Clustering Social Networks

Nina Mishra[1,4], Robert Schreiber[2], Isabelle Stanton[1⋆], and Robert E. Tarjan[2,3]
{nmishra,istanton}@cs.virginia.edu,
{rob.schreiber,robert.tarjan}@hp.com

[1] Department of Computer Science, University of Virginia
[2] HP Labs
[3] Department of Computer Science, Princeton University
[4] Search Labs, Microsoft Research

Abstract. Social networks are ubiquitous. The discovery of close-knit clusters in these networks is of fundamental and practical interest. Existing clustering criteria are limited in that clusters typically do not overlap, all vertices are clustered and/or external sparsity is ignored. We introduce a new criterion that overcomes these limitations by combining internal density with external sparsity in a natural way. An algorithm is given for provably finding the clusters, provided there is a sufficiently large gap between internal density and external sparsity. Experiments on real social networks illustrate the effectiveness of the algorithm.

1 Introduction

Social networks have gained popularity recently with the advent of sites such as MySpace, Friendster, Facebook, etc. The number of users participating in these networks is large, e.g., a hundred million in MySpace, and growing. These networks are a rich source of data as users populate their sites with personal information. Of particular interest in this paper is the graph structure induced by the friendship links.

A fundamental problem related to these networks is the discovery of clusters or communities. Intuitively, a cluster is a collection of individuals with dense friendship patterns internally and sparse friendships externally. We give a precise definition of a cluster shortly. There are many reasons to seek tightly-knit communities in networks, for instance, target marketing schemes can be designed based on clusters, and it has been claimed that terrorist cells can be identified [12].

What is a good cluster in a social network? There are numerous existing criteria for defining good graph clusters, and accompanying each criterion is a multitude of algorithms. One popular criterion is based on finding clusters of high conductance. The conductance of a cut A, B is the ratio of the number of edges crossing the cut to the minimum of the volume of A and B, where the volume of A is the number of edges emanating from the vertices in A. The conductance

⋆ Supported by an NPSC Graduate Fellowship and Google Anita Borg Scholarship.

of a cluster is the minimum conductance of any cut in the cluster. A spectral algorithm is typically used to discover these clusters where the eigenvector of a matrix related to the adjacency matrix can be used to find a good cut of the graph into subgraphs A, B. The process is then recursively repeated (on A and B) until k clusters are found (where k is an input parameter) or until the conductance of the next best cut is larger than some threshold. Formal guarantees can be proved for some variants of this basic algorithm [9].

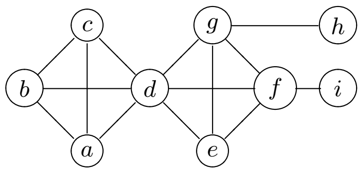

Fig. 1. Overlapping clusters.

Cut-based graph clustering algorithms produce a strict partition of the graph. This is particularly problematic for social networks as illustrated in Fig. 1. In this graph, d belongs to two clusters $\{a, b, c, d\}$ and $\{d, e, f, g\}$. Furthermore, h and i need not be clustered. A cut-based approach will either put $\{a, b, c, d, e, f, g\}$ into one cluster, which is not desirable since e, f, g have no edges to a, b, c, or cut at d, putting d into one of the clusters, say $\{a, b, c, d\}$, but leaving d out of $\{e, f, g\}$ which then leaves a highly connected vertex outside of the cluster.

This example motivates a new formulation of the graph clustering problem that does not stipulate that each vertex belong to exactly one cluster. Our objective is to identify clusters that are internally dense, i.e., each vertex in the cluster is adjacent to at least a β-fraction of the cluster, and externally sparse, i.e., any vertex outside of the cluster is adjacent to at most an α-fraction of the vertices in the cluster. For a vertex v and a subset of vertices C, the notation $E(v, C)$ below denotes the set of edges between v and C.

Definition 1. *Given a graph, $G = (V, E)$, where every vertex has a self-loop*[5] *$C \subset V$ is an* (α, β)-cluster *if*

1. ***Internally Dense:*** $\forall v \in C,\ |E(v, C)| \geq \beta |C|$
2. ***Externally Sparse:*** $\forall u \in V \setminus C,\ |E(u, C)| \leq \alpha |C|$

Given $0 \leq \alpha < \beta \leq 1$*, the* (α, β)-clustering problem *is to find all (α, β)-clusters.*

The new clustering criterion does not seek a strict partitioning of the data. To see why clusters can overlap, return to Fig. 1. Both $\{a, b, c, d\}$ and $\{d, e, f, g\}$ are $(\frac{1}{4}, 1)$-clusters. Furthermore, h and i do not fall into an (α, β)-cluster if $0 \leq \alpha < \frac{1}{2} < \beta \leq 1$, and consequently would not be clustered.

[5] This is a technical assumption needed to ensure that $\beta = 1$ clusters are possible.

Observe that as $\beta \to 1$, the cluster C approaches a clique and as $\alpha \to 0$, C tends to a disconnected component. We want $\alpha < \beta$, since we want vertices outside of a cluster to have fewer neighbors in the cluster than vertices that belong to the cluster.

While we use social networks as a motivating context, our problem statement and algorithms apply to the more general context of graph clustering.

Contributions We begin by investigating combinatorial properties of (α, β)-clusters. We bound the extent to which two clusters can overlap. For two clusters of equal size, we show that they overlap in at most $\min\{1-(\beta-\alpha), \alpha/(2\beta-1)\}|C|$ vertices. For certain values of α and β, it is possible for one cluster to be contained in another. However, we show that if the ratio of the size of the largest cluster to the smallest cluster is at most $\frac{1-\alpha}{1-\beta}$ then one cluster cannot be contained in another. Finally, we give a loose upper bound on the number of $(\alpha, 1)$-clusters of size s, $O((n/s)^{\alpha s+1})$, where n is the number of vertices.

Next, we introduce the notion of a ρ-champion of a cluster: a vertex with at most $\rho|C|$ neighbors outside of the cluster C. We prove that in the case that there is a large gap between $\alpha/2$ and β i.e., $\beta > \frac{1}{2} + \frac{\rho+\alpha}{2}$, there can be at most n (α, β)-clusters with ρ-champions of a given cluster size and there is a simple deterministic algorithm for finding all such clusters in time $O(m^{0.7}n^{1.2} + n^{2+o(1)})$, where m is the number of edges.

To determine whether the theoretical constructs we introduced actually exist in practice, we tested our algorithms on three real networks: High Energy Physics Co-authors, Theory Co-authors and Live Journal. Experiments show that our algorithm is able to find 90% of the ground-truth clusters of practical interest more quickly than previous algorithms.

2 Related Work

Our (α, β)-clustering formulation is new, but has been considered in restricted settings under different guises. The problem of finding the $(0, \beta)$-clusters in a graph can be reduced to first finding connected components and then outputting the components that are β-connected. This problem can be solved efficiently via depth first search in $O(|E| + |V|)$ time for a graph $G = (V, E)$. Also, the problem of finding $(1 - \frac{1}{n}, 1)$-clusters is equivalent to finding the maximal cliques in a graph. This problem has a rich history. Known algorithms find all maximal cliques in time that depends polynomially on the size of the graph and the number of maximal cliques [18, 8].

The problem of finding $((1-\epsilon)\beta, \beta)$-clusters, for small ϵ, has also been studied under the name of finding quasi-cliques. Abello et al. [1] present a method for finding subgraphs with average connectivity β. Hartuv and Shamir [6] find densely connected subgraphs where $\beta > 1/2$ via a min-cut algorithm. These algorithms do not consider an external sparsity (α) criterion. We will give an example (Fig. 2) where if these algorithms were used to find $(1/n, 1-1/2n)$-clusters (of which there is only 1), they return 2^n $(\frac{n-1}{n}, 1)$-clusters.

Spectral clustering is a very popular method that involves recursively splitting the graph using various criteria, e.g., the principal eigenvector of the adjacency matrix. Successful approaches have been employed by [9, 16, 10, 17, 15], among many others. All of these approaches do not allow overlapping clusters which is one of the main goals of our work.

Newman and others have advocated modularity as an optimization criterion for graph partitioning [15]. The modularity of a partition is the amount by which the number of edges between vertices in the same subset exceeds the number predicted by the degree-distribution preserving random graph model of Chung [2]. Newman proposed several methods for optimizing modularity, among them a spectral approach, and others have found competitive methods as well.

Flake et al. [4] use a recursive cut approach intended to optimize the expansion of the clustering but use Gomory-Hu trees [5] to find the cut instead of eigenvectors. The expansion of a cut is very similar to the conductance of a cut. The minimum quality of the clustering is guaranteed by adding a sink to the graph. Again, the goal of this work is different from ours in that a partitioning is constructed, disallowing overlapping clusters.

Modeling flow through a network is another way to cluster a graph [4, 3]. MCL models flow through two alternating Markov processes, expansion and inflation. MCL has been widely used for clustering in biological networks but requires that the graph be sparse and only finds overlapping clusters in restricted cases. (α, β)-Clustering has no restrictions on the general structure of the graph and allows clusters of different sizes to overlap.

There has also been considerable work in finding communities on the web. Kumar et al. [13] approach the problem as one of finding bicliques as the cores of communities. While our approach can be adapted to find bicliques, we deal with more general community structures.

3 Combinatorics of (α, β)-clusters

In this section, we discuss various combinatorial properties of (α, β)-clusters including cluster overlap, containment and number of clusters.

Prior to doing so, we make a quick remark about the value of β. If $\beta < \frac{1}{2}$ then it is possible to have a cluster containing two disconnected components, i.e., a subset of vertices with a cut of size 0 could form a cluster. Imagine two cliques K_n with no edges in between them. If $\beta < \frac{1}{2}$ then these two disconnected cliques form one $(0, \frac{1}{2})$-cluster. Consequently, we insist that $\beta > \frac{1}{2}$. In that case, a cluster is necessarily connected; select any two vertices u and u' in the cluster, since u is adjacent to more than half of the cluster and so is u', there must be at least one vertex that they both neighbor. Thus, there is a path of length at most two between any two vertices in a cluster. We will use this fact later in some of our analysis. In this paper, we assume that $\beta > 1/2$ so that all clusters are connected, although it would be interesting to consider other restrictions that enforce connectedness.

Notation We use the following notation to describe our results. For a graph $G = (V, E)$, n denotes the number of vertices and m denotes the number of edges. For a subset of vertices $A \subseteq V$, $|A|$ denotes the number of vertices in A. $E(v, A)$ denotes the set of edges between a vertex v and a subset of vertices A. The neighbors of a vertex v are denoted by $\Gamma(v)$. The function $\tau(v) = \Gamma(v) \cup \Gamma(\Gamma(v))$ indicates all neighbors of path distance 1 or 2 from v.

Cluster Overlap Given two (α, β)-clusters A, B where $|A| \geq |B|$, we now determine the maximum size of the overlap, namely $|A \cap B|$. In the case where $\beta = 1$, $|A \cap B|$ can be no larger than $\alpha|B|$ (otherwise, there would be a vertex outside of B that is adjacent to more than α of B). Alternatively, in the case where $\alpha = 0$, $|A \cap B|$ must be 0. More generally, we seek a bound for arbitrary values of α and β. We express the overlap as the fraction of vertices in A, i.e., $\gamma = \frac{|A \cap B|}{|A|}$.

Proposition 1. *For two (α, β)-clusters, A and B, where $|A| \geq |B|$, an upper bound on the ratio of the intersection, $|A \cap B|$, to the larger one, $|A|$, is $\gamma = \min(1 - (\beta - \alpha\frac{|B|}{|A|}), \frac{\alpha}{2\beta - 1}\frac{|B|}{|A|})$. When $\beta - \alpha\frac{|B|}{|A|} > \frac{1}{2}$, $\frac{\alpha}{2\beta-1}\frac{|B|}{|A|}$ is the minimum and otherwise $1 - (\beta - \alpha\frac{|B|}{|A|})$ is the minimum.*

Cluster Containment Given that clusters can overlap, it is natural to ask if one cluster can be contained in another. In some circumstances, α and β may be such that clusters are contained in each other. For example, consider two cliques, C and D, each containing n vertices. Assume that each vertex in C is adjacent to two vertices in D. When $\beta = \frac{1}{2} + \frac{2}{n}$ and $\alpha = \frac{2}{n}$, $C \cup D$ is an (α, β) cluster that contains both C and D.

If we want to prevent our algorithm from finding clusters where one is contained in another, we can do so by requiring that the ratio of the largest to the smallest cluster is at most $\frac{1-\alpha}{1-\beta}$.

Corollary 1. *Let A and B be (α, β)-clusters and assume that $|B| \leq |A|$. If $\frac{|A|}{|B|} < \frac{1-\alpha}{1-\beta}$ then B can not be contained in A.*

The larger the gap between α and β, the larger the bound. For example, if $\alpha = 1/4$ and $\beta = 3/4$, then the larger cluster must be at least 3 times larger than the smaller before the smaller can be contained in the larger. Similarly, if $\alpha = 1/8$ and $\beta = 7/8$ then the ratio is 7.

Bounding the Number of $(\alpha, 1)$-clusters We next consider the problem of upper bounding the number of $(\alpha, 1)$-clusters. We give a superpolynomial bound on the number of clusters of a fixed size $s = f(n)$. More generally, it would be interesting to bound the number of possible (α, β)-clusters, but our analysis here is focused on cliques.

We wish to bound the number of $(\alpha, 1)$-clusters of size $s = f(n)$ in a graph $G = (V, E)$ where $|V| = n$. We know that no two clusters can overlap in more than αs vertices from Prop. 1.

Proposition 2. *Let $G = (V, E)$ where $|V| = n$. If $\mathcal{C}$ is the set of $(\alpha, 1)$-clusters of size s in G then $|\mathcal{C}| = O(\binom{n}{s}^{\alpha s+1})$.*

Proof. From Prop. 1, two clusters of size s can share at most αs vertices. Let us ignore the edges in the graph and consider clusters as subsets of vertices. Now we can say that every subset of size $\alpha s + 1$ must appear in at most one set in our collection. There are a total of $\binom{n}{s}$ subsets of size s and each of these subsets contains $\binom{s}{\alpha s+1}$ subsets of size $\alpha s + 1$. By simple combinatorics we can have at most $\binom{n}{\alpha s+1}/\binom{s}{\alpha s+1}$ clusters of size s. The bound $|\mathcal{C}| \leq \binom{n}{\alpha s+1}/\binom{s}{\alpha s+1} = O(\binom{n}{s}^{\alpha s+1})$ follows from Stirling's Approximation.$\square$ [6]

We note that this bound is tight when $\alpha = 0$ and when α approaches 1. If we let $\alpha = 0$ then the bound indicates that the number of clusters is at most $\frac{n}{k}$. This is tight because clusters cannot overlap at all. At the other extreme, consider the complement of the graph shown in Fig. 2. Let $\alpha = \frac{n-1}{n}$ and $\beta = 1$. For $k = n$ the bound on the number of clusters from our bound is 2^n. This number is realized since the set of clusters is $B = \{b_1 \ldots b_n | b_i = x_i \vee y_i\}$. $|B| = 2^n$ so the bound is tight in this case. We can construct a graph of this type for all α of the form $\frac{n-1}{n}$ so we have a tight exponential bound for these α values.

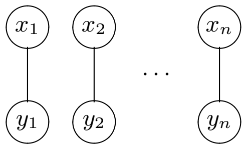

Fig. 2. A graph G where $\overline{G}$ has exponentially many clusters.

We believe that the bound given in Prop. 2 overcounts the number of clusters when $\alpha \leq \frac{1}{2}$. We note that our examples of graphs that meet the exponential bound all have $\alpha \geq \frac{1}{2}$. Consider the case where we have two $(\alpha, 1)$-clusters, A and B that overlap in αs vertices. Let D be a third cluster such that $|A \cap D| = |B \cap D| = \alpha s$ but $A \cap B \cap D = \emptyset$. This is allowed by the construction in Prop. 2. Let $u \in A \cap C$ and $v \in B \cap C$. Since $u, v \in C$ and $\beta = 1$ $(u, v) \in E$. However, u is already connected to $\alpha|B|$ in the form of $A \cap B$, so we have an α violation. Therefore, we counted D as an (α, β)-cluster when we should not have.

Another criticism of counting $(\alpha, 1)$-clusters with Prop. 2 is that edges are completely ignored. Consider K_4 where $s = 3$ and $\alpha = 1/3$. The bound allows 3 clusters of size 3. In reality, due to α violations, there are none.

4 An Algorithm for Finding Clusters with Champions

In this section, we make some restrictions to the general (α, β)-clustering problem, motivate these restrictions and then give an algorithm for finding clusters of

[6] This exactly corresponds to the construction of a Steiner System.

this restricted form. Specifically, we first justify a gap between internal density and external sparsity. Next, we introduce the notion of a champion of a cluster. Intuitively, a vertex champions a cluster if it has more affinity into the cluster than out of it. We then give a simple, deterministic algorithm for finding all (α, β)-clusters with ρ-champions in a graph, assuming that $\beta > \frac{1}{2} + \frac{\rho+\alpha}{2}$.

Gap Between Internal Density and External Sparsity To motivate a gap between internal density and external sparsity, consider Fig. 2. Observe that depending on the choice of α and β, the number of clusters may be exponential in the size of the graph. In practice, an algorithm that outputs more clusters than vertices is quite undesirable – especially given that social networks are massively large data sets. Thus, we seek a restriction that will reduce the number of clusters. The restriction considered in this paper is a large gap between β and $\alpha/2$.

Champions To motivate champions, observe that for $\overline{G}$ of G given in Fig. 2, each vertex in each cluster has as many neighbors outside the cluster as within it. There is no vertex that "champions" the cluster in the sense that many of its neighbors are in the cluster. For example, theoretical physicists form a community in part because there are some champions that have more friends that are theoretical physicists than not. Specifically, if *every* vertex in a subset A has as many neighbors out of A as into A, then it is arguable if A is really even a cluster. This motivates us to formally define the notion of a ρ-champion.

Definition 2. *A vertex $c \in C$ ρ-*champions *a cluster C if $|\Gamma(c) \cap V \setminus C| \leq \rho|C|$, for some $0 \leq \rho \leq 1$.*

Deterministic Algorithm We now claim that if $\beta > \frac{1}{2} + \frac{\rho+\alpha}{2}$ or $\alpha < (2\beta - 1)(\beta - \rho)$ then there are at most n clusters with ρ-champions and further that there is a simple deterministic algorithm for finding the clusters. In the following, we make the simplifying assumption that every cluster has the same size. The lemma can be suitably modified in the case of clusters of different sizes.

Lemma 1. *If either $\beta > \frac{1}{2} + \frac{\rho+\alpha}{2}$ or $\alpha < (2\beta - 1)(\beta - \rho)$ then there are at most n (α, β)-clusters of size s with ρ-champions.*

A large gap between β and $\frac{1}{2} + \frac{\alpha+\rho}{2}$ yields a simple algorithm for deterministically pinning down all the clusters. Let the input to the algorithm be α, β, the graph G and the size s of the clusters to be found.

Algorithm 1 Deterministic Clustering Algorithm, when $\beta > \frac{1}{2} + \frac{\alpha+\rho}{2}$.

```
1: Input: $\alpha, \beta, s, G$
2: for each $c \in V$ do
3:     $C = \emptyset$
4:     for each $v \in \tau(c)$ do
5:         if $|\Gamma(v) \cap \Gamma(c)| \geq (2\beta - 1)s$ then add $v$ to $C$.
6:     end for
7:     if $C$ is an $(\alpha, \beta)$-cluster then output $C$.
8: end for
```

The following lemma shows that if v and c share sufficiently many neighbors, then v is necessarily part of the cluster C that c champions.

Lemma 2. *Let C be an (α, β)-cluster and c its ρ-champion. Let $\beta > \frac{1}{2} + \frac{\rho+\alpha}{2}$. A vertex v is in the cluster C if and only if $|\Gamma(v) \cap \Gamma(c)| \geq (2\beta - 1)|C|$.*

When the size of the cluster is fixed, Lemma 2 also implies that C is unique. Since we can bound the number of clusters of each size to n, we can also bound the total number of (α, β)-clusters with ρ-champions to be $O(n^2)$. Additional bounds to guarantee uniqueness when the size of the cluster is allowed to vary can be easily obtained.

Consequently, we have the following theorem.

Theorem 1. *Let $G = (V, E)$ be a graph and $\beta > \frac{1}{2} + \frac{\rho+\alpha}{2}$. Algorithm 1 exactly finds all (α, β)-clusters of size s that have ρ-champions in time $O(m^{0.7}n^{1.2} + n^{2+o(1)})$.*

To interpret the theorem, when clusters have ρ-champions where $\rho = \alpha$, a separation of $\frac{1}{2}$ is needed between β and α in order for the algorithm to find all the clusters. The worse the champion, the fewer the number of valid α and β values where the algorithm is guaranteed to succeed. For example, if $\rho = 3\alpha$ then the gap between β and α must be larger, namely $\beta > 2\alpha + \frac{1}{2}$.

The running time follows from the fact that the algorithm computes the number of neighbors that each pair of vertices share. We can precompute $|\Gamma(v_i) \cap \Gamma(v_j)|$ for all $i, j \in V$ by noting that if A is the adjacency matrix of G then $(A^T A)_{i,j} = |\Gamma(v_i) \cap \Gamma(v_j)|$. Yuster and Zwick [19] show that matrix multiplication can be performed in $O(m^{0.7}n^{1.2}+n^{2+o(1)})$ time. Checking the α, β conditions requires $O(m^{0.7}n^{1.2} + n^{2+o(1)} + n(\tau(c) + n)) = O(m^{0.7}n^{1.2} + n^{2+o(1)})$ time.

In the case G is a typical social network, G has small average degree and A is a sparse matrix . If we let d be the average degree of the graph then $m = dn/2$. Thus, for small d, the algorithm runs in $O(d^{0.7}n^{1.9} + n^{2+o(1)})$ time.

5 Experiments

We introduced the notion of a ρ-champion and gave an algorithm for finding (α, β)-clusters with ρ-champions. A natural next question is: Do (α, β)-clusters with ρ-champions even exist in real graphs? And, if so, do most (α, β)-clusters have ρ-champions? To answer the first question, we study three real networks induced by co-authorship among high energy physicists, co-authorship among theoretical computer scientists, as well as a real, online social networking site known as LiveJournal. To answer the second question, we need an algorithm that can find (α, β)-clusters independent of whether they have ρ-champions. The best previous algorithm for this problem is due to Tsukiyama et al [18] that finds all maximal cliques in a graph, i.e., all $(\alpha, 1)$-clusters.

Our experiments uncovered a few surprising facts. First, our simple algorithm was able to find $\approx 90\%$ of the maximal cliques in these graphs where $\alpha \leq \frac{1}{2}$. Next, among the cliques we missed, we found that there was no strong ρ-champion.

Finally, our algorithm was orders of magnitudes faster than Tsukiyama's. In short, our algorithm more quickly discovers clusters of practical interest, i.e., small α, small ρ and large β.

Data Sets and Tsukiyama's Algorithm As mentioned, three data sets were used: the High Energy Physics Theory Co-Author graph (HEP) [7], the Theory Co-Author graph (TA) and a subset of the LiveJournal graph (LJ) [14]. LiveJournal is a website that allows users to create weblogs and befriend other LiveJournal users. We obtained a crawl of a subset of this site. In our graph the vertices correspond to usernames and the edges to friendships. In the Theory and HEP Co-Author graphs, authors are vertices and edges correspond to co-authors. Some basic statistics about these graphs are given below.

Data set	Size	Avg Deg.	Min Deg.	Max Deg.	Avg $\tau(v)$	Min $\tau(v)$	Max $\tau(v)$
HEP	8,392	4.86	1	63	40.58	2	647
TA	31,862	5.75	1	567	172.85	1	8,116
LJ	581,220	11.68	1	1,980	206.15	6	15,525

Tsukiyama's algorithm finds all maximal cliques in a graph via an inductive characterization: given the maximal cliques involving the first i vertices, the algorithm shows how to extend this set to the maximal cliques involving the first $i + 1$ vertices. The algorithm's running time is polynomial in the size of the graph and the number of maximal cliques. More details can be found in [18].

Results In this section we present numerical results comparing the ground truth of Tsukiyama's Algorithm with our Algorithm 1. For this experiment we were only interested in cliques of size 5 or larger with α values of 0.5 or less. These are the cliques that Algorithm 1 could reasonably find. We pruned the output of Tsukiyama's algorithm to contain just these cliques. We found that the HEP graph had a total of 126 cliques satisfying this definition; our algorithm found 115, or 91%. Similarly, the Theory graph had 854 cliques and our algorithm found 797 or 93%. In Figures 3, 4 and 5 we show the α and ρ distributions of the cliques found by Tsukiyama compared with the α distribution of those found by Algorithm 1. When a bar is cut off a number is placed next to the bar to indicate the true value. Bars have only been cut off when Algorithm 1 found all of the cliques that Tsukiyama's Algorithm found.

In both Theory and HEP, the distribution of ρ-values among the clusters found is exactly as our theorems predict, i.e., ρ is almost always less than $\frac{1}{2}$. And, interestingly, for LiveJournal, the distribution of ρ-values is better than our theorems predict in that we find 876 clusters where ρ is larger than 1/2. Indeed, we find some clusters where ρ is as large as 1.2.

Timing Our experiments were run on a machine with 2 dual core 3 GHz Intel Xeons and 16 Gigabytes of RAM. We report wall-clock time for all of our experiments.

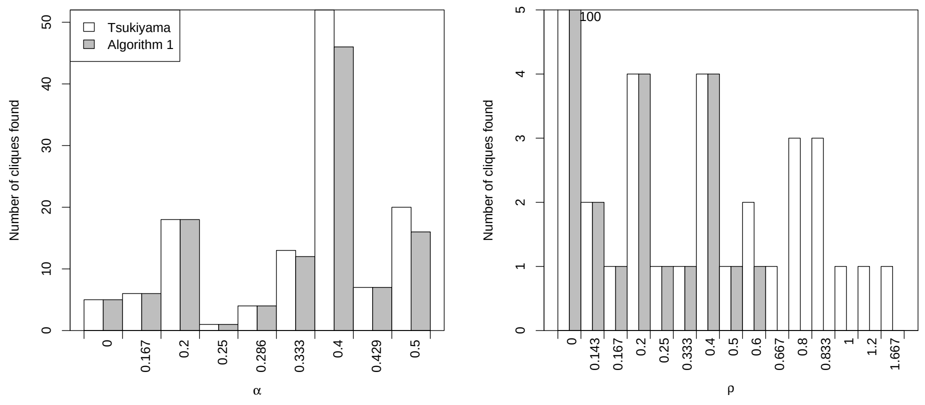

Fig. 3. For the HEP graph, α and ρ distributions are shown for the cliques found by Tsukiyama's algorithm vs. the cliques found by Algorithm 1. Our algorithm found 115 out of 126 maximal cliques.

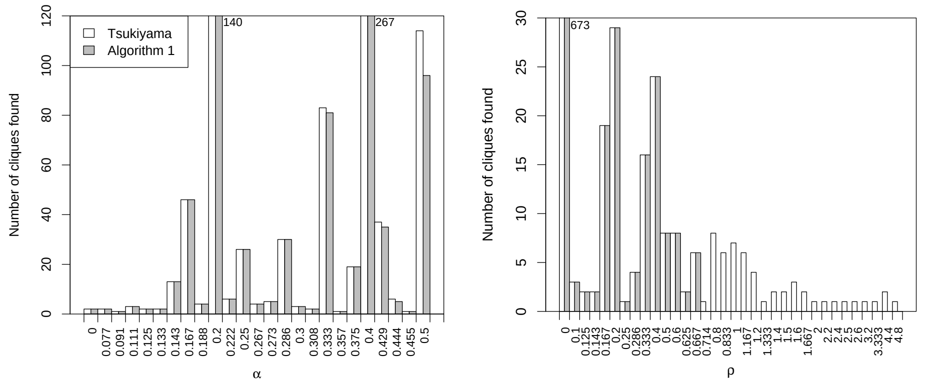

Fig. 4. For the TA graph, α and ρ distributions are shown for the cliques found by Tsukiyama's Algorithm vs. the cliques found by Algorithm 1. Our algorithm found 797 out of 854 maximal cliques.

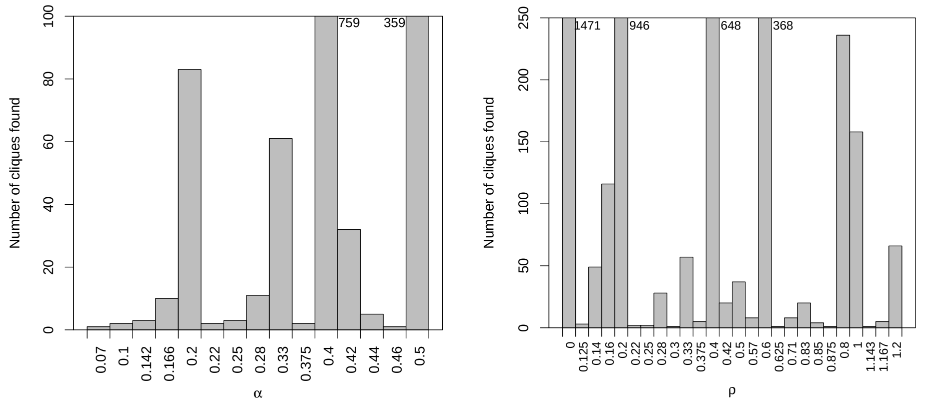

Fig. 5. α and ρ distributions for Algorithm 1 for the LJ graph. Tsukiyama's algorithm was too slow to generate ground truth results for this graph. Our Algorithm 1 found 4289 cliques.

Experiment	HEP	TA	LJ
Alg. 1, $(\alpha, \beta) = (0.5, 1)$	8 secs	2 min 4 sec	3 hours 37 min
Tsukiyama	8 hours	36 hours	N/A

Note that after one week of running Tsukiyama et al.'s algorithm on the LJ data set, the algorithm did not complete. In fact, only 6% of the graph had been considered. However, our Algorithm 1 found 4289 cliques of size greater than 5 with $\alpha \leq .5$ in a few hours.

6 Summary and Future Work

We introduced a new criterion for discovering overlapping clusters that captures intuitive notions of internal density and external sparsity. We also give a deterministic algorithm for discovering clusters assuming each cluster has a champion and there is a sufficiently large gap between internal density and external sparsity. Experiments indicate that our algorithm succeeds in finding good clusters.

While we assume $\beta > \frac{1}{2}$ to enforce cluster connectedness, we believe this assumption is too strong. In particular, a subgraph can be connected while β is much less than $\frac{1}{2}$, e.g., a long cycle. Furthermore, $\beta > \frac{1}{2}$ precludes our algorithm from finding very large clusters because the average degree of a vertex in a social network is typically small.

Generalizations of (α, β)-clustering to weighted and directed graphs are also of interest. Our work assumes that edges are not weighted. But in real social networks, there is a strength of connectivity between pairs of individuals corresponding to how often they communicate. This weight could be exploited in the discovery of close-knit communities. In addition, some networks induce directed graphs. For example, the direction of edges in email networks plays an important role in definining communities otherwise spam mailers would belong to every cluster.

Decentralized and streaming algorithms are essential for modern networks such as instant messaging or email graphs. In particular, it is often difficult to even collect the graph in one centralized location [11]. Thus, algorithms that can compute clusters with only local information are needed. Further, given that social networks are dynamic data sets, i.e., vertices and edges come and go, streaming graph clustering algorithms are an important avenue for future research.

Acknowledgments We are grateful to Mark Sandler for early discussions, Ying Xu for the LiveJournal crawl and the anonymous reviewers for their useful comments.

References

1. J. Abello, M. G. C. Resende, and S. Sudarsky. Massive quasi-clique detection. *LATIN: Latin American Symposium on Theoretical Informatics*, 2286:598–612, 2002.

2. W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, STOC'2000 (Portland, Oregon, May 21-23, 2000)*, pages 171–180, 2000.
3. S. Van Dongen. A new cluster algorithm for graphs. Technical report, Universiteit Utrecht, July 10 1998.
4. G. W. Flake, R. E. Tarjan, and K. Tsioutsiouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 1(4):385–408, 2004.
5. R. E. Gomory and T. C. Hu. Multi terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9:551–571, 1961.
6. E. Hartuv and R. Shamir. A clustering algorithm based on graph connectivity. *IPL: Information Processing Letters*, 76:175–181, 2000.
7. KDD Cup'03 HEP-TH. http://www.cs.cornell.edu/projects/kddcup/datasets.html.
8. D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988.
9. R. Kannan, S. Vempala, and A. Vetta. On clusterings — good, bad and spectral. *Proceedings of the 41th Annual Symposium on Foundations of Computer Science*, pages 367–377, 2000.
10. G. Karypis and V. Kumar. A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. *J. Parallel Distrib. Comput.*, 48(1):71–95, 1998.
11. D. Kempe and F. McSherry. A decentralized algorithm for spectral analysis. In *Proceedings of the thirty-sixth annual ACM Symposium on Theory of Computing (STOC-04)*, pages 561–568, New York, June 13–15 2004. ACM Press.
12. V. Krebs. Uncloaking terrorist networks. *First Monday*, 7(4), 2002.
13. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1481–1493, May 1999.
14. LiveJournal. www.livejournal.com.
15. M. E. J. Newman. Modularity and community structure in networks. *National Academy of Sciences*, 103:8577–8582, February 2006.
16. J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.
17. D. A. Spielman and S. Teng. Spectral partitioning works: Planar graphs and finite element meshes. *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, 37:96–105, 1996.
18. S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa. A new algorithm for generating all the maximal independent sets. *SIAM J. Comput*, 6(3):505–517, 1977.
19. R. Yuster and U. Zwick. Fast sparse matrix multiplication. *ACM Transactions on Algorithms*, 1(1):2–13, July 2005.