Hierarchical Clustering of the OEIS Graph

Hadley Black
Mentor: James Abello

2015 DIMACS REU

July 17, 2015

Overview

Graph Partitioning and Clustering

- *Partitioning:* Given an undirected weighted graph $G = (V, E, c, w)$ and $k > 1$, find a partition $\{V_1, ..., V_k\}$ of the vertex set V such that the total edge-cut

$$\sum_{i<j} w\Big((u, v) \in E \Big| u \in V_i, v \in V_j\Big)$$

is minimal.

Graph Partitioning and Clustering

- *Partitioning:* Given an undirected weighted graph $G = (V, E, c, w)$ and $k > 1$, find a partition $\{V_1, ..., V_k\}$ of the vertex set V such that the total edge-cut

$$\sum_{i<j} w\Big((u, v) \in E \Big| u \in V_i, v \in V_j\Big)$$

is minimal.

- *Clustering:* Partitioning, but k is not given in advance.

Overview

Outline

Bottom-up hierarchical clustering method.

Process:

1. *Pre-processing:*
    1. Vertex splitting.
    2. Graph peeling.
2. *Main steps:*
    1. Initial coarsening phase.
    2. Projection and refinement phase.
    3. Second coarsening phase with a size constraint.

Label Propagation

Label Propagation

Initialize each node as its own cluster. For each $u \in V$ (by increasing ID) move u to cluster V_i where

$$\sum_{v \in N_u} w\Big((u,v) \in E \Big| v \in V_i\Big) \text{ is maximized.}$$

Quotient Graph

Let $\{V_1, \ldots V_k\}$ be a k-clustering of G. The corresponding *quotient graph* is defined $Q = (V_Q, E_Q)$ where $V_Q = \{V_1, \ldots V_k\}$ and $E_Q = \Big\{(V_i, V_j) \Big| \exists (u,v) \in E_G, u \in V_i, v \in V_j\Big\}$. Edge and vertex weights: $w(V_i, V_j) = \sum_{u \in V_i, v \in V_j} w(u,v)$ and $w(V_i) = |V_i|$.

Projection and Refinement

For each G_i in the hierarchy (in descending order starting with G_k):

1. *Refine* the cluster at the current level using one iteration of *size-constrained* label propagation to obtain the new quotient graph G'_{i+1}.
2. *Project* the clustering to a *finer* graph in the hierarchy. For $v \in V_{G_{i-1}}$: $cluster_{G_i}[v] = cluster_{G'_{i+1}}[cluster_{G_i}[v]]$

Note: A mean cluster size constraint is applied: When applying label propagation to some vertex u, if a move to cluster V_i would result in $w(V_i) > \frac{|V|}{k}$, then that move is not considered.

Second Coarsening Phase

Build a hierarchy tree by iteratively running label propagation and computing the corresponding quotient graphs as before.

Differences:

- Initial cluster IDs for vertices in G_1 are given from refinement phase.
- As in during *refinement*, a mean cluster size constraint is applied.

Hierarchy Trees and Antichains

Hierarchy tree:

- A directed rooted tree.
- Leaves correspond to vertices in the original graph.
- Internal nodes correspond to clusters.

Hierarchy Trees and Antichains

Hierarchy tree:

- A directed rooted tree.
- Leaves correspond to vertices in the original graph.
- Internal nodes correspond to clusters.

Antichain:

- A set of vertices from the hierarchy tree such that there is no directed path between any pair.
- An antichain A is *maximal* if every *leaf* is reachable from some vertex in A.

Overview

Label Propagation Complexity

- One iteration of LP: For each vertex we examine each of its incident edges. Cost: $\mathcal{O}(n+m)$.

Label Propagation Complexity

- One iteration of LP: For each vertex we examine each of its incident edges. Cost: $\mathcal{O}(n+m)$.
- Building the quotient graph: This requires a single scan of the edge set. Cost: $\mathcal{O}(m)$.

Geometric Size Reduction

- The size of the vertex sets of graphs in the hierarchy decrease *geometrically*.
- Exponential decrease in a discrete domain.

Geometric Size Reduction

- The size of the vertex sets of graphs in the hierarchy decrease *geometrically*.
- Exponential decrease in a discrete domain.
- Thus we perform $\mathcal{O}(\log n)$ iterations of LP to obtain the full hierarchy tree.

Geometric Size Reduction

- The size of the vertex sets of graphs in the hierarchy decrease *geometrically*.
- Exponential decrease in a discrete domain.
- Thus we perform $\mathcal{O}(\log n)$ iterations of LP to obtain the full hierarchy tree.
- Complexity of the algorithm : $\mathcal{O}\big((n+m)\log n\big)$.

Overview

Added feature: Vertex Splitting

- If $deg(v) > U = \lfloor \log_2 |V| \rfloor$: v becomes $\{v_1, v_2 ..., v_k\}$ where $k = \lceil \frac{deg(v)}{U} \rceil$.
- For our data set $U = 18$.
- Helps avoid massive clusters centered around high degree vertices (sequences) such as the *primes* and *fibonacci numbers*.

Added feature: Peeling

- Iteratively remove vertices v where $deg(v) = 1$.
- This removes the peripheral forest of G.
- Treat each anchor point as a cluster containing the corresponding peripheral tree at the lowest level in the hierarchy.

Overview

Outline

Bottom-up hierarchical clustering method.

Process:

1. *Pre-processing:*
    1. Vertex splitting.
    2. Graph peeling.
2. *Main steps:*
    1. Initial coarsening phase.
    2. Projection and refinement phase.
    3. Second coarsening phase with a size constraint.

References

- (Semi-)External Algorithms for Graph Partitioning and Clustering by Akhremtsev, Sanders, and Schulz. 2014.
- Ask Graph View: The Design of a Large Scale Graph Visualization System by J. Abello, F. Van Ham, N. Krishnan. 2006.
- Simple Linear Algorithms for Mining Graph Cores by Yang Xiang. 2014.