COMS E6998: Advanced Topics in Computational Learning Theory.
Author: Hila Becker
Date: May 5, 2005

A Survey of Correlation Clustering

Abstract

The problem of partitioning a set of data points into clusters is found in many applications. Correlation clustering is a clustering technique motivated by the the problem of document clustering, in which given a large corpus of documents such as web pages, we wish to find their optimal partition into clusters. While most commonly used clustering algorithms such as k-means, k-clustering sum and k-center require prior knowledge of the number of clusters that we wish to divide the data into, for the case of classifying web documents, finding the number of clusters is not a trivial task. Correlation Clustering, introduced by Bansal, Blum and Chawla [1], provides a method for clustering a set of objects into the optimal number of clusters, without specifying that number in advance. In this paper we present two different approximation algorithms for the Correlation Clustering problem. We then discuss some open problems and give our intuition as to how to approach them.

1 Introduction

Clustering is the task of partitioning data points into groups based on their similarity. Clustering techniques are commonly used in fields such as machine learning and data mining. This project is motivated by the problem of clustering a large corpus of documents, such as web pages, when we do not want to establish a set number of clusters k to partition the data into. Most known clustering algorithms such as k-means, k-sum and k-center, require the user to specify the number of clusters they wish to obtain prior to the execution of the algorithm. However, in the case that one does not want to place such a constraint on the task, as in the case of clustering web pages where the number of clusters cannot be easily determined, the mentioned algorithms do not apply.

Recently, [1] came up with a new clustering technique named Correlation Clustering that does not require a bound on the number of clusters that the data is partitioned into. Rather, Correlation Clustering divides the data into the optimal number of clusters based on the similarity between the data points. In their paper, [1] Bansal et al. discuss two objectives of correlation clustering: minimizing disagreements and maximizing agreements between clusters. The decision version of these optimization problems was shown in [1] to be NP-Complete using a reduction from X3C.

We assume that there exists a classifier function f, such that given two data points x and y, outputs '+' if x, y are similar and '-' if they are dissimilar. We can view the data points as nodes in a connected graph G whose edges are labeled according to the classifier. The task of maximizing agreements is therefore trying to maximize the number of '+' edges connecting nodes in the same cluster plus '-' edges connecting nodes across clusters, while minimizing disagreements is trying to minimize the '-' labeled edges inside cluster plus the number of '+' edges across clusters.

Correlation Clustering can be viewed as an agnostic learning problem, where we try to learn the classifier function f from past data, and our goal is to partition the current data in order to optimize the correlation with f. The problem of agnostic learning is trying to find the best representation of the target function f using a hypothesis class with limited representational power. More specifically, the edges of the graph can be viewed as labeled examples and we try to represent the classifier

f using a hypothesis class of vertex clusters. So whenever we have (u, v) and (v, w) as positive examples, (u, w) must also be a positive example, therefore we might not be able to represent f in the best possible way.

Most of the work on this clustering method is fairly new, as it was only introduced in 2002 by [1]. For the rest of this paper, we will discuss some of the major results as well as a couple of open questions and an intuition for their solution.

2 Definitions

Let $G = (V, E)$ be a graph on an vertices with edge weights $c_e \geq 0$. Let $e(u, v) \in \{+, -\}$ be the label of the edge (u, v). The positive neighborhood of u is $N^+(u) = \{u\} \cup \{v : e(u, v) = +\}$ and the negative neighborhood of u is $N^-(u) = \{u\} \cup \{v : e(u, v) = -\}$. Let OPT represent the optimal clustering, and for a clustering C, we let $C(v)$ be the set of vertices that are in the same cluster as v. Consider a clustering $C = \{C_1, C_2, \ldots, C_n\}$. A negative labeled edge inside a cluster is considered a *negative mistake* and a positive labeled edge between clusters is considered a *positive mistake.* If our goal is to minimize disagreements, we minimize the weight of positive edges between clusters and absolute valued weight of negative edges inside clusters. When maximizing agreements we wish to maximize the weight of positive edges inside clusters plus the absolute valued weights of negative edges between clusters.

3 Previous Work

3.1 Overview

In the original paper that introduced the problem, Bansal et al. [1] showed a constant factor approximation algorithm for minimizing disagreements, based on the principle of counting *erroneous triangles* which are triangles with two positive labeled edges and one negative labeled edge. In addition, they also showed a PTAS for maximizing agreements similar to the PTAS for MAXCUT on dense graphs, focusing on complete graphs, as well as graphs with edge labels -1 and 1.

The work on minimizing disagreement was extended in [3] where an $O(logn)$ approximation algorithm is given for general graphs is presented using linear programming and region-growing techniques. Demaine and Immmorlica [3] also proves that the problem of minimizing disagreements is as hard as the APX-hard problem *minimum multicut.*

Charikar et al. [2] gave a factor 4 approximation algorithm for minimizing disagreements on complete graphs, and also a factor $O(logn)$ approximation for general graphs. In addition they showed a factor 0.7664 approximation for general graphs and proved that finding a PTAS is APX-hard. Similar results on the hardness of minimizing disagreements were obtained independently by Emanuel et al.[4]. Lastly, Swamy [6] gave a 0.7666-approximation algorithm for maximizing agreements in general graphs with non-negative edge weights using semidefinite programming.

We will show the constant factor approximation algorithm for minimizing disagreements in complete graphs presented in [1] as well as the $O(logn)$ approximation algorithm of [3] for minimizing disagreements in general graphs. Each of these algorithms uses a different method for approximating and optimizing the clustering. Looking at these different methods is useful when considering the open questions related to this topic.

3.2 Minimizing Disagreements in Complete Graphs

Bansal et al. [1] show an approximation algorithm for clustering by minimizing disagreements in complete graphs. Their approach is to show that the number of errors (disagreements) made by the algorithm is bounded by a constant factor of OPT. They argue that the number of mistake that the algorithm makes can be attributed to the number of *erroneous triangles*, where two edges are labeled postive and one edge is labeled negative. We will first present the argument that if we get a partial clustering that is sufficiently "clean" then we can attribute the errors to erroneous triangle such that the triangle we choose are almost completely edge disjoint (Lemma 1). We will then argue that must exist a clustering OPT' which is close to optimal, where every non-singleton cluster is sufficiently "clean". At the end we can present an algorithm that will guarantee such a clustering with the number of mistakes in singleton clusters bouned by OPT' and the number of mistakes in "clean" clusters bounded using Lemma 1.

Definition 1 *A vertex v is δ-good with respect to cluster $C \subseteq V$ if*

- $|N^+(v) \cap C| \geq (1-\delta)|C|$
- $|N^+(v) \cap (V \setminus C)| \leq \delta|C|$

A vertex v that is not δ-good with respect to C is said to be δ-bad with respect to C.
C is δ-clean if all $v \in C$ are δ-good with respect to C.

Lemma 2 *The number of mistakes made by a clustering of V where all clusters are δ-clean for delta $\leq 1/4$ is at most 8 times the number of mistakes made by OPT.*

Proof We bound the number of mistakes made by OPT by the number of edge-disjoint erroneous triangles, which are triangles with two '+' edges and one '-' edge. We can do so since OPT must make at least one mistake for each such triangle. We therefore present a bound on the clustering of V, which is at most 8 times the number of edge-disjoint erroneous triangles. Since the total number of mistakes made by the algorithm is the number of negative mistakes plus the number of positive mistakes, we need to consider both possibilities.

For negative mistakes, pick a negative labeled edge (u, v) such that u and v belong to the same cluster C_i. We then choose a node $w \in C_i$ such that (u, w) and (v, w) are both positive labeled edges, and therefore (u, v) will be a negative edge in the erroneous triangle (u, v, w). We need to argue that for all (u, v) we can choose a w such that no other negative edges having either u or v as one of their endpoints, also choose w. We know that C_i is δ-clean, so by definition u and v have at most $\delta|C_i|$ neighbors in C_i. Therefore for (u, v) we can choose at least $(1-2\delta)|C_i|$ vertices w such that (u, w) and (v, w) are positive. We can also infer that at most $2\delta|C_i| - 2$ could have been chosen by other negative edges containing either u or v. We thus have at least $(1-2\delta)|C_i| - (2\delta|C_i| - 2) = (1-4\delta)|C_i| + 2$. choices of w. Since we have $\delta \leq 1/4$, there will always be a w for (u, v) to choose. Since each positive edge (v, w) can be chosen for negative mistakes on v or negative mistakes on w, (v, w) may be chosen at most twice. Therefore, if we use edge disjoint erroneous triangles, we can account for at least 1/4 of the negative mistakes.

For positive mistakes, consider the edge (u, v) where $u \in C_i$ and $v \in C_j$, since a positive mistake refers to an positive edge between clusters. We choose a node $w \in C_i$ such that (u, w) is positive and (v, w) is negative. Similarly, there will be at least $|C_i| + \delta(|C_i| + |C_j|)$ vertices that obey this condition, of which at most $\delta(|C_i| + |C_j|)$ will be taken. Just as in the case of negative mistakes, each positive edge can be chosen at most twice so we can account for at least 1/4 of the positive mistakes by using edge-disjoint erroneous triangles.

Therefore, having both positive and negative mistakes we can choose edge disjoint erroneous triangles to account for at least 1/8 of the mistakes made by the clustering. ■

Lemma 3 *There exists a clustering OPT' such that all non-singleton clusters are δ-clean and the number of mistakes made by OPT' is at most $(\frac{9}{\delta^2}+1)$ times the number of mistakes made by OPT.*

Proof We apply a "clean-up" procedure to the optimal clustering that results in the clustering OPT'. Let $C_1, C_2, ..., C_n$ be the clusters of OPT. Let $S = \emptyset$.

- For $i = 1$ to k do:
  - If there are more than $\frac{\delta}{3}|C_i|$ $\frac{\delta}{3}$-bad vertices in C_i, "dissolve" the cluster C_i by letting $C'_i = \emptyset$ and $S = S \cup C_i$.
  - Else let B_i be the set of $\frac{\delta}{3}$-bad vertices in C_i. Then $S = S \cup B_i$ and $C'_i = C_i$ B_i
- Output the clustering of OPT' = $C'_1, C'_2, ..., C'_n$, $\{x\}_{x \in S}$.

We will show that the mistakes made by OPT and OPT' are related, starting by showing that each C'_i is δ-clean. This claim is trivial when C'_i is empty. Otherwise, we know that $|C_i| \geq |C'_i| \geq (1-\frac{\delta}{3})|C_i|$. For every vertex v in cluster C'_i the following holds:

$$|N^+(v) \cap C'_i| \geq (1-\frac{\delta}{3})|C_i| - \frac{\delta}{3}|C_i| > (1-\delta)|C'_i|$$

Also, for $\bar{C}'_i = V \setminus C'_i$

$$|N^+(v) \cap \bar{C}'_i| \leq \frac{\delta}{3}|C_i| + \frac{\delta}{3}|C_i| \leq \frac{2\delta}{3}\frac{|C'_i|}{1-\delta/3} < \delta|C'_i|(as\delta < 1)$$

This shows that every C'_i is δ-clean. Next, we determine the number of mistakes. In the case where we dissolve a cluster C_i then the number of mistakes associated with it is at least $\frac{\delta}{3}^2|C_i|^2/2$. Thus, the number of mistakes we get by dissolving this cluster is at most $|C_i|^2/2$. If the cluster C_i was not dissolved then the number of mistakes associated with it is at least $\frac{\delta}{3}|C_i||B_i|/2$. So using the above procedure we only add at most $|C_i||B_i|$ mistakes. Note that dissolving a cluster adds at most $\frac{\delta^2}{9}$ fraction of the mistakes made by the optimal cluster, and not dissolving a cluster adds at most $\frac{\delta}{6} < \frac{\delta^2}{9}$ fraction of the mistakes made by the optimal cluster. Therefore, we have shown the existence of a clustering OPT' in which all non-singleton clusters are δ-clean and the number of mistakes made by OPT' is at most $(\frac{9}{\delta^2}+1)$ times the number of mistakes made by OPT.

■

We now present an algorithm that tries to find clusters similar to OPT'. Note that the clusters C'_i in OPT are non-singleton clusters, while the clusters added to S are singleton clusters.

Algorithm I

- Choose a vertex v arbitrarily from the set of vertices V
  1 Let $A(v) = N^+(v)$
  2 While there exists a vertex x in $A(v)$ such that x is 3δ-bad with respect to $A(v)$, remove x from $A(v)$.
  3 Let $Y = \{y | y \in V$, y is 7δ-good with respect to $A(v)\}$. Add all vertices in Y to $A(v)$.
- Remove all vertices of $A(v)$ from the set of vertices V
- Repeat until
  - No vertices are left in V
  - All sets $A(v)$ are empty, in which case output all remaining vertices as singleton clusters.

Let $A_1, A_2, A_3, \dots$ be the clusters that the algorithm outputs, and let Z be the set of singleton clusters. We would like to show that the number of mistakes made by *Algoritm I* is within a constant factor of the number of mistakes made by OPT. To do so, we want to use the following theorem.

Theorem 4 *$\forall j \exists i$ such that $C'_j \subseteq A_i$, and every A_i is 11δ-clean*

Since the proof of this theorem is lengthy and based on several lemmas, we will not detail it here but rather give a brief sketch. For the complete proof please refer to [1].

Proof sketch: First, we must show that if $v \in C'_i$ where C'_i is a δ-clean cluster in OPT', then any vertex $w \in C'_i$ is 3δ-clean with respect to $N^+(v)$. We then show that given an arbitrary set X, if $v_1 \in C'_i$ and $v_2 \in C'_j$ then v_1 and v_2 cannot be both 3δ-good with respect to S. From these two facts, we get that after step 2 of the algorithm, no two vertices from distinct C'_i and C'_j can be both in $A(v)$. Then we claim that each A_i is either a subset of S or contains exactly one of the clusters C'_j.

Then we go on to show that for every $j, \exists i$ such that $C'_j \subseteq A_i$. We do so by proving that for any vertex v chosen by the algorithm such that $v \in C'_j$, the algorithm does not remove any vertex from $N^+(v) \cap C'_j$ during step 2. This can be easily shown by induction. We get that at the end of step 2 of the algorithm, A'_i contains at least $(1-\delta)|C'_j|$ vertices of C'_j. The first part of the theorem follows. Then we finally show that every A_i is 11δ-clean, using the fact that in step 3 we add vertices that are 7δ-good with respect to A'_i. We use this to show that $N^+(v) \cap \bar{A}_i \leq 7\delta|A'_i| \leq 11\delta|A_i|$.

■

We bound the number of mistakes of our algorithm by the number of mistakes made by OPT' and OPT. We refer to mistakes for which both vertices are associated with some clusters A_i and A_j as *internal mistakes*, and mistakes that have one node in the set Z as *external mistakes.*

Lemma 5 *The total number of external mistakes made by Algorithm I is less than the number of external mistakes made by OPT'.*

Proof We know that no vertex $v \in C'_i$ can be contained in the set of singleton clusters Z. Therefore, $Z \subseteq S$. Any external mistakes made by the algorithm correspond to positive labeled edges with one endpoint in Z. These are also mistakes in OPT' since those edges are incident on vertices in S. The lemma follows. ■

We now need to relate the number of internal mistakes made by *Algorithm I* to the number of internal mistakes made by OPT' and OPT. Using Lemma 2, we can see that the clustering of the vertices obtained using *Algorithm I*

Lemma 6 *The total number of internal mistakes made by Algorithm I is at most 8 times the number of mistakes made by OPT.*

Using lemmas 5, 6 and 3, we proved that our algorithm gives a constant factor approximation to OPT.

Theorem 7 *The number of mistakes made by algorithm I is at most* $9(\frac{1}{\delta^2}+1)$ *times the mistakes made by OPT.*

3.3 Minimizing Disagreements in General Weighted Graphs

In this section, we present an $O(logn)$ approximation algorithm for minimizing disagreements in general weighted graphs which was given by Demaine et al. [5]. This algorithm uses a combination of Linear Programming, Rounding and Region-Growing techniques. This algorithm first solves a linear program and then uses the resulting fractional values to determine the distance between two vertices, where larger distance corresponds to weaker similarity. In the last step of the algorithm, we use the region-growing technique to group close vertices together and round the fractional values.

Consider the graph $G = (V, E)$. Let x_{uv} be a boolean variable representing the edge label of $e(u, v) \in E$, $u, v \in V$. Given a clustering C, let $x_{uv} = 0$ if u and v are in the same cluster, and $x_{uv} = 1$ if they are in different clusters. Recall that the number of mistakes in clustering C is the sum of the positive and negative mistakes. We define the weight of the clustering as the sum of the sum of the weight of erroneous edges in C. For a clustering C, the weight of the clustering is $w(C) = w_p(C) + w_n(C)$ where $w_p(C) = \sum\{e = (u, v) \in E^+, u \notin C(v)\}$ and $w_n(C) = \sum\{e = (u, v) \in E^-, u \in C(v)\}$. Note that $(1-x_{uv}) = 1$ if the edge (u,v) is inside a cluster and $(1 - x_{uv}) = 0$ otherwise. Let c_e denote an edge $e(u, v) \in E$, we can rewrite the weight of C as

$$w(C) = \sum_{e\in E^-} c_e(1 - x_e) + \sum_{e\in E^+} c_e x_e$$

In order to minimize disagreements, we need to find an assignment to x_{uv} that minimizes the weight such that $x_{uv} \in \{0, 1\}$ and x_{uv} satisfies the triangle inequality. We formulate the problem as a linear program

$$Minimize \sum_{e\in E^-} c_e(1 - x_e) + \sum_{e\in E^+} c_e x_e$$

Such that $x_{uv} \in [0, 1]$, $x_{uv} = x_{vu}$ and $x_{uv} + x_{vw} \geq x_{uw}$.

We show a procedure to round this LP in order to get an $O(logn)$ approximation using the region-growing technique. Region-growing refers to the procedure of growing a "ball" around nodes in a graph by iteratively adding nodes of fixed distance r to the ball, until all nodes belong to some "ball". In our algorithm, the set of nodes contained in each ball corresponds to the set of nodes that make up each cluster.

Definition 8 *A "ball" $B(u,r)$ of radius r around a node u is the set of nodes v such that $x_{uv} \leq r$, as well as the subgraph induced by these nodes and the $(\frac{r-x_{ux}}{x_{vw}})$ edges (v,w) with only one endpoint $v \in B(u,r)$.*

Definition 9 *The cut of a set S is the weight of the positive edges with exactly one endpoint in S.*

$$Cut(S) = \sum_{|(v,w) \cap S|=1, (v,w) \in E^+} c_{vw}$$

The cut of a "ball" is $Cut(S)$ where $S = \{v | v \in B(u,r)\}$.

Definition 10 *The volume of a set S is the weighted distance of edges (u,v) such that $u, v \in S$.*

$$Vol(S) = \sum_{(v,w) \subset S, (v,w) \in E^+} c_{vw} x_{vw}$$

The volume of $B(u,r)$ includes the fractional weighted distance of the positive edges leaving $B(u,r)$. If $(v,w) \in E^+$ is a cut positive edge of $B(u,r)$ with $v \in B(u,r)$ and $w \notin B(u,r)$ then (v,w) contributes weight of $c_{vw}(r - x_{uv})$ to the volume of $B(u,r)$. Let I be the initial volume of the "ball", so the volume of $B(u,0)$ is I.

We would show an algorithm that rounds a fractional solution to an integral solution. Let F be the volume of the graph G. Assume that the weight of the positive mistakes made by the fractional solution is F. Let the initial volume $I = F/n$ and let c be a constant.

Algorithm II

- Arbitrarily choose a node $u \in G$.
- Set radius $r = 0$.
- Increase r by $min\{(d_{uv} - r) > 0 : v \notin B(u,r)\}$ so that $B(u,r)$ includes another edge. Repeat until $Cut(B(u,r)) \leq c \ln(n+1) \times Vol(B(u,r))$.
- Output the set of nodes in $B(u,r)$ as one cluster of C.
- Remove the vertices and incident edges of $B(u,r)$ from G.
- Repeat the above steps until G is empty.

We now analyze this algorithm to show that the cost of the resulting solution is not significantly larger than the cost of the fractional solution. We use OPT to refer to the optimal solution and $FRAC(x_{uv})$ and $Round(x_{uv})$ to refer to the fractional and rounded solutions of x_{uv} in the LP respectively. We show that *Algorithm II* gives an $O(logn)$ approximation to the cost of positive between clusters edges and the cost of negative edges inside clusters.

Let $\mathcal{B}$ be the set of balls found by *Algorithm II.*

$$w_p(ROUND) = \sum_{(u,v) \in E^+} c_{uv} ROUND(x_{uv}) = \frac{1}{2} \sum_{B \in \mathcal{B}} Cut(B) \tag{1}$$

Since our algorithm grows each B until $Cut(B) \leq c\ln(n+1) \times Vol(B)$ we get from (1)

$$w_p(ROUND) \leq \frac{c}{2}\ln(n+1) \times \sum_{B\in\mathcal{B}} Vol(B) \tag{2}$$

By the design of the algorithm, all generated balls are disjoint so using (2) and our prior assumption $F = w_p(FRAC)$ we get

$$\begin{aligned} w_p(ROUND) &\leq \frac{c}{2}\ln(n+1) \times \left(\sum_{(u,v)\in E^+} c_{uv} FRAC(x_{uv}) + \sum_{B\in\mathcal{B}} \frac{F}{n} \right) \\ &\leq \frac{c}{2}\ln(n+1) \times (w_p(FRAC) + F) \\ &\leq c\ln(n+1) \times w_p(FRAC) \end{aligned}$$

To observe the $O(logn)$ approximation we claim that the balls returned by the algorithm have radius $r \leq 1/c$, which follows from the lemma [5], [10]

Lemma 11 *For any vertex u and a family of balls $B(u,r)$, the condition $Cut(B(u,r)) \leq c\ln(n+1) \times Vol(B(u,r))$ is achieved by some $r \leq 1/c$.*

We now show that the algorithm guarantees an $O(1)$ approximation to the cost of negative edges. We do so by using the above lemma to guarantee the bound on the radius and proving that the solution is a $\frac{c}{c-2}$-approximation of the cost of negative edges inside clusters. Let $\mathcal{B}$ be the set of balls found by *Algorithm II.* We get,

$$\begin{aligned} w_n(FRAC) &= \sum_{(u,v)\in E^-} c_{uv}(1 - FRAC(x_{uv})) && (3) \\ &\geq \sum_{B\in\mathcal{B}} \sum_{(u,v)\in B\cap E^-} c_{uv}(1 - FRAC(x_{uv})) && (4) \\ &\geq \sum_{B\in\mathcal{B}} \sum_{(u,v)\in B\cap E^-} c_{uv}(1 - 2/c) && (5) \\ &\geq (1 - 2/c) \sum_{B\in\mathcal{B}} \sum_{(u,v)\in B\cap E^-} c_{uv}(1 - 2/c) && (6) \\ &= \frac{c-2}{c} w_m(ROUND) && (7) \end{aligned}$$

Equation (5) follows from (4), the triangle inequality and the fact that $r \leq 1/c$. The algorithm guarantees an $O(1)$ approximation given that $c > 2$ in the approximation-ratio $\frac{c}{c-2}$.

We therefore get the total number of mistakes made by the algorithm to be

$$\begin{aligned} w(ROUND) &= w_p(ROUND) + w_n(ROUND) \\ &\leq c\ln(n+1) \times w_p(OPT) + \frac{c}{c-2} \times w_n(OPT) \\ & max\left\{ c\ln(n+1), \frac{c}{c-2} \right\} w(OPT) \end{aligned}$$

So in total the number of mistakes made by *Algorithm II* is $O(logn)$ of OPT, where $c > 2$.

4 Open Problems

Although the majority of the open problems posed by the authors of the original paper [1] were solved in subsequent papers, some of them as recently as February of 2005, there are several questions that remain open and improvements that can still be made to some of the approximation factors. The first problem proposed in [3], [5], is to find an $O(logn)$ approximation algorithm for minimizing disagreements in general weighted graphs, that is combinatorial rather using LP. However, given the hardness results shown in [5] it is unlikely to find an approximation to this problem better than $O(\log n)$.

Another problem mentioned in [1], [3], [5] is trying to use Correlation Clustering with different objective functions. In particular, an interesting objective would be maximizing agreements minus disagreements, which corresponds to maximizing the correlation. The only known result for this objective was given by [11] using an $(\Omega(1/\log n))$-approximation algorithms. There is still room for improvements on algorithms for maximizing the correlation, as it was even described by the authors of [11] as being "a long way from the best-known hardness of approximation result".

For maximizing agreements on weighted general graphs, there is a trivial 1/2-approximation by either putting all nodes in one cluster or every node in a separate cluster depending on the number of positive edges. The .7776-approximation algorithm is given in [6] using semidefinite programming for general *unweighted* graphs. It is still open whether we can get a better approximation for general weighted graphs.

4.1 Approach and Intuition

We decided to look at several approaches to tackle these open problems. For the problem of giving a combinatorial $O(\log n)$ approximation for minimizing disagreements in general weighted graphs, we looked at the method presented in this paper, which gives a constant-factor approximation for minimizing disagreements on complete graphs by counting the number of erroneous triangles. Since the bound given by counting triangles will not be a sufficient approximation, we attempted to find an algorithm that bounds the number of error by the number of erroneous cycles in a cycle cover of the graph, although no results were found. Due to our limited background on approximation algorithms, it was necessary to gather information from several sources such as [8] [9] and [12] on construction methods for approximation algorithms. In particular, we were interested in linear programming and semidefinite programming formulations since these techniques proved useful for approximating some of the related problems. We also decided to look into the obvious relation of the correlation clustering problem to the MIN/MAX CUT. This seemed to be a likely direction to turn to since the PTAS given for maximizing agreements in [1] is modeled using the PTAS given for MAX CUT on dense graphs. Also, [5] and [2] gave hardness of approximation results for minimizing disagreements in general weighted graphs based on the minimum Multicut problem. Our intuition was that this would be of particular interest when trying to give a better approximation for general unweighted graphs, but it was difficult to find a comparable approximation algorithm that could be adapted to solve our problem as well. Lastly, the region-growing technique for solving the problem of minimizing disagreements that was presented in this paper seemed like an approach that would be useful in practice. We chose to implement the algorithm using this approach for clustering a large set of Web pages. We selected different samples of 200 pages each from different genres to test the clustering algorithm on. The accuracy of the algorithm compared to clustering by manual inspection was about than 95%. Even though the results have a dependency on the classifier function used to decide whether two documents are similar, we believe that the accuracy of the results points to the fact that using the region-growing technique can help get better approximation results for the open correlation clustering problems that were not shown to have a tight approximation factor.

References

[1] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. In Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, pages 238250, Vancouver, Canada, November 2002.

[2] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. In Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, pages 524533, 2003.

[3] Erik D. Demaine and Nicole Immorlica. Correlation clustering with partial information. In Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pages 113, Princeton, NJ, August 2003.

[4] Dotan Emanuel and Amos Fiat. Correlation clustering minimizing disagreements on arbitrary weighted graphs. In Proceedings of the 11th Annual European Symposium on Algorithms, pages 208220, 2003.

[5] Erik D. Demaine, Nicole Immorlica, Dotan Emanuel and Amos Fiat. Correlation Clustering in General Weighted Graphs. Special issue on approximation and online algorithms 2005.

[6] Chaitanya Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, pages 526527, New Orleans, LA, 2004. Society for Industrial and Applied Mathematics.

[7] Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. Clustering for mining in large spatial databases. KI-Journal, 1, 1998. Special Issue on Data Mining. ScienTec Publishing.

[8] Ari Freund. The Local Ratio and Primal-Dual Methods in Approximation Algorithms, Seminar (203.3485). *http://cs.haifa.ac.il/courses/approx_algo/*

[9] Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. Journal of the ACM, 33:533550, 1986.

[10] Approximation algorithms. Vijay V. Vazirani. Berlin ; New York : Springer, c2001.

[11] Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendiecks inequality. In Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, pages 5460, 2004.

[12] Optimization Software and Theory. Henry Wolkowicz. *http://orion.math.uwaterloo.ca/ hwolkowi/henry/software/readme.html#combopt*