Scalable Algorithms for Correlation Clustering on Large Graphs

Nathan Cordner

Boston University

13 April 2023

Outline

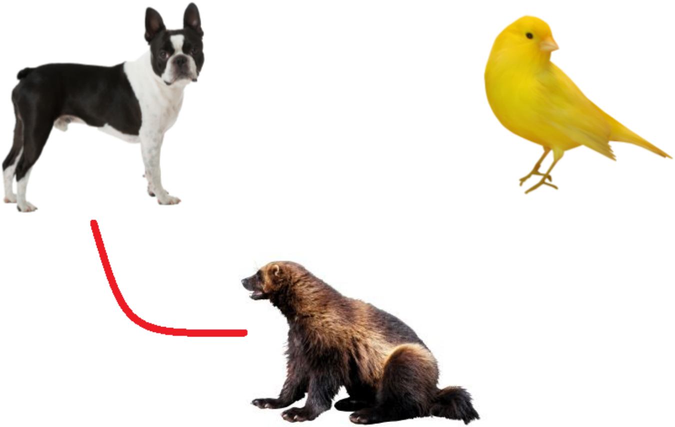
- **Correlation Clustering Introduction**
- Scalable Algorithms for
  - Cluster Improvement
  - Constrained Cluster Sizes
  - Constrained Number of Clusters
  - Consensus Clustering

Clustering Problem

Which animals belong together?

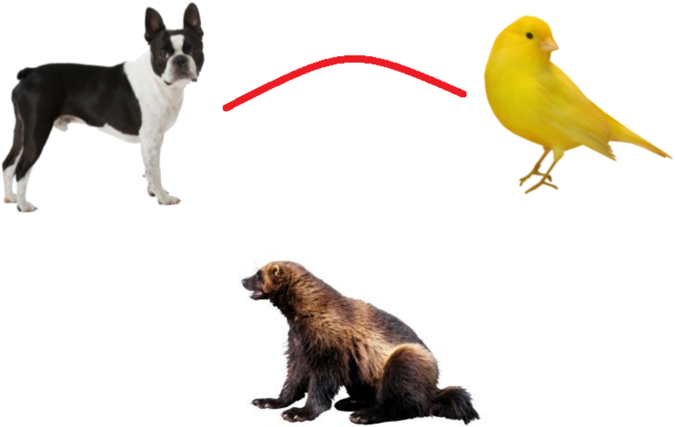
Clustering Problem

Which animals belong together?



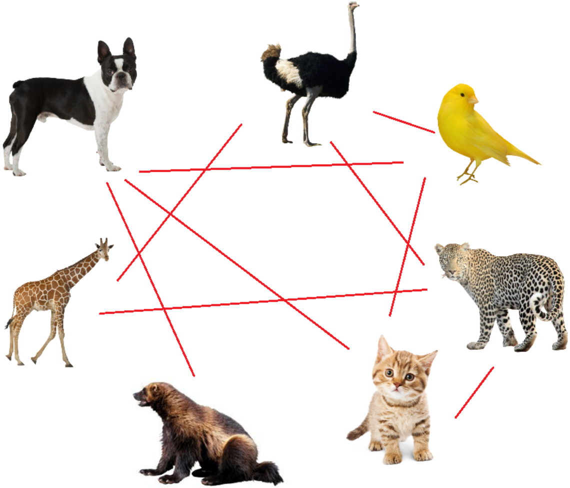
Clustering Problem

Which animals belong together?



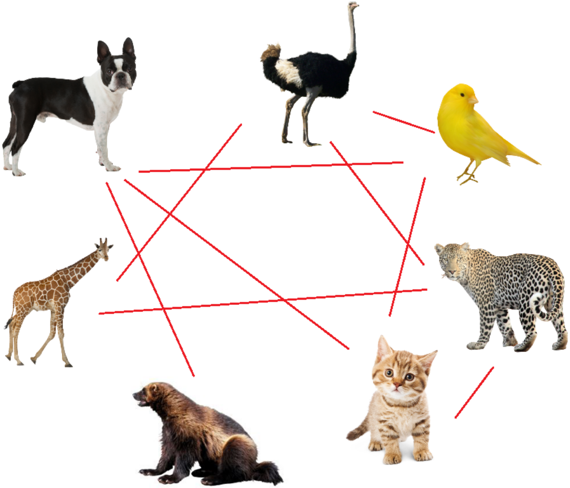
Clustering Problem

Which animals belong together?



Clustering Problem

Which animals belong together?



Correlation Clustering [BBC04]

Given a complete graph $G = (V, E)$

- $E = E^+ \cup E^-$

Want to cluster $+$ edges and separate $-$ edges

- Maximize Agreements
- **Minimize Disagreements**

Some Applications

- Classification
- Entity Resolution
- Communities in Social Networks

The Pivot Algorithm [ACN08, AL09]

Neighborhood Oracle $N(u) = \{v \in V \mid \{u, v\} \in E^+\}$

Pivot$(V, E = E^+ \cup E^-)$:

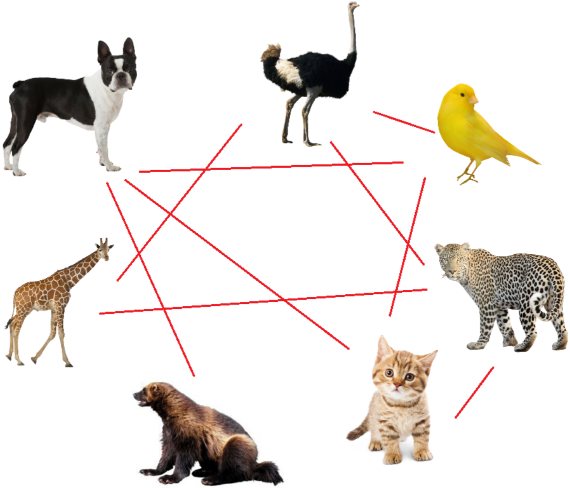
- Pick random pivot node $u \in V$
- Set $C = \{u\}$
- For all $v \in N(u)$:
  - If $v \in V$: Add v to C
- Repeat on $V = V \setminus C$ until empty
- Return completed clustering

Runs in $O(|V| + |E|^+)$ time

Randomized expected 3-approximation

The Pivot Algorithm [ACN08, AL09]

Example:



Other Linear Time Methods

The Vote Algorithm [ES09]

- Pick unclustered nodes one at a time
  - First node creates its own cluster
  - All others: add to existing cluster, or create own
  - Greedily minimize increase in clustering cost
- Runs in $\Theta(|V| + |E|^+)$ time

LocalSearch: given a clustering,

- Each node decides whether to stay or move clusters
- Iterate until improvements converge
- Each *iteration* is $\Theta(|V| + |E|^+)$

LP Methods

Linear Program for Correlation Clustering:

$$\min \sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\notin E^+} (1 - x_{uv})$$

$$\begin{aligned} x_{uv} + x_{vw} &\geq x_{uw} && \text{for all } u, v, w \in V \\ x_{uv} &\in [0, 1] && \text{for all } u, v \in V \end{aligned}$$

- $O(|V|^2)$ variables, $O(|V|^3)$ constraints!

Rounding Methods:

- 2.5-approx [ACN08]
- 2.06-approx [CMSY15]
- $(1.994+\epsilon)$-approx [CLN22]

One Algorithm to Rule Them All?

Pivot has been successfully used to cluster

- Social network graphs [KPT11]
- Protein-protein interaction graphs [KPT11; HWH15]
- Event graphs generated from news stories [CMB17]

Pivot has been adapted for

- Probabilistic graphs [KPT11; MTG20]
- Chromatic correlation clustering [KSZC21]
- Fair correlation clustering [AEKM20]
- Data streaming and online settings [ACGM15; LMVW21]
- Query constraints [GKBT20]
- Cluster size constraints [PM15]

Deterministic and parallel versions [ZW09; CDK14; PORJ15]

Motivating Questions

- How do Pivot and the other linear time algorithms perform when tested against slower algorithms with better approximation guarantees?
- Can we boost Pivot's clustering quality in various settings without diminishing its run time advantages?
- What practical improvements can we make for CC algorithms applied to consensus clustering?

Contributions

How do Pivot and the other linear time algorithms perform when tested against slower algorithms with better approximation guarantees?

- We show experimentally that Pivot, Pivot with LocalSearch, and Vote perform well when compared against state-of-the-art algorithms
  - Clustering costs are close to, or even lower than, state-of-the-art algorithms
  - Running times are much, much quicker!
- We also show that adaptations of these algorithms perform well in other CC settings

Contributions

Can we boost Pivot's clustering quality in various settings without diminishing its run time advantages?

- We develop a lightweight LocalSearch method (InnerLocalSearch) that show experimentally converges much faster than LocalSearch while still providing a significant reduction in clustering cost
- We also demonstrate InnerLocalSearch's applicability to constrained cluster sizes and the related consensus clustering problem

Contributions

What practical improvements can we make for CC algorithms applied to consensus clustering?

- We develop a memory-efficient implementation of Pivot and other CC algorithms to use on larger graphs
- We also show a clustering-sampling method that improves running time while only incurring small increases of clustering cost

Outline

- ~~Correlation Clustering Introduction~~
- Scalable Algorithms for
  - **Cluster Improvement**
  - Constrained Cluster Sizes
  - Constrained Number of Clusters
  - Consensus Clustering

Goals:

- Compare Pivot, LS, and Vote with state-of-the-art methods
- Develop the InnerLocalSearch method and show its competitiveness with LS on larger data sets

Pivot Comparison

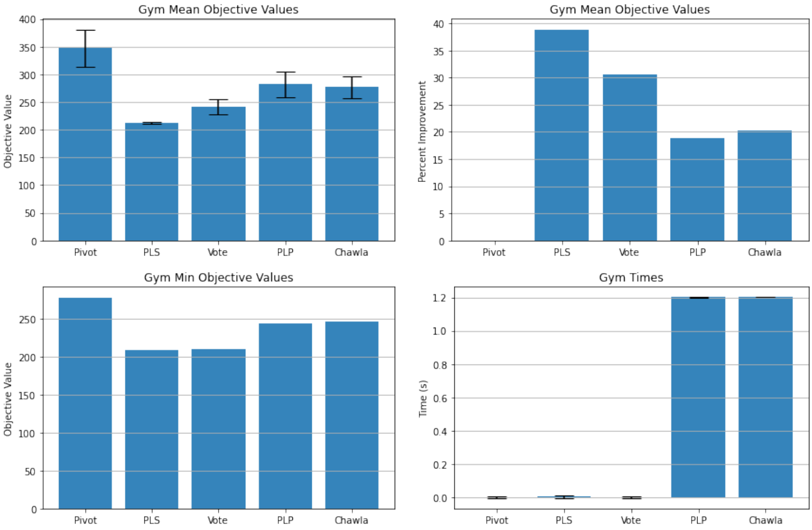
How does Pivot compare to the state-of-the-art approximation algorithms?

Methods tested:

- **Pivot** and **Vote**
- **PLS**: Pivot with LocalSearch
- **PLP**: 2.5-approx LP rounding [ACN08]
- **Chawla**: 2.06-approx LP rounding [CMSY15]

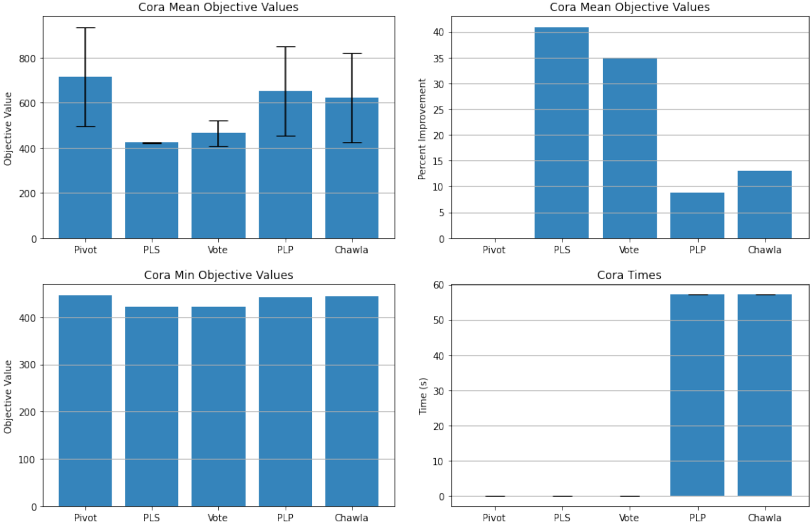
Pivot Comparison

Gym: $|V| = 94$, $|E|^+ = 465$



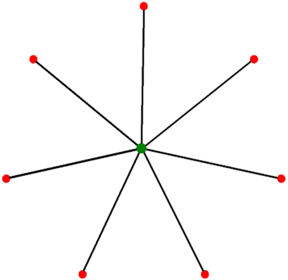
Pivot Comparison

Cora200: $|V| = 190$, $|E|^+ = 1,588$



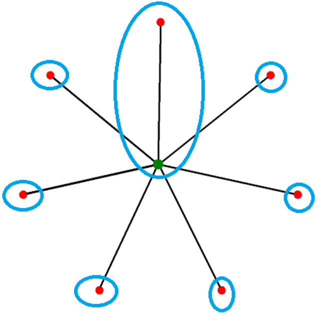
Drawbacks of Pivot

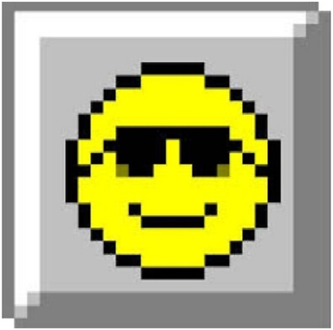
Pivot can form sparse, star-shaped clusters



Drawbacks of Pivot

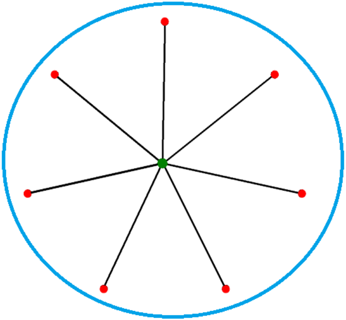
Pivot can form sparse, star-shaped clusters





Drawbacks of Pivot

Pivot can form sparse, star-shaped clusters



Can we do better and still maintain scalability?

Cluster Improvement

LocalSearch: given a clustering,

- Each node decides whether to stay or move clusters
- Iterate until improvements converge
- Slow: each iteration is $\Theta(|V| + |E|^+)$
- Somewhat popular though [MTG20; AEKM20]

New Idea: run LS *inside* clusters only

- Will not generate same level of improvement as a full LS, but will converge much faster

InnerLocalSearch

InnerLocalSearch: on graph G

- Obtain clusters $C_1, \ldots, C_k$ from Pivot(G)
- Let G_i be the graph induced by C_i
- Return LocalSearch$(G_1), \ldots,$ LocalSearch(G_k)

Properties

- Iteration time: $O(\min\{|V|d, |V| + |E|^+\})$
  - d is size of largest Pivot cluster
  - Tends to converge much faster than LS
- Easily run in parallel
- Improves cluster costs from Pivot (nearly like LS)
- Immediately applies to CC generalizations

LocalSearch Comparison

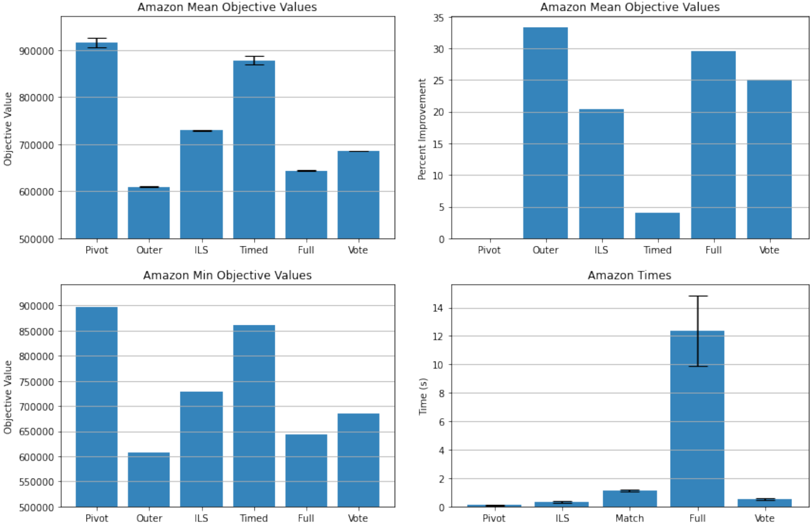
How does ILS compare to LS and other methods?

Methods tested:

- **Pivot**, **ILS**, and **Vote**
- **Outer**: inter-cluster cost from Pivot result (benchmark for ILS improvement)
- **Timed**: Pivot with LocalSearch, using time limit set by ILS convergence
- **Match**: time required for Pivot with LocalSearch to reach same level of improvement as ILS
- **Full**: Pivot with LocalSearch run to convergence

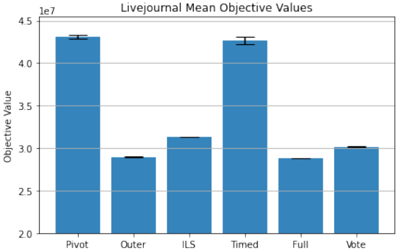
LocalSearch Comparison

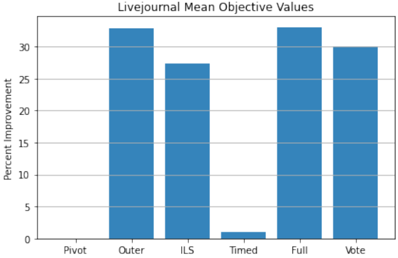
Amazon: $|V| = 334,863$, $|E|^+ = 925,872$, $d = 107.1$

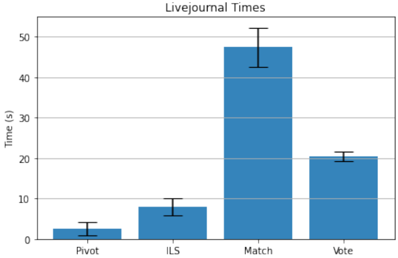


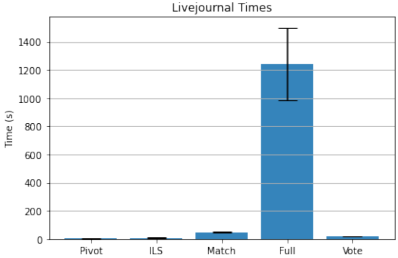
LocalSearch Comparison

Livejournal: $|V| = 3,997,962$, $|E|^+ = 34,681,189$, $d = 717.9$



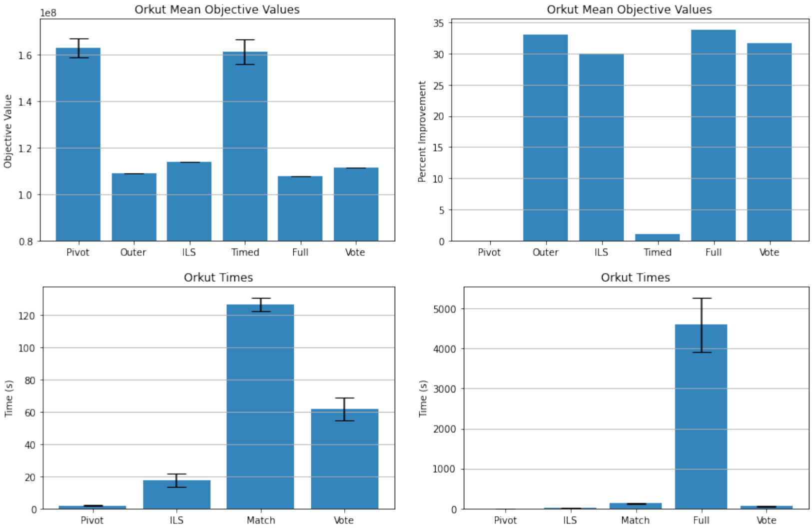






LocalSearch Comparison

Orkut: $|V| = 3,072,441$, $|E|^+ = 117,185,083$, $d = 1679.8$



Outline

- ~~Correlation Clustering Introduction~~
- Scalable Algorithms for
  - ~~Cluster Improvement~~
  - **Constrained Cluster Sizes**
  - Constrained Number of Clusters
  - Consensus Clustering

Goals:

- Compare Pivot, LS, and Vote with state-of-the-art methods
- Compare performance of Pivot, ILS, LS, and Vote on larger data sets

Constrained Cluster Sizes

Two kinds of constraints:

- **Uniform** [PM15]: given $K \geq 1$, all clusters must have size $\leq K$
- **Non-Uniform** [JXLW20]: every node v can only be in a cluster of size at most K_v

Soft constraints: some violations of clusters sizes allowed

Hard constraints: all size constraints must be observed

Uniform Constrained Cluster Sizes

Linear Program for Uniform Size-Constrained CC:

$$\min \sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\notin E^+} (1 - x_{uv})$$

$$\begin{aligned} x_{uv} + x_{vw} &\geq x_{uw} && \text{for all } u, v, w \in V \\ \sum_{v\in V}(1 - x_{uv}) &\leq K && \text{for all } u \in V, \\ x_{uv} &\in [0, 1] && \text{for all } u, v \in V \end{aligned}$$

LP rounding algorithms

- 6-approx [PM15]
- 5.37-approx [JCTZ21]

Uniform Constrained Cluster Sizes

Pivot adaptations [PM15]

- 7-approx by removing a smallest set of $+$ edges
- 11-approx for random removal

Our Approach:

- **Pivot**: pivot node chooses $K - 1$ neighbors at random if full Pivot cluster is too large
- **Vote** and **LocalSearch**: only add nodes to clusters that are not yet at capacity
- **InnerLocalSearch**: no changes!

Pivot Comparison

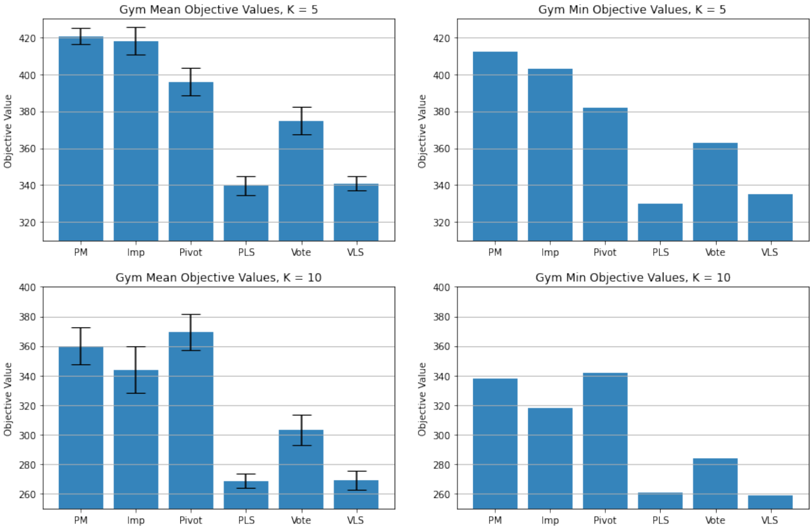
How does Pivot compare to the state-of-the-art approximation algorithms?

Methods tested:

- **Pivot** and **Vote**
- **PLS**: Pivot with LocalSearch
- **VLS**: Vote with LocalSearch
- **PM**: 6-approx LP rounding [PM15]
- **Imp**: 5.37-approx LP rounding [JCTZ21]

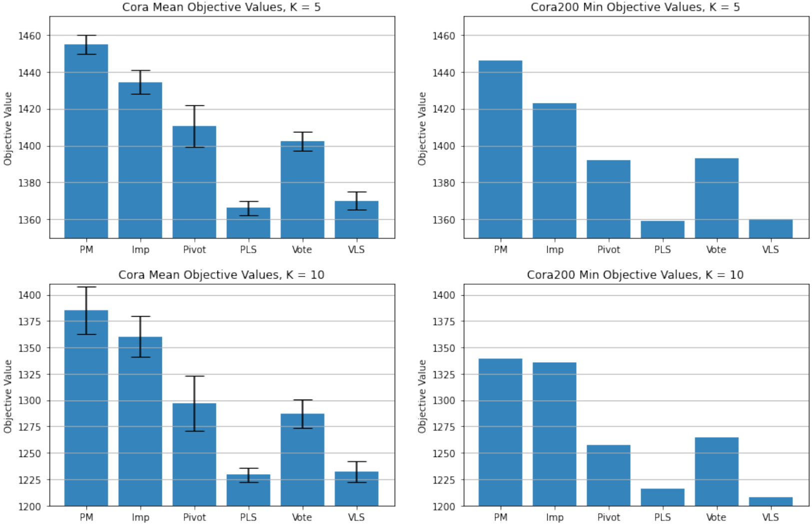
Pivot Comparison

Gym: $|V| = 94$, $|E|^{+} = 465$



Pivot Comparison

Cora200: $|V| = 190,\ |E|^{+} = 1,588$



LocalSearch Comparison

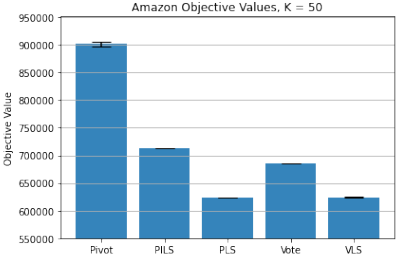
How does ILS compare to LS and other methods?

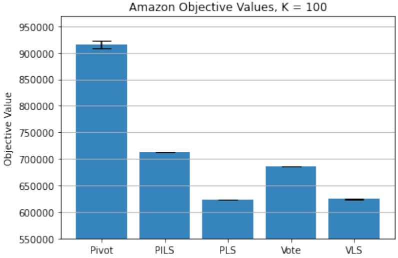
Methods tested:

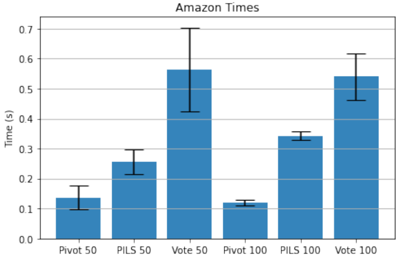
- **Pivot**, **ILS**, and **Vote**
- **PLS**: Pivot with LocalSearch (5-minute time limit)
- **VLS**: Vote with LocalSearch (5-minute time limit)

LocalSearch Comparison

Amazon: $|V| = 334,863$, $|E|^+ = 925,872$

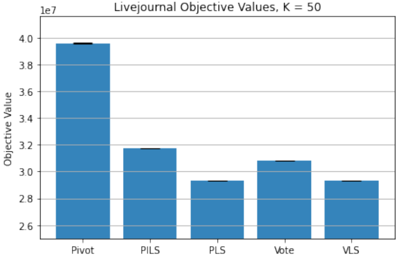


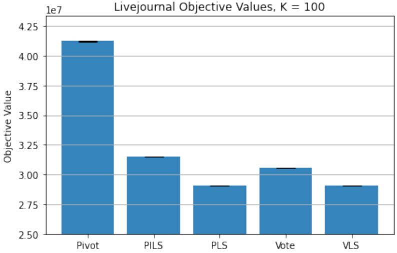


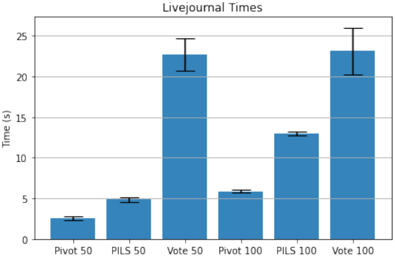


LocalSearch Comparison

Livejournal: $|V| = 3,997,962$, $|E|^+ = 34,681,189$

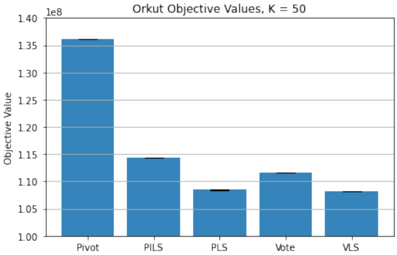


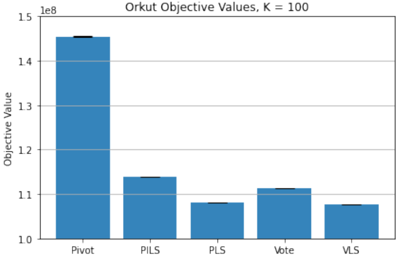


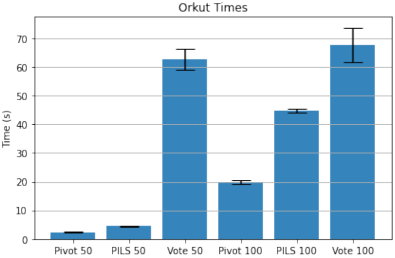


LocalSearch Comparison

Orkut: $|V| = 3,072,441$, $|E|^+ = 117,185,083$







Non-Uniform Constrained Cluster Sizes

Linear Program for Non-Uniform Size-Constrained CC:

$$
\begin{aligned}
\min \sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\notin E^+} (1 - x_{uv}) \\
x_{uv} + x_{vw} &\geq x_{uw} && \text{for all } u, v, w \in V \\
\sum_{v\in V} (1 - x_{uv}) &\leq K_u && \text{for all } u \in V, \\
x_{uv} &\in [0, 1] && \text{for all } u, v \in V
\end{aligned}
$$

LP rounding algorithms [JXLW20]

- Soft: $(2/\alpha)$-approximation where every cluster satisfies $|C| \leq 1/(1-\alpha)\min_{u\in C}\{K_u\}$, $\alpha \in (0, 1/2]$
- Hard: $2K$-approximation where $K = \max_{u\in V}\{K_u\}$

Non-Uniform Constrained Cluster Sizes

Our Approach:

- **Pivot**: set initial size bound of new cluster equal to pivot node; include neighbors at random, adjusting size bound after each one and rejecting neighbors once size limit is reached
- **Vote** and **LocalSearch**: only add nodes to clusters that are not yet at capacity, adjusting capacity as needed
  - Ordered Vote first sorts nodes by increasing size of K_u
  - LocalSearch also maintains (min) priority queue of size constraints for each cluster in order to update size constraints quickly when nodes leave
- **InnerLocalSearch**: no changes!

Pivot Comparison

How does Pivot compare to the state-of-the-art approximation algorithms?

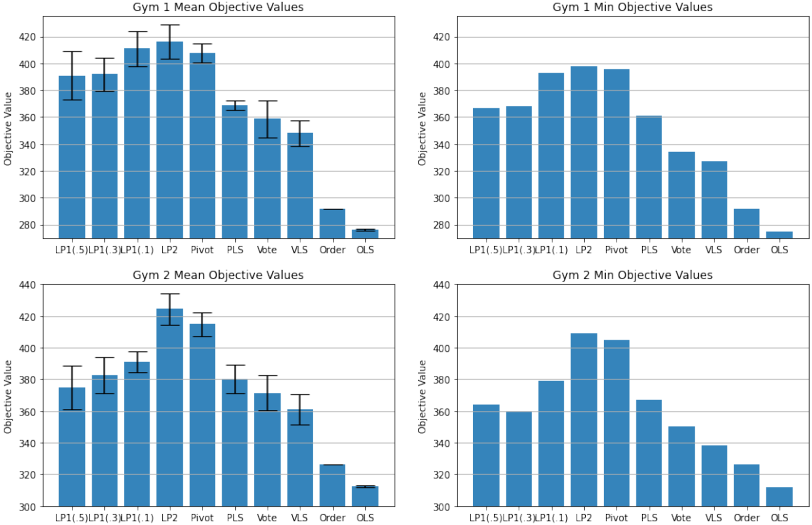
Methods tested:

- **Pivot**, **Vote**, and Ordered Vote (**Order**)
- **PLS**, **VLS**, **OLS**: Pivot, Vote, and Ordered Vote with LocalSearch
- **LP1(α)**: Soft constraint $(2/\alpha)$-approx LP rounding
- **LP2**: Hard constraint $2K$-approx LP rounding

Size bounds: randomly chosen from 1 to $|N(v)| + 1$ for each node v

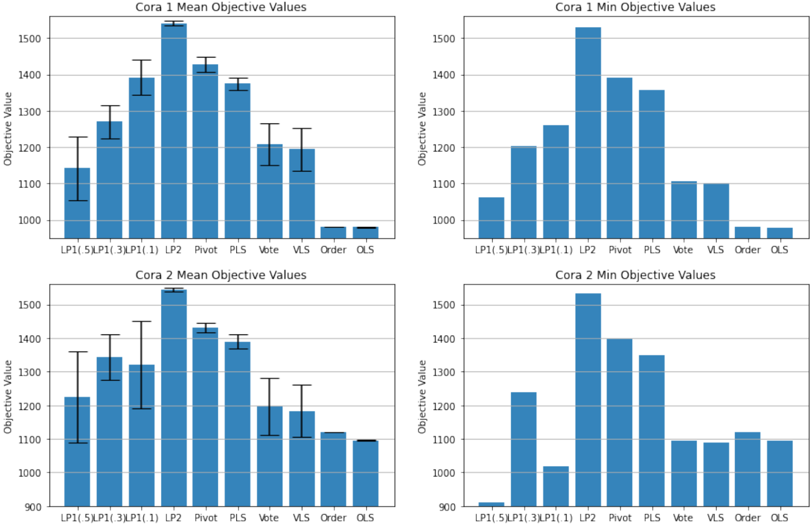
Pivot Comparison

Gym: $|V| = 94$, $|E|^+ = 465$



Pivot Comparison

Cora200: $|V| = 190,\ |E|^+ = 1,588$



LocalSearch Comparison

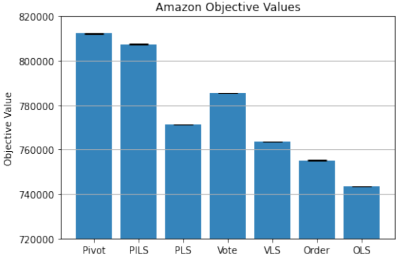
How does ILS compare to LS and other methods?

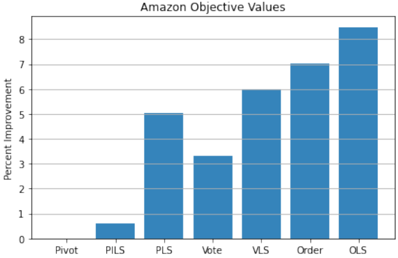
Methods tested:

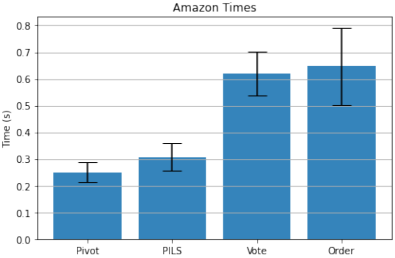
- **Pivot**, **Vote**, and Ordered Vote (**Order**)
- **PLS**, **VLS**, **OLS**: Pivot, Vote, and Ordered Vote with LocalSearch (5-minute time limit)
- **PILS**: Pivot with InnerLocalSearch

LocalSearch Comparison

Amazon: $|V| = 334,863$, $|E|^+ = 925,872$

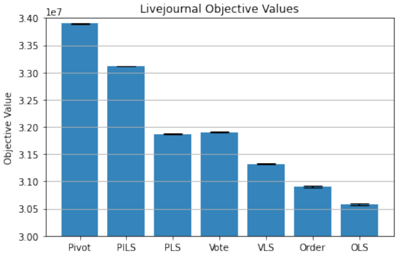


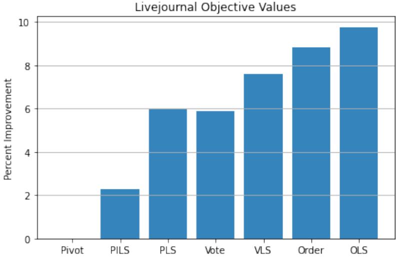


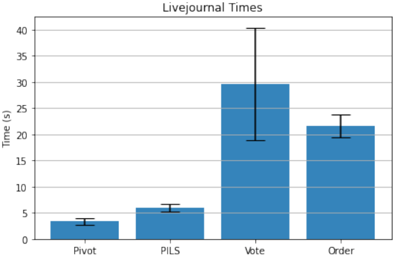


LocalSearch Comparison

Livejournal: $|V| = 3,997,962$, $|E|^+ = 34,681,189$

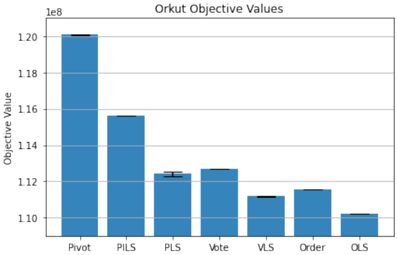


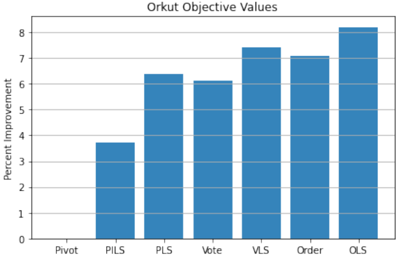


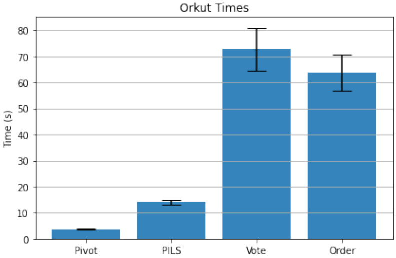


LocalSearch Comparison

Orkut: $|V| = 3,072,441,\ |E|^+ = 117,185,083$







Outline

- ~~Correlation Clustering Introduction~~
- Scalable Algorithms for
  - ~~Cluster Improvement~~
  - ~~Constrained Cluster Sizes~~
  - **Constrained Number of Clusters**
  - Consensus Clustering

Goals:

- Adapt Pivot and Vote methods to work in new constrained setting
- Compare Pivot, LS, and Vote with state-of-the-art methods
- Compare performance of Pivot, LS, and Vote on larger data sets

Constrained Number of Clusters

Minimize cost with clustering of size $\leq K$

- $K = 2$:
  - Pivot-like 3-approximation [BBC04]
  - LocalSearch 2-approximation [CSW08]
- General case:
  - PTAS [GG06]
  - Improved PTAS [KS09]
  - K-approximation [IN16]
  - K-LocalSearch heuristic [C17, TCD19]
- Proposed Algorithms:
  - K-Pivot, K-Vote
  - Blend (start with Pivot, finish with Vote)

Previous Methods

PTAS approaches:

- $(1+\epsilon)$-approximation factor
- Giotis, Guruswami: $|V|^{O(9^K/\epsilon^2)} \log |V|$ running time
- Karpinski, Schudy: $|V|^2 2^{O((K^6 \log K)/\epsilon^2)}$ running time
- Both methods rely on brute-force searches to identify best possible clusterings on large $(O(\log |V|), O(K^4 \log K))$ subsets of nodes

K-LocalSearch

- Randomly assign nodes to one of K clusters
- Run LocalSearch until convergence

Previous Methods

Bansal et al. 3-approximation $(K = 2)$:

- For every node, generate one Pivot cluster and assign remaining nodes to a second cluster
- Return 2-clustering with minimum cost

Il'ev and Navrotskaya K-approximation:

- For every node, generate $K - 1$ Pivot clusters and assign remaining nodes to cluster K
- Run LocalSearch to convergence on each clustering and return the one with minimum cost
- Running time: $O(|V|^3 I)$ where I is the largest number of iterations required by LS

Proposed Methods

K-Pivot:

- Form K Pivot clusters
- Merge additional Pivot clusters to the *current smallest* cluster

K-Vote:

- Run Vote until K clusters are formed
- Continue assigning nodes to existing clusters that minimize increase of clustering cost

Blend:

- Form K Pivot clusters
- Continue assigning nodes to existing clusters that minimize increase of clustering cost

Implementation Notes

Priority Queues

- Once K clusters are formed, we initialize a PQ to track cluster sizes
- K-Pivot merges new clusters to current smallest
- K-Vote and K-LS methods can either merge a node u to a cluster that contains $v \in N(u)$ or to the current smallest cluster
- $O(|V| \log K + |E|^+)$ running time

Pivot Comparison

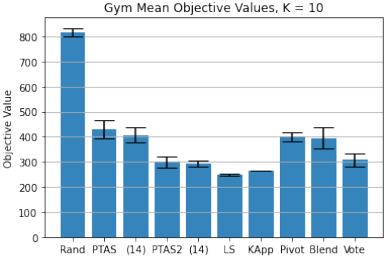
How does Pivot compare to the state-of-the-art approximation algorithms?

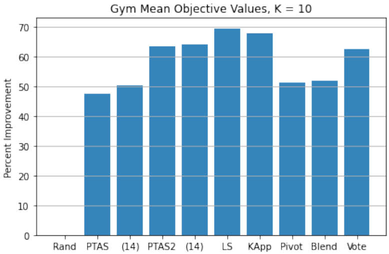
Methods tested:

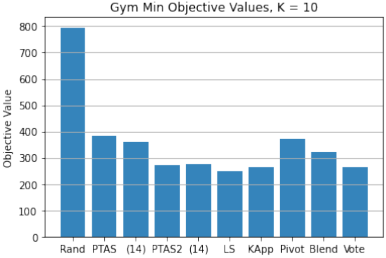
- **Rand**: random K-clustering, used as baseline
- **PTAS**: the Giotis and Guruswami PTAS
  - Sample size limited to 13 (14) nodes
- **PTAS2**: the Karpinski and Schudy PTAS
  - Sample size limited to 13 (14) nodes
- **KApp**: the Il'ev and Navrotskaya K-approximation
- **Pivot**, **Vote**, **Blend**, and **LS**

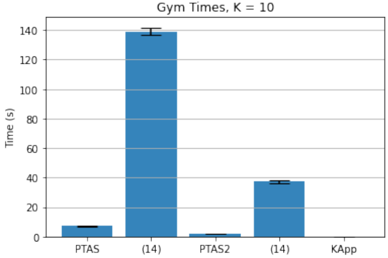
Pivot Comparison

Gym: $|V| = 94$, $|E|^+ = 465$



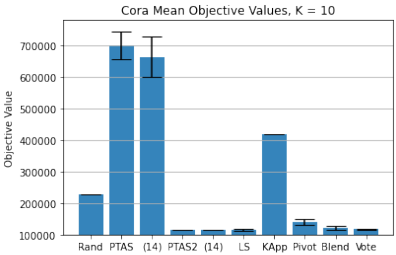


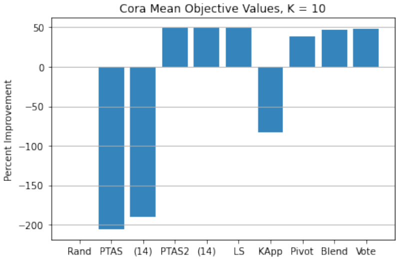


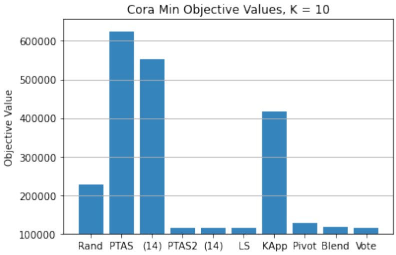


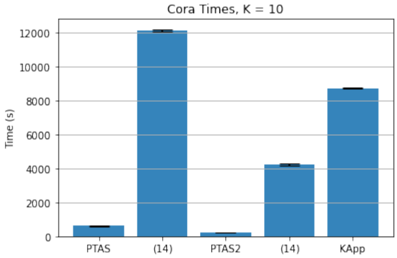
Pivot Comparison

Cora: $|V| = 1,879,\ |E|^+ = 64,955$









LocalSearch Comparison

How do the baseline methods compare against each other before and after LS improvements?

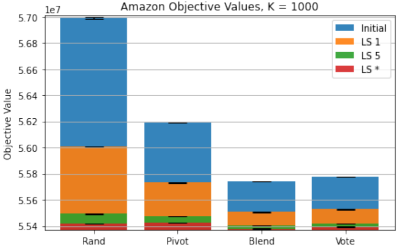
Methods tested:

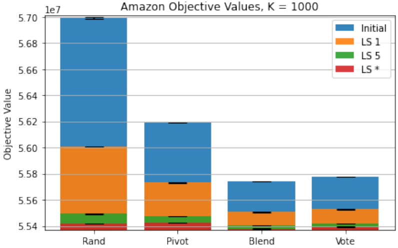
- **Pivot**, **Vote**, **Blend**, and **Rand**
- **LS** is applied to each baseline result with various time constraints (LS * runs to convergence)

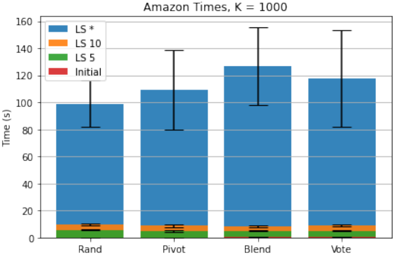
Note that ILS does not work here!

LocalSearch Comparison

Amazon: $|V| = 334,863$, $|E|^+ = 925,872$

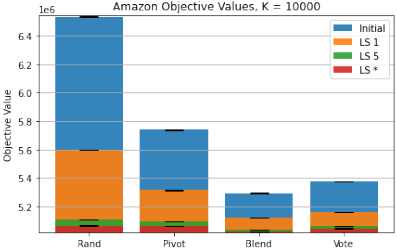


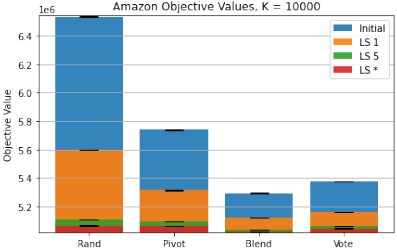


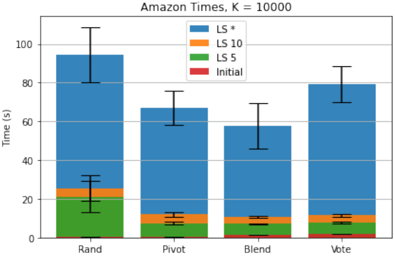


LocalSearch Comparison

Amazon: $|V| = 334,863$, $|E|^+ = 925,872$

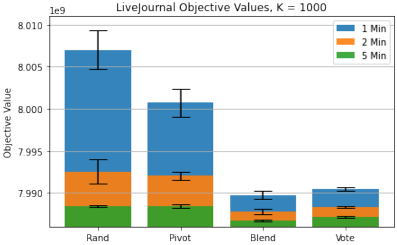


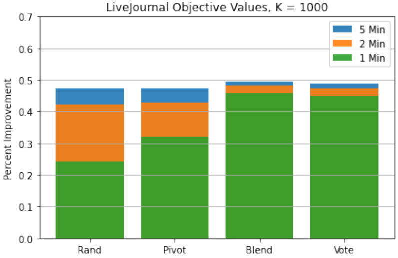


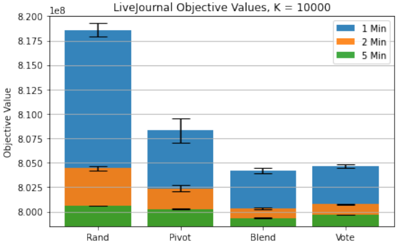


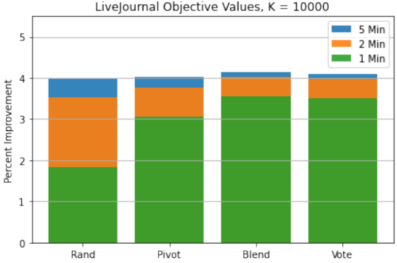
LocalSearch Comparison

Livejournal: $|V| = 3,997,962,\ |E|^+ = 34,681,189$



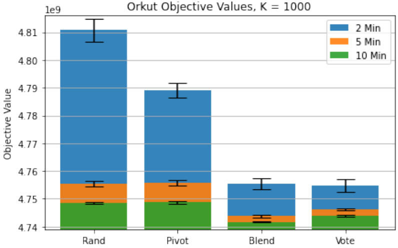


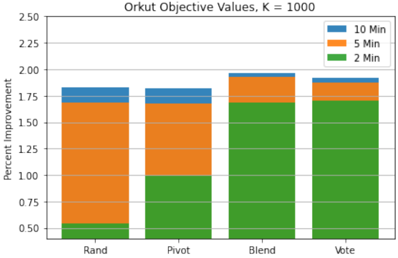


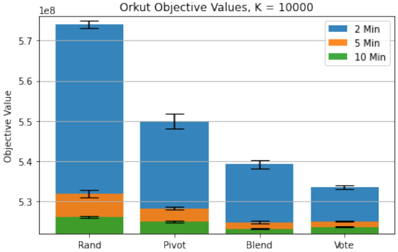


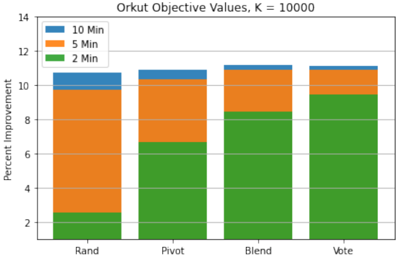
LocalSearch Comparison

Orkut: $|V| = 3,072,441$, $|E|^+ = 117,185,083$









Outline

- ~~Correlation Clustering Introduction~~
- Scalable Algorithms for
  - ~~Cluster Improvement~~
  - ~~Constrained Cluster Sizes~~
  - ~~Constrained Number of Clusters~~
  - **Consensus Clustering**

Goals:

- Develop a memory-efficient implementation of Pivot and other CC algorithms for larger graphs
- Demonstrate a clustering-sampling method to improve running time without too much impact on clustering cost

Weighted CC [BBC04, ACN08]

Every pair of nodes u, v has weights $w_{uv}^{+}, w_{uv}^{-} \geq 0$

- Clustering Cost:

$$\sum_{u,v \text{ in different clusters}} w_{uv}^{+} + \sum_{u,v \text{ in same cluster}} w_{uv}^{-}$$

Given $G = (V, E, w)$

- Form the unweighted *majority instance* G_w
  - Place $\{u, v\}$ in E_w^{+} if $w_{uv}^{+} > w_{uv}^{-}$
  - Place $\{u, v\}$ in E_w^{-} if $w_{uv}^{-} > w_{uv}^{+}$
  - Break ties arbitrarily
- Run Pivot on $G_w = (V, E_w = E_w^{+} \cup E_w^{-})$

Weighted CC [ACN08, KPT11]

Probability Constraints: $w^+_{uv} + w^-_{uv} = 1$

- Notation: $p(u,v) = w^+_{uv}$, $1 - p(u,v) = w^-_{uv}$

Relation to original CC problem

- $\{u,v\} \in E^+ \Leftrightarrow p(u,v) = 1$
- $\{u,v\} \in E^- \Leftrightarrow p(u,v) = 0$

Pivot Approximation Results

- 5-approx with probability constraints
- 2-approx with PC and the triangle inequality

$$w^-_{uv} \leq w^-_{ux} + w^-_{xv}$$

Consensus Clustering

Given clusterings $\mathcal{C}_1 \ldots, \mathcal{C}_k$ of node set V

- Find clustering $\mathcal{C}$ minimizing $\sum_{i=1}^{k} \text{Disagree}(\mathcal{C}, \mathcal{C}_i)$
- $\text{Disagree}(\mathcal{C}, \mathcal{C}_i) =$ number of node pairs (u, v) clustered together in only one input clustering

Relation to Correlation Clustering

- $p(u, v) =$ number of input clusterings where u, v are clustered together divided by k
- Edge weights $(1 - p)$ satisfy the triangle inequality

Consensus Clustering

Previous Pivot Problems [ACN08; GF08]

- Time inefficiency: $O(k|V|^2)$ to compute all edges
- Space inefficiency: $O(|V|^2)$ to store all edges

Improvement # 1: only compute edges as needed

- Precompute cluster labels for each node

Improvement # 2: reduce number of input clusterings

- Picking one input clustering at random: 2-approx
- Pivot on full set of inputs: 1.57-approx [ACN08]
- What about in between?

Consensus Clustering

Improvement 1 Examples

Mushrooms

- 22 input clusterings, 8124 nodes
- Edges: 57.59 s; average Pivot run: 0.0082 s
- Labels: 0.029 s; average Pivot run: 0.0129 s

Facebook Government

- 100 input clusterings, 7057 nodes
- Edges: 272.52 s; average Pivot run: 0.054 s
- Labels: 0.121 s; average Pivot run: 144.85 s

Consensus Clustering

Sampling input clusterings

- Assume k large and sample $R < k$ input clusterings
- $p =$ true probability u, v are clustered together (assume $p < 1/2$)
- Let $X =$ number of sampled clusters where u, v are clustered together
- Model X as a Binomial rv with R trials and success probability p
- Pivot algorithm "makes a mistake" when $X > R/2$

Consensus Clustering

What is $\mathbb{P}(X > R/2)$?

- Normalize: $Z = (X - pR)/\sqrt{Rp(1-p)}$
- Estimate $\mathbb{P}(X > R/2)$ using

$$\mathbb{P}\left(Z > \frac{R/2 - pR}{\sqrt{Rp(1-p)}}\right) = \mathbb{P}\left(Z > \frac{\sqrt{R}(1/2 - p)}{\sqrt{p(1-p)}}\right)$$

- Let $f(R,p) = \sqrt{R}(1/2 - p)/\sqrt{p(1-p)}$
- Find $\mathbb{P}(Z > f(R,p))$ by evaluating

$$\mathsf{Err}(R,p) := 1 - \Phi(f(R,p)),$$

where Φ is the standard normal CDF

Consensus Clustering

Lemma: the expected cost multiple of edge (u, v) due to error in a Pivot clustering is

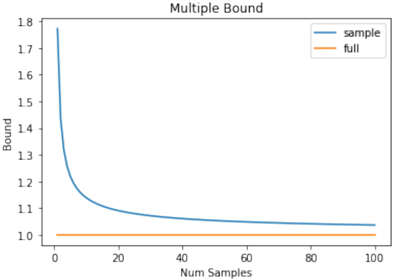
$$p \cdot (1 - \mathsf{Err}(R, p)) + (1 - p) \cdot \mathsf{Err}(R, p)$$

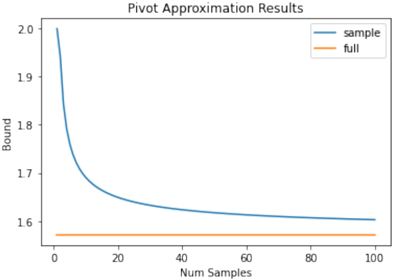
Cost multiple upper bound:

$$g(R) = \max_{p \in [0,1/2]} \{[p \cdot (1 - \mathsf{Err}(R, p)) + (1 - p) \cdot \mathsf{Err}(R, p)]/p\}$$

Consensus Clustering

Theorem: Pivot is a $((6g(R) + 5)/7)$-approx algorithm





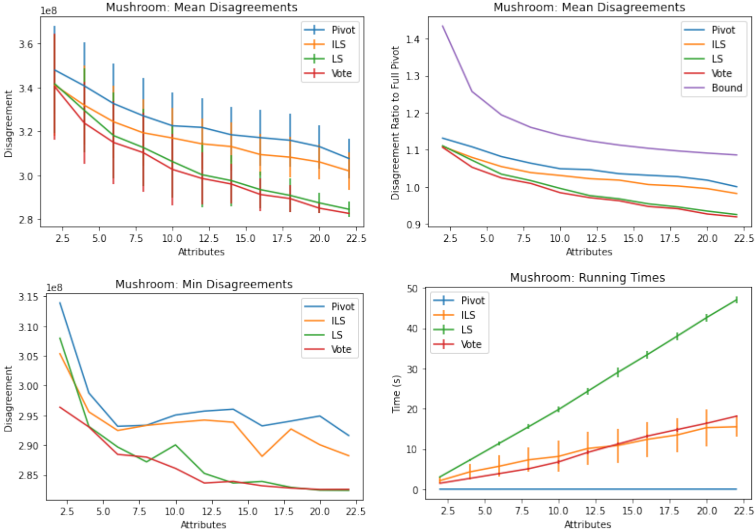
Experiments

Methods tested:

- **Pivot**, Pivot with LocalSearch (**LS**), Pivot with InnerLocalSearch (**ILS**), and **Vote**
- Different attribute levels are tested for each algorithm
- **Bound** shows the theoretical cost increase limit
- Edge weights are computed on-the-fly
- LS and ILS restricted to one iteration only

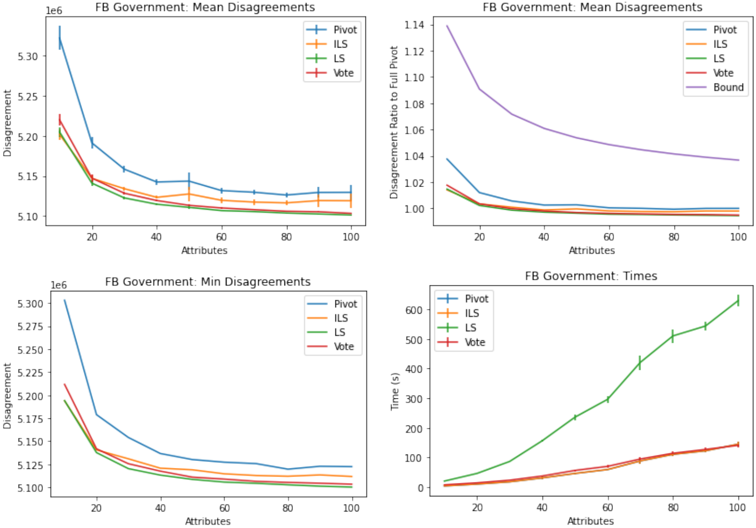
Experiments

Mushrooms: $k = 22$, $|V| = 8,214$



Experiments

Facebook Government: $k = 100$, $|V| = 7,057$



Conclusion

Accomplishments

- We demonstrated that Pivot, LS, and Vote were competitive with current state-of-the-art algorithms in general and constrained CC settings
- We developed and demonstrated the usefulness of ILS for Pivot on larger graphs
- We developed two practical improvements for applying CC algorithms to consensus clustering

Conclusion

Future Work

- Can we demonstrate improved approximation bounds for Pivot and related methods for constrained CC settings?
- How well do these methods perform in other CC settings?

References

AEKM20 Ahmadian, Epasto, Kumar, and Mahdian. *Fair correlation clustering*. 2020

ACGM15 Ahn, Cormode, Guha, McGregor, and Wirth. *Correlation clustering in data streams*. 2015

ACN08 Ailon, Charikar, and Newman. *Aggregating inconsistent information: ranking and clustering*. 2008

AL09 Ailon and Liberty. *Correlation clustering revisited: the "true" cost of error minimization problems*. 2009

BBC04 Bansal, Blum, and Chawla. *Correlation clustering*. 2004

BEK21 Bun, Elias, and Kulkarni. *Differentially private correlation clustering*. 2021

CMSY15 Chawla, Makarychev, Schramm, and Yaroslavtsev. *Near optimal lp rounding algorithm for correlation clustering on complete and complete k-partite graphs*. 2015

C17 Chehreghani. *Clustering by shift*. 2017

CDK14 Chierichetti, Dalvi, and Kumar. *Correlation clustering in mapreduce*. 2014

CMB17 Christiansen, Mobasher, and Burke. *Using uncertain graphs to automatically generate event flows from news stories*. 2017

CLN22 Cohen-Addad, Lee, and Newman. *Correlation clustering with sherali-adams*. 2022

References

CSW08 Coleman, Saunderson, and Wirth. *A local-search 2-approximation for 2-correlation-clustering*. 2008

ES09 Elsner and Schudy. *Bounding and comparing methods for correlation clustering beyond ILP*. 2009

GKBT20 García-Soriano, Kutzkov, Bonchi, and Tsourakakis. *Query-efficient correlation clustering*. 2020

GG06 Giotis and Guruswami. *Correlation clustering with a fixed number of clusters*. 2006

GF08 Goder and Filkov. *Consensus clustering algorithms: comparison and refinement*. 2008

HWH15 Halim, Waqas, and Hussain. *Clustering large probabilistic graphs using multi-population evolutionary algorithm*. 2015

IN16 Il'ev and Navrotskaya. *A local search for a graph correlation clustering*. 2016

JCTZ21 Ji, Cheng, Tan, and Zhao. *An improved approximation algorithm for capacitated correlation clustering problem*. 2021

JXLW20 Ji, Xu, Li, and Wang. *Approximation algorithms for two variants of correlation clustering problem*. 2020

KS09 Karpinski and Schudy. *Linear time approximation schemes for the gale-berlekamp game and related minimization problems*. 2009

References

KSZC21 Klodt, Seifert, Zahn, Casel, Issac, and Friedrich. *A color-blind 3-approximation for chromatic correlation clustering and improved heuristics*. 2021

KPT11 Kollios, Potamias, and Terzi. *Clustering large probabilistic graphs*. 2011

LMVW21 Lattanzi, Moseley, Vassilvitskii, Wang, and Zhou. *Robust online correlation clustering*. 2021

MTG20 Mandaglio, Tagarelli, and Gullo. *In and out: optimizing overall interaction in probabilistic graphs under clustering constraints*. 2020

PORJ15 Pan, Papailiopoulos, Oymak, Recht, Ramchandran, and Jordan. *Parallel correlation clustering on big graphs*. 2015

PM15 Puleo and Milenkovic. *Correlation clustering with constrained cluster sizes and extended weights bounds*. 2015

TCD19 Thiel, Chehreghani, and Dubhashi. *A non–convex optimization approach to correlation clustering*. 2019

ZW09 Zuylen and Williamson. *Deterministic pivoting algorithms for constrained ranking and clustering problems*. 2009