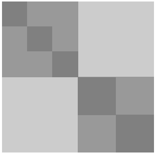
Hierarchical Clustering

- Planted hierarchy
- Higher probability of edge between nodes with deeper common ancestor
- Adjacency matrix for graphs with planted hierarchy



Hierarchical Clustering: Random Graph Models

- Random graphs with k-bottom level clusters (k can be function of n)
- Each bottom level cluster is sufficiently large
- Hidden (planted) hierarchical structure over the k bottom-level clusters



Can we identify a hierarchical cluster-tree that is an $O(1)$ or $(1 + \epsilon)$-approximation w.r.t. Dasgupta's cost function for such randomly generated graphs?

Spectral Algorithm for Planted (Flat) Clusters

Probability Matrix

$$\begin{bmatrix} 0.6 & 0.6 & 0.6 & 0.3 & 0.3 & 0.3 \\ 0.6 & 0.6 & 0.6 & 0.3 & 0.3 & 0.3 \\ 0.6 & 0.6 & 0.6 & 0.3 & 0.3 & 0.3 \\ 0.3 & 0.3 & 0.3 & 0.6 & 0.6 & 0.6 \\ 0.3 & 0.3 & 0.3 & 0.6 & 0.6 & 0.6 \\ 0.3 & 0.3 & 0.3 & 0.6 & 0.6 & 0.6 \end{bmatrix}$$

Adjacency Matrix

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

- Probability matrix is low rank; adjacency matrix (realized graph) may be full rank
- Projecting adjacency matrix onto top k (e.g., 2) singular vectors reveals planted partition

Spectral Algorithm: Random Hierarchical Graphs

Algorithm: Linkage++

Input: Graph $G = (V, E)$

- Project adjacency matrix A of G to top k- singular vectors to obtain $\mathbf{x}_i \in \mathbb{R}^k$ for every $i \in V$

- Perform single linkage on $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ using Euclidean distances in $\mathbb{R}^k$ until k clusters are obtained

- Perform single linkage on the k-clusters using edge density in G between these clusters

Output: Resulting hierarchical tree

Spectral Algorithm: Random Hierarchical Graphs

Theorem. Linkage++ Performance

Provided the following conditions hold:

- The smallest bottom-level cluster has $\tilde{\Omega}(\sqrt{n})$-nodes
- Each probability is $\omega(\sqrt{\log n/n})$

Then the Linkage++ outputs a tree with cost at most $(1+\epsilon)\text{OPT}$ with respect to the Dasgupta cost function with probability at least $1-o(1)$.

Spectral Algorithm: Random Hierarchical Graphs

Theorem. Linkage++ Performance

Provided the following conditions hold:

- The smallest bottom-level cluster has $\tilde{\Omega}(\sqrt{n})$-nodes
- Each probability is $\omega(\sqrt{\log n / n})$

Then the Linkage++ outputs a tree with cost at most $(1 + \epsilon)\text{OPT}$ with respect to the Dasgupta cost function with probability at least $1 - o(1)$.

- Proof involves results from McSherry (2001) combined with analysis of linkage algorithms
- Different algorithm using semi-definite programming extends to wider ranges of semi-random graph models

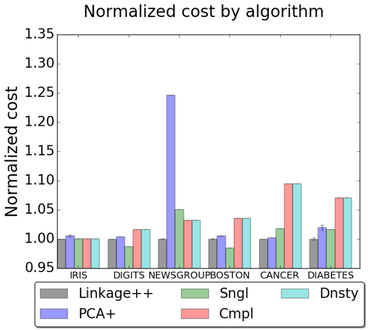
Back to practice

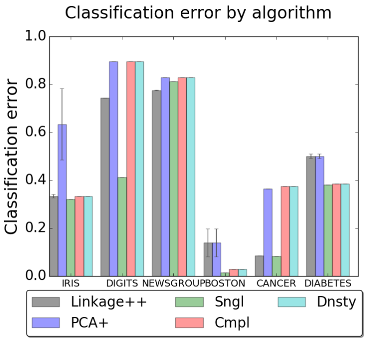
Evaluation of algorithms on synthetic (planted hierarchical random graphs) and a few UCI datasets

Report Dasgupta cost and classification error for various algorithms

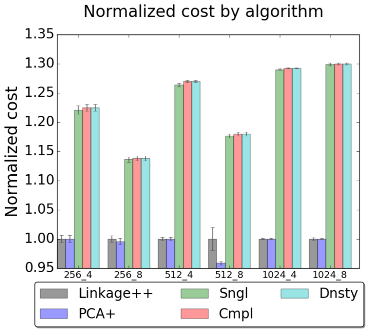
- Linkage++
- PCA+ (perform PCA and then average linkage)
- Sngl (Single linkage directly on graph)
- Cmpl (Complete linkage directly on graph)
- Dnsty (Average linkage directly on graph)

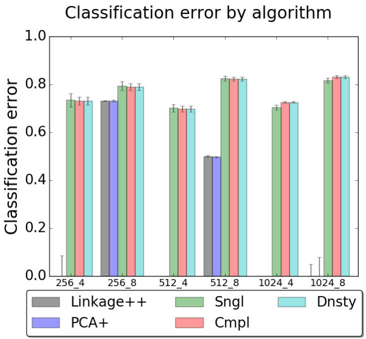
Experimental Results: UCI Datasets





Experimental Results: Synthetic Data





Conclusion

- Hierarchical clustering is a fundamental problem in data analysis that has mainly been studied through procedures rather than as an optimization problem
- Axiomatic study of admissible cost functions, provides a way to analyse quantitatively the performance of algorithms
- Efficient approximation algorithm for Dasgupta's cost function based on recursive sparsest-cut. Cannot get constant factor assuming SSEH.
- Beyond worst-case analyis:
  - Random graphs with planted hierarchies
  - Linkage++ (Spectral methods + linkage algorithms) gives $(1+\epsilon)$-approximation with high probability and efficient in practice

Open Questions

- Open Question: Improve the definition of real-world inputs for hierarchical clustering (maybe based on the stability conditions for flat clustering)
- Open Question: (semi-)streaming algorithms for real-world inputs