Finding Endogenously Formed Communities

Maria-Florina Balcan* Christian Borgs† Mark Braverman ‡ Jennifer Chayes§
Shang-Hua Teng¶

March 2, 2012

Abstract

A central problem in e-commerce is determining overlapping communities (clusters) among individuals or objects in the absence of external identification or tagging. We address this problem by introducing a framework that captures the notion of communities or clusters determined by the relative affinities among their members. To this end we define what we call an affinity system, which is a set of elements, each with a vector characterizing its preference for all other elements in the set. We define a natural notion of (potentially overlapping) communities in an affinity system, in which the members of a given community collectively prefer each other to anyone else outside the community. Thus these communities are endogenously formed in the affinity system and are "self-determined" or "self-certified" by its members.

We provide a tight polynomial bound on the number of self-determined communities as a function of the robustness of the community. We present a polynomial-time algorithm for enumerating these communities. Moreover, we obtain a local algorithm with a strong stochastic performance guarantee that can find a community in time nearly linear in the of size the community (as opposed to the size of the network).

Social networks and social interactions fit particularly naturally within the affinity system framework – if we can appropriately extract the affinities from the relatively sparse yet rich information from social networks and social interactions, our analysis then yields a set of efficient algorithms for enumerating self-determined communities in social networks. In the context of social networks we also connect our analysis with results about (α, β)-clusters introduced by Mishra, Schreiber, Stanton, and Tarjan [16, 17]. In contrast with the polynomial bound we prove on the number of communities in the affinity system model, we show that there exists a family of networks with superpolynomial number of (α, β)-clusters.

1 Introduction

Affinity Systems The problem of identifying endogenously[1] formed overlapping communities or clusters arises in many contexts within e-commerce: finding overlapping communities in a social network, clustering

*School of Computer Science, College of Computing, Georgia Institute of Technology, Atlanta, Georgia.
†Microsoft Research, New England, Cambridge, MA.
‡Princeton University and University of Toronto
§Microsoft Research, New England, Cambridge, MA.
¶Computer Science Department, University of Southern California.
[1]endogenous: growing or developing from within; originating within.

retail products using collaborative filtering, clustering documents using citation information, classifying videos using viewing logs, etc. In such settings one needs to cluster the set of objects into meaningful, potentially overlapping subsets by only using information about relations between the objects. In this paper we develop the notion of an affinity system to model these scenarios.

An affinity system is a collection of elements with a set of "preferences" each of these elements has over other elements within the system. These preferences may be expressed as a vector of rankings, or, more generally, as a vector of non-negative weights representing affinities. For example, when clustering videos, affinities may represent the likelihood of the videos to be co-watched, with videos that are co-watched more often "ranking" each other higher. When clustering documents, a document will "prefer" documents it cites over documents it doesn't.

Perhaps the most natural application of affinity systems is to the study of social networks. Social interaction is often determined by affinities among the members. For example, in daily life, we often stay more in touch with people we like more. When we go to a conference, we often hang out more with people with whom we share more interests. Therefore, these social interactions, and their manifestations as online social networks fit well within the affinity system paradigm.

Endogenously Formed Communities in Affinity Systems A central question concerning groups of individuals, documents, products, etc., is how to determine communities, or *overlapping* clusters that capture the coherence among their members. For example, in the context of retail products discussed above, it may be useful to automatically "tag" the products with multiple categories for subsequent personalized marketing. In the context of professional networks, a person may belong to multiple explicit or implicit communities, for example a scientist may simultaneously belong to the community of Economists and the community of Computer Scientist. The question of finding overlapping communities is closely related to the very well studied question of clustering [10], but is much more general, since now elements may (and will) belong to multiple communities.

In this paper we formalize a natural notion of self-determined community and develop efficient algorithms to identify overlapping communities of this type as well as general bounds on the number of such communities. Self-determined communities correspond to subsets that collectively prefer each other more than they prefer those outside the subset, where preference is defined by the rankings or weights of the affinity system. These communities are endogenously formed in the affinity system. What is particularly nice about this formulation is that we do not require that the subsets be of pre-specified sizes. For example, a solution of the flexible capacity roommate problem would group together people who prefer living with each other to living with anyone else in another room. Switching to the context of social and professional networks, an academic community can be viewed as a group of scholars which appreciates the work of others in the community to that of the work of people outside their community. In all these cases, the overlapping communities or clusters are self-certified or self-determined.

More formally, we study the mathematical structure of self-determined communities in an affinity system and design efficient algorithms for discovering them. In our most basic model, we have n members $V = \{1, ..., n\}$ in an affinity system, and we assume each member i states a strict ranking π_i of all members in the order of her preferences. To evaluate whether a subset S of size $|S| = k$ is a good community, imagine that each member $s \in S$ casts a vote for each of its k most preferred members $\pi_s(1 : k)$. The number of votes that member i receives, $\phi_S(i) = |\{i \in \pi_s(1 : |S|)|s \in S\}|$, is the collective preference given by S. We say S is *self-determined* if everyone in S receives more votes from S than everyone outside S.

Different self-determined communities may have different degree of coherence or robustness depending on

both the fraction of votes received by the community members as well as the gap between the fraction of votes received by the community members and the non-community members. To capture this, we say S is a (θ, α, β) *self-determined community*, for $0 \leq \beta < \alpha \leq 1$ and $\theta > 0$ if

- each member $s \in S$ casts a vote for each of its $\theta|S|$ most preferred members $\pi_s(1 : \theta|S|)$.
- for each $i \in S$, the amount of vote i receives, $\phi_S^{(\theta)}(i) = |\{i \in \pi_s(1 : \theta|S|)|s \in S\}|$, is at least $\alpha|S|$.
- for each $j \notin S$, the amount of vote j receives, $\phi_S^{(\theta)}(j) = |\{j \in \pi_s(1 : \theta|S|)|s \in S\}|$, is at most $\beta|S|$.

We start by analyzing how many communities can exist in an affinity system. Interestingly, we show that for constants α, β, θ we have a polynomial bound of $n^{O(\log(1/\alpha)/\alpha)}$ on the number of (θ, α, β)-self-determined communities. Our analysis, using probabilistic methods, also yields a polynomial-time algorithm for enumerating these communities. Moreover, we show that our bound is nearly tight, by exhibiting an affinity system with $n^{\Omega(1/\alpha)}$ (θ, α, β)-self-determined communities.

We then present a local community finding algorithm that is very efficient for an interesting range of parameters. This algorithm, when given robustness parameters θ, α, β, and a member $v \in V$, either returns a (θ, α, β)-self-determined community of size t in time $O(f(\alpha, \beta, \theta) \cdot t \log t)$ or an empty set. The algorithm satisfies the following performance guarantee: if $\alpha > 1/2$, if v is chosen uniformly at random from a (θ, α, β)-self-determined community S, then with probability $\Omega(2\alpha - 1)$, our local algorithm will successfully recover S and so in time dependent only (and nearly) on the $|S|$ and not on the size of the entire affinity system. As a consequence of this analysis, we can show that in the (natural) case when $\alpha > 1/2$ we obtain a *near-linear* algorithm for finding all self-determined communities, substantially improving on the polynomial-time guarantee discussed above. Quasi-linear local algorithms are particularly important in the context of studying internet-scale networks, where even quadratic-time algorithms are not feasible, and where one sometime does not have access to the entire network but only to a local portion of it. The quasi-linear algorithm is one of our main technical contributions, as its techniques can potentially be used to convert other polynomial cluster-detection algorithms into local quasi-linear algorithms – at least in the average case.

We also study *multi-facet affinity systems* where each member may have a number of different rankings of other members. For example, member i may have two rankings $\pi_{i,fun}$ and $\pi_{i,science}$, where first ranks members by how much fun i thinks they are and the second ranks them according to academic affinity. In this context, we say S is a self-determined community if there exists a vector of choices of rankings (in this case, in $\{fun, science\}^{|S|}$) such that if members vote according to their associated choice, the resulting votes self-certify S. We prove that if each member has a constant number of rankings, all our results can be extended, even though there could be exponential number of combinations of rankings.

Our results can be extended to weighted affinity systems where the affinities of each member are given by a numerical weighting rather than just an ordinal ranking. For example, member i may give her most preferred member weight 1, next two preferred members weight 0.7, next one weight 0.5, and so on. A weighted affinity system can be expressed as $A = \{V, a_1, ..., a_n\}$, where a_i is a n-dimensional vector $a_i = (a_{i,1}, ..., a_{i,n})$ and $0 \leq a_{i,j} \leq 1$ specifies the degree of affinity that i has for j. One can naturally define (θ, α, β)-self-determined communities for weighted affinity systems. The only requirement is that members are only allowed to cast votes up to a total weight of θt when voting for a community of size t, while respecting the affinity system. We show that all our bounds and algorithmic results extend to weighted affinity systems with only a slight loss in the parameters.

Endogenously Formed Communities in Social Networks Our general formulation enables us to shed light on the challenging task of defining and finding overlapping communities in social networks [19, 17, 16, 14, 15]. Typically, a social network can be viewed as graph $G = (V, E)$, where the edges could be either undirected (e.g., the Facebook social network determined by friend list) or directed (e.g. the Twitter network). An edge could be unweighted or weighted (e.g., the Skype phone-call networks or the Facebook network based on the number of times that one person writes on the wall of others).

It turns out that a social network can be realized as a projection of an affinity system. Indeed, although our affinity systems are typically dense, their projections as social networks can be very sparse as are many observed social networks. We can think of our observed social network interactions as being induced (in various ways) by the underlying latent set of affinities. To be precise, given a social network $G = \{V, W, w\}$ with weights $w = (w_{ij})$, we would like to recover the communities in the original affinity system. A natural way to do this is to lift the social network back to an affinity system $A = \{V, a_1, ..., a_n\}$ and then to solve the problem in the lifted system. For example, several natural approaches for lifting based on different beliefs about how the social network may have emerged from an underlying set of affinities include:

1. *Direct Lifting*: One can directly lift to an affinity system by defining $a_{i,j} = w_{i,j}$ if $(i, j) \in E$, otherwise $a_{i,j} = 0$ (we assume WLOG that $w_{i,j} \in [0, 1]$)).
2. *Shortest Path Lifting*: If $G = (V, E)$ is an unweighted social network, and the shortest path distance from i to j is $d_{i,j}$, one may define $a_{i,j} = 1/d_{i,j}$. The shortest path lifting can be extended to weighted cases by appropriated normalization.
3. *Personal Page Rank Lifting*: Let p_i be the personal PageRank vector [2] of vertex i, we define $a_{i,j} = p_{i,j}/\max(p_i)$.
4. *Effective Resistance Lifting*: Let $r_{i,j}$ be effective resistance of from i to j by viewing G a network of resistors, using $1/w(e)$ as the resistance of $e \in E$ [9], we define $a_{i,j} = min_k(r_{i,k}/r_{i,j})$.

Each style of lifting corresponds to a particular belief on how this social network may have emerged from a latent underlying affinity system. For instance, Direct Lifting corresponds to the belief that a social network $G = (V, E)$, such as the Twitter network, arose from a latent affinity $A' = (V, \{a'_1, ..., a'_n\})$ by a process in which each member i connects to the d_i top most elements according to the affinity system A'. In other words, i follows j, i.e., (i, j) is a directed edge in this social network, if j is among the d_i top most elements of i according to A'. Similarly, one can think of Shortest Path Lifting as corresponding to the belief that the social network serves as an approximate spanner of the underlying affinity system [18], and Effective Resistance Lifting corresponds to the belief that a social network is approximately based on some spectral sparsification of those underlying affinities [20].

Given a social network, once we derive a corresponding affinity system A, we may use our notion of self-determined community and apply our algorithms and analysis to obtain communities in the original network. From our analysis for affinity systems, we immediately obtain that there is a polynomial number of such communities in a social network, and they can be enumerated in polynomial time.

We note that while the input social network is potentially very sparse, appropriate lifting procedures can produce an affinity system better reflecting the true relationships between entities. Moreover, many can be performed locally, allowing for our local algorithm to determine meaningful communities especially efficiently.

We also note that our study of multi-facet affinity systems allows us to model and analyze communities in more complex social networks – such as such Google+ with circles which enable its users to share different things with different circles of people. This extension may also enable us to model interdisciplinary subfields according to scientific works or interactions.

Self-determined Communities and (α, β)-clusters In this paper, we also provide several new results for communities defined as (α, β)-clusters, a notion introduced by Mishra, Schreiber, Stanton, and Tarjan [16] for analyzing (unweighted) social networks. In their definition, S is an (α, β)-cluster (for $\alpha > \beta$) if for every $i \in S$, the number of neighbors coming from S is at least $\alpha|S|$ and for every $j \notin S$, the number of neighbors coming from S is at most $\beta|S|$. We prove that there exists a family of networks with superpolynomial number of (α, β)-clusters. For instance, if $\alpha = 1$ and $\alpha - \beta = 0.01$, then in $G(n, 1/2)$, the Erdös-Renyi random graph with $1/2$ edge probability, the expected number of (α, β)-clusters is $n^{\Omega(\log n)}$. We also show that under the assumption that the planted clique problem is hard, even finding a *single* (α, β)-cluster is computational hard. Interestingly, our notion of communities in social networks obtained via direct lifting is quite similar to the notion of (α, β)-clusters, with the only difference that we bound the total amount of votes a member may cast.[2] This twist seems to be essential to obtain only a polynomial number of such communities and to be able to enumerate them in polynomial time.

Related Work Problems of clustering or grouping data (based on network or pairwise similarity information or ranked data) have been extensively studied in many different fields. The classic goals have been to produce either a partition or a hierarchal clustering of the data [10, 6, 4, 8, 5, 13]. With the rise of online social networks, there has been significant recent interest in identifying overlapping clusters, or communities, in networks ranging from professional contact networks to citation networks to product-purchasing networks, with many heuristics and optimization criteria being proposed [14, 15, 19, 16, 17, 12]. However, much of this work has disallowed natural communities such as those containing highly popular nodes [21, 22, 14, 15] or not given general guarantees on the number or computation time needed to find all overlapping communities meeting natural criteria [7, 16, 17, 12]. By contrast, our new formalization leads to natural communities and efficient algorithms for identifying all such communities. Additionally, our model allows us to deal with asymmetries in the input in a very natural way.

Independently, in recent work [3] consider several assumptions (that are between worst case and average case) concerning community structure and provide efficient algorithms in these settings. Remarkably, while their setting is somewhat different from ours some of their algorithms are similar in spirit.

2 Preliminaries and Notation

In our most basic model, we consider an affinity system with n members $V = \{1, ..., n\}$ and assume that each member $i \in V$ states a strict ranking π_i of all members in the order of her preferences. Let $\Pi = \{\pi_1, \ldots, \pi_n\}$. For $t > 0$, $S \subseteq V$, $i \in V$ we denote by $v_S^t(i)$ the number of members in S that place i among the topmost t elements of their preference list. That is $v_S^t(i) = |\{s \in S | i \in \pi_s(1:t)\}|$. For $\theta > 0$, we let $\phi_S^\theta(i) := v_S^{\lceil \theta |S| \rceil}(i)$. We define a natural notion of self-determined community as follows:

[2] For example, for the direct lifting of G, the notion of the community we obtain is as follows: S is a (θ, α, β)-self-determined community in G if every $i \in S$ receives at least $\alpha|S|$ collective vote and everyone not in S receives at most $\beta|S|$ collective vote. If G is unweighted, $(i, j) \in E$, $i \in S$, and d_i is the out-degree of i, then one way to set up the affinity system is to let i contribute $\min(1, \theta|S|/d_i)$ to the collective vote of j.

Definition 1 *Given three positive parameters θ, α, β, where $\beta < \alpha \leq 1$ and an affinity system (V, Π) we say that a subset S of V is an (θ, α, β)* self-determined *community with respect to (V, Π) if we have both*

(1) For all $i \in S$, $\phi_S^\theta(i) \geq \alpha\,|S|$.

(2) For all $j \notin S$, $\phi_S^\theta(j) \leq \beta\,|S|$.

Throughout the paper, we will denote by $\gamma = \alpha - \beta$. Fixing θ, we say that "i votes for j with respect to a subset S" if $j \in \pi_i(1 : \lceil \theta\,|S| \rceil)$. When S is clear from the context we say that i votes for j.

Note that communities *may overlap*. As a simple example, assume we have two sets A_1 and A_2 of size $n/2$ with $n/8$ nodes in common (representing, say, researchers in Algorithms and researchers in Complexity). Assume each node in $A_i \setminus A_j$ ranks first the nodes in A_i and then the nodes in A_j and that each node in $A_i \cap A_j$ ranks the nodes in $A_i \cup A_j$ arbitrarily. Then each A_i is a $(1, 3/4, 1/4)$ self-determined community.

We also consider (more general) weighted affinity systems, where the preferences of each member i involve numerical weightings (degrees of affinity) rather than just an ordinal ranking. A weighted affinity system is expressed as $A = \{V, a_1, ..., a_n\}$, where a_i is a n-dimensional vector $a_i = (a_{i,1}, ..., a_{i,n})$ and $0 \leq a_{i,j} \leq 1$ specifies the degree of affinity that i has for j. For example, i may give her top-ranked node a weight of 1, she might have a tie between its second and third-ranked nodes giving both a weight of 0.7, and so on. If member i chooses not to vote for a given node, this can be modeled by giving that node a weight of 0.

We can naturally extend our notion of (θ, α, β)-self-determined communities to weighted affinity systems. In the definition of voting, the only requirement is that members are only allowed to cast votes up to a total weight of θt when voting for a community of size t. For example, to evaluate whether a subset S is a good community, each member $s \in S$ casts a weighted vote as follows: s determines a prefix of the weights (sorted from highest to lowest) of total value $\theta|S|$ and zeros out the rest. If there are ties at the boundary, a natural conversion is to scale down the weights of those nodes just at the boundary to make the sum exactly equal to $\theta|S|$. In general, we denote the resulting vector (after capping the amount of vote a member casts when voting for a community of size t) as $\mathrm{a}_s^{\theta|S|}$. The amount of the weight that member $i \in V$ receives from S is $\mathrm{a}_S^\theta(i) = \sum_{s \in S} \mathrm{a}_{s,i}^{\theta|S|}$. Given these, we can define an (θ, α, β) *weighted self-determined* community as follows:

Definition 2 *Given $\theta, \alpha, \beta \geq 0$, $\beta < \alpha \leq 1$ and an weighted affinity system (V, A) we say that a subset S of V is an (θ, α, β)* weighted self-determined *community with respect to (V, A) if we have both*

(1) For all $i \in S$, $\mathrm{a}_S^\theta(i) \geq \alpha\,|S|$.

(2) For all $j \notin S$, $\mathrm{a}_S^\theta(j) \leq \beta\,|S|$.

We note that given an (weighted) affinity system and a set S we can *test* in time polynomial in n whether a proposed set S is a (θ, α, β)-self-determined community or not. Also, fixing a (θ, α, β)-self-determined community S, one can easily show that there exists a multiset U of size $k(\gamma) = 2\log(4n)/\gamma^2$ such that the set of elements i voted by at least a $(\alpha - \gamma/2)$ fraction of U (or in the weighted case, the set of elements i receiving $(\alpha - \gamma/2)|U|$ total vote from U) is identical to S. This then implies a very simple quasi-polynomial procedure for finding all self-determined communities, as well as an $n^{O(\log n/\gamma^2)}$ upper bound on the number of (θ, α, β)-self-determined communities. (See Appendix A.1 for details).

In this paper we present a *multi-stage* approach for finding an unknown community in an affinity system that provides much better guarantees for interesting settings of the parameters. At a generic level, this algorithm takes as input information I about an unknown community S and outputs a list $\mathcal{L}$ of subsets of V s.t. if information I is correct with respect to S, then with high probability $\mathcal{L}$ contains S. This algorithm has two main steps: it first generates a list $\mathcal{L}_1$ of sets S_1 s.t. at least one of the elements in $\mathcal{L}_1$ is a rough approximation to S in the sense that S_1 nearly contains S and it is not much larger than S. In the second step, it runs a purification procedure to generate a list $\mathcal{L}$ that contains S. (See Algorithm 1.) Both steps have to be done with care by exploiting properties of self-determined communities and we will describe in detail in the following sections ways to implement both steps of this generic scheme. We also discuss how to adapt this scheme for outputting a self-determined community in a local manner, for enumerating all self-determined communities, as well as extensions to multi-facet affinity systems and applications of our analysis to social networks.

Algorithm 1 A generic algorithm for identifying an unknown community S

Input: Preference system (V, Π), information I about an unknown community S.

(1) Using information I to generate a list $\mathcal{L}_1$ of sets S_1 s.t. at least one of the elements in $\mathcal{L}_1$ is a rough approximation to S.

(2) Run a purification procedure to generate a list $\mathcal{L}$ s.t. at least one of the elements in $\mathcal{L}$ is identical to S.

(3) Remove from the list $\mathcal{L}$ all the sets that are not self-determined communities.

Output: List of self-determined communities $\mathcal{L}$.

3 Finding Self-determined Communities

In this section we show how to instantiate the generic Algorithm 1 if the information we are given about the unknown community S is its size and the parameters θ, α, and β. We show that this leads to a polynomial time algorithm in the case where θ, α, and β are constant. We start with a structural result showing that for any self-determined community S there exist a small number of community members s.t. the union of their votes contains almost all S.

Lemma 1 *Let S be a (θ, α, β)-self-determined community. Let $\gamma = \alpha - \beta$, $M = \log(16/\gamma)/\alpha$. There exists a set U, $|U| \leq M$ s.t. the set $S_1 = \{i \in V | \exists s \in U, i \in \pi_s(1 : \theta|S|)\}$ satisfies $|S \setminus S_1| \leq (\gamma/16)|S|$.*

Proof: Note that any subset $\tilde{S}$ of S receives a total of at least $\alpha|\tilde{S}||S|$ votes from elements of S, which implies that for any such $\tilde{S}$ there exists $i_{\tilde{S}} \in S$ that votes for at least $\alpha|\tilde{S}|$ members of $\tilde{S}$. Given this, we find the desired elements $i_1, \ldots, i_M \in S$ greedily one by one. Formally, let $S_1 = S$. Let $i_1 \in S$ be an element that votes for at least a $\alpha|S_1|$ elements in S_1. Let S_2 be the set S minus the set of elements voted by i_1. In general, at step $l \geq 2$, there exists $i_l \in S$ that votes by at least a α fraction of S_l. Let S_{l+1} be the set S_l minus the set of elements voted by i_l. We clearly have $|S_{i+1}| \leq (1-\alpha)^i|S_1|$, so $|S_{M+1}| \leq (\gamma/16)|S_1|$ for $M = \log(16/\gamma)/\alpha$. By construction the set $U = \{i_1, \ldots, i_M \in S\}$ satisfies the desired condition. ■

Given Lemma 1, we can use the following procedure for generating a list that contains a rough approximation to S which covers at least a $1 - \gamma/16$ fraction of S and whose size is at most $\log(16/\gamma)|S|$.

Algorithm 2 Generate rough approximations

Input: Preference system (V, Π), information I (parameters θ, α, β, size t).

- Set $\mathcal{L} = \emptyset$, $\gamma = \alpha - \beta$, $k_1(\theta, \alpha, \gamma) = \log(16/\gamma)/\alpha$.
- Exhaustively search over all subsets U of V of size $k_1(\theta, \alpha, \gamma)$; for each set U add to the list $\mathcal{L}$ the set $S_1 \subseteq V$ of points voted by at least an element in U (i.e., $S_1 = \{i \in V | \exists s \in U, i \in \pi_s(1 : \theta t)\}$).

Output: List of sets $\mathcal{L}$.

We now describe a lemma that will be useful for analyzing the purification step, suggesting how we convert a rough approximation to S into a list of candidate much-closer approximations to S.

Lemma 2 *Fix a (θ, α, β)-self-determined community S. Let $\gamma = \alpha - \beta$, $t = |S|$, and $S_1 \subseteq V$, $|S_1| = M\theta t$ s.t. $|S \setminus S_1| \leq \gamma t/16$. Let U be a set of k points drawn uniformly at random from $\tilde{S} = S \cap S_1$. Let S_2 be the subset of points in S_1 that are voted by at least an $\alpha - \gamma/2$ fraction of nodes in U, i.e., $S_2 = \{i \in S_1 | v_U^{\theta t}(i) \geq (\alpha - \gamma/2)|U|\}$. If $k = 8\log(32\theta M/\delta\gamma)/\gamma^2$, then with probability $\geq 1 - \delta$, we have the symmetric difference $|\Delta(S_2, S)| \leq \gamma t/8$.*

Proof: We start by showing that the points in $\tilde{S}$ are voted by at least a $\gamma/2$ larger fraction of $\tilde{S}$ than the points in $S_1 \setminus \tilde{S}$. Let $i \in \tilde{S}$. Since S is (θ, α, β)-self-determined, at least αt points in S vote for i and since $|S \setminus \tilde{S}| \leq \gamma t/16$ we get that at least $(\alpha - \gamma/16)t$ points in $\tilde{S}$ vote for i. Since $|\tilde{S}| \leq t$, we obtain that at least a $\alpha - \gamma/16$ fraction of points in $\tilde{S}$ vote for i. Let j be a point in $S_1 \setminus S$. We know that at most βt points in $\tilde{S}$ vote for j and since $|\tilde{S}| \geq (1 - \gamma/16)t$, we have that at most a $\alpha - 3\gamma/4$ fraction of points in $\tilde{S}$ vote for j.

Fix $i \in S_1$. By Hoeffding's inequality, since U is a set of $8\log(32\theta M/\delta\gamma)/\gamma^2$ points drawn uniformly at random from $\tilde{S}$ we have that with probability at least $1 - \gamma\delta/(16\theta M)$ the fraction of points in $\tilde{S}$ that vote for i is within $\gamma/4$ of the fraction of points in U that vote for i. These together with the above observations imply that the expected size of $|\Delta(S_2, \tilde{S})|$ is $(\gamma\delta/(16\theta M))\theta M t = \gamma\delta t/16$. By Markov's inequality we obtain that there is at most a δ chance that $|\Delta(S_2, \tilde{S})| \geq \gamma t/16$. Using the fact $|\tilde{S} \setminus S| \leq \gamma t/16$ we finally get that with probability $1 - \delta$ we have $|\Delta(S_2, S)| \leq \gamma t/8$. ■

Algorithm 3 Purification procedure

Input: Preference system (V, Π), information I (parameters θ, α, β, γ, $k_2(\theta, \alpha, \gamma)$, $N_2(\theta, \alpha, \gamma)$, size t), list of rough approximations $\mathcal{L}_1$.

- For each element $S_1 \in \mathcal{L}_1$, repeat $N_2(\theta, \alpha, \gamma)$ times
- Sample a set U_2 of $k_2(\theta, \alpha, \gamma)$ points at random from S_1. Let $S_2 = \{i \in S_1 | v_{U_2}^{\theta t}(i) \geq (\alpha - \gamma/2)|U_2|\}$.
- Let $S_3 = \{i \in V | v_{S_2}^{\theta t}(i) \geq (\alpha - \gamma/2)|S_2|\}$. Add S_3 to the list $\mathcal{L}$.

Output: List of sets $\mathcal{L}$.

We now show how Lemmas 1 and 2 can be used to identify and enumerate communities.

Theorem 1 *Fix a (θ, α, β)-self-determined community S. Let $\gamma = \alpha - \beta$, $k_1(\theta, \alpha, \gamma) = \log(16/\gamma)/\alpha$, $k_2(\theta, \alpha, \gamma) = \frac{8}{\gamma^2}\log\left(\frac{32\theta k_1}{\gamma\delta}\right)$, $N_2(\theta, \alpha, \gamma) = O((\theta k_1)^{k_2}\log(1/\delta))$. Using Algorithm 2 together with Algo-*

rithm 3 for steps (1) and (2) of Algorithm 1, we have that with probability $\geq 1-\delta$ one of the elements in the list $\mathcal{L}$ we output is identical *to S.*

Proof: Since when running Algorithm 2 we search over all subsets of U of V of size $k_1(\theta,\alpha,\gamma)$, by Lemma 1 in one of the rounds we find a set U s.t. the set of points S_1 that are voted by at least an element in U cover a $1-\gamma/16$ fraction of S. So, $\mathcal{L}_1$ contains a rough approximation to S.

Since $|S|=t$, U_2 is a set of k_2 elements drawn at random from $\tilde{S}=S\cap S_1$ with probability $\geq (t/(2t\theta k_1))^{k_2}$. Therefore for $N_2=O((2\theta k_1)^{k_2}\log(1/\delta))$, with probability $\geq 1-\delta/2$ in one of the rounds the set U_2 is a set of k_2 elements drawn at random from $\tilde{S}$. In such a round, by Lemma 2, with probability $\geq 1-\delta/2$ we get a set S_2 such that $|\Delta(S_2,S)|\leq \gamma t/8$. A simple calculation shows that $S_3=S$. ■

Corollary 1 *The number of (θ,α,β)-self-determined communities in an affinity system (V,Π) satisfies $B(n)=n^{O(\log(1/\gamma)/\alpha)}\left(\frac{\theta\log(1/\gamma)}{\alpha}\right)^{O\left(\frac{1}{\gamma^2}\log\left(\frac{\theta\log(1/\gamma)}{\alpha\gamma}\right)\right)}$ and with probability $\geq 1-1/n$ we can find all of them in time $B(n)\mathrm{poly}(n)$.*

We note that Theorem 1 and Corollary 1 apply even if some nodes do not list all members of V in their preference lists, and then some nodes in a community S have fewer than $\theta|S|$ votes in total. If θ, α, and β are constant, then Corollary 1 shows that the number of communities is $O\left(n^{\log(1/\gamma)/\alpha}\right)$ which is *polynomial* in n and they can be found in polynomial time. We can show that the dependence on $n^{1/\alpha}$ is necessary:

Theorem 2 *For any constant $\theta\geq 1$, for any $\alpha\geq 2\sqrt{\theta}/n^{1/4}$, there exists an instance such that the number of (θ,α,β)-self-determined communities with $\alpha-\beta=\gamma=\alpha/2$ is $n^{\Omega(1/\alpha)}$.*

PROOF SKETCH: Consider $L=\sqrt{n}$ blobs B_1, ..., B_L each of size $\sqrt{n}$. Assume that each point ranks the points inside its blob first (in an arbitrary order) and it then ranks the points outside its blob randomly. One can show that with non-zero probability for $l\leq n^{1/4}/(2\sqrt{\theta})$ any union of l blobs satisfies the (θ,α,β)-self-stability property with parameters $\alpha=1/l$ and $\gamma=\alpha/2$. Full details appear in Appendix A.1. ■

3.1 Self-determined Communities in Weighted Affinity Systems

We provide here a simple efficient reduction from the weighted case to the non-weighted case.

Theorem 3 *Given a weighted affinity system (V,A), θ, α, β, $\epsilon<\alpha$, and a community size t, there is an efficient procedure that constructs a non-weighted instance (V',Π) along with a mapping f from V' to V, s.t. for any (θ,α,β) community S in V there exists a $(\theta,\alpha-\epsilon,\beta)$ community S' in (V',Π) with $f(S')=S$.*

Proof: Given the original weighted instance (V,A), we construct a non-weighted instance (V',Π) as follows. For each $s\in V$, we create a blob B_s of k nodes in V'. For any $s,\tilde{s}\in V$, if p is the weight $a^{\theta t}_{s,\tilde{s}}$ with which s votes for $\tilde{s}$, we connect B_s to $B_{\tilde{s}}$ with $G_{k,k,\lfloor pk\rfloor}$, where $G_{k,k,\lfloor pk\rfloor}$ is a bipartite graph with k nodes on the left and k nodes on the right such that each edge on the left has out-degree $\lfloor pk\rfloor$ and each node on the right has in-degree $\lfloor pk\rfloor$. Clearly all nodes in V' rank at most $k|S|\theta$ other nodes (and do not have an opinion about the rest). Let $k=1/\epsilon$. Consider a community S in (V,A). For any $s\in S$ and for each node in $i\in B_s$ the total vote from nodes in $B_{\tilde{s}}$ for $\tilde{s}\in S$ (when evaluating whether $\cup_{\tilde{s}\in S}B_{\tilde{s}}$ is a good community or not)

is at least $\alpha|S|k - |S| \geq k|S|(\alpha - \epsilon)$. Moreover, for $s \notin S$ and for each node in B_s we have the total vote from the nodes in $B_{\tilde{s}}$ for $\tilde{s} \in S$ is at most $\beta|S|k$. Therefore $\cup_{\tilde{s}\in S} B_{\tilde{s}}$ is a legal $(\theta, \alpha - \epsilon, \beta)$-self-determined community of size kt in the non-weighted instance (V', A). ■

Using this reduction we immediately get the following result:

Theorem 4 *For any θ, α, β, $\gamma = \alpha - \beta$, the number of weighted (θ, α, β)-self-determined communities is $B(n) = (n/\gamma)^{O(\log(1/\gamma)/\alpha)} \left(\frac{2\theta \log(1/\gamma)}{\alpha}\right)^{O\left(\frac{1}{\gamma^2}\log\left(\frac{\theta\log(1/\gamma)}{\alpha\gamma}\right)\right)}$ and we can find them in time $B(n)\mathrm{poly}(n)$.*

Proof: We perform the reduction in Theorem 3 with $\epsilon = \gamma/2$ and use the algorithm in Theorem 1 and the bound in Corollary 1. The proof follows from the fact that the number of vertices in the new instance has increased by only a factor of $2/\gamma$. We also note that each set output on the reduced instance can then be examined on the original weighted affinity system, and kept iff it satisfies the community definition with original parameters. ■

3.2 Self-determined Communities in Multi-faceted Affinity Systems

A multi-faceted affinity system is a system where each node may have more than one rankings of other nodes. Suppose that each element i is allowed to have at most f different rankings $\pi_i^1, \ldots, \pi_i^f$. We say that the pair (S, ψ) is a multi-faceted community where $\psi : S \to \{1, \ldots, f\}$, if S is a community where $\psi(i)$ specifies which ranking facet should be used by element i. In other words, as before, let $\phi^{\theta}_{S,\psi}(i) := |\{s \in S | i \in \pi_s^{\psi(s)}(1 : \lceil\theta|S|\rceil)\}|$. Then (S, ψ) is a (α, β, θ)-multifaceted community if for all $i \in S$, $\phi^{\theta}_{S,\psi}(i) \geq \alpha|S|$, and for all $j \notin S$, $\phi^{\theta}_{S,\psi}(j) < \beta|S|$.

We show that for a bounded f, even though there may be exponentially many functions ψ, it is not harder to find multifaceted communities than to find regular communities. Note that all our sampling algorithms can be adapted as follows. Once a representative sample $\{i_1, \ldots, i_k\}$ of the community S is obtained, we can guess the facets $\psi(i_1), \ldots, \psi(i_k)$ while adding a multiplicative f^k factor to the running time. We can thus get the set S_2 approximating S in the same way as it is found in Algorithms 2 and 3 while adding a multiplicative factor of $f^{k_1+k_2}$ to the running time. We thus obtain a list $\mathcal{L}$ that for each multi-faceted community (S, ψ) contains a set S_2 such that $\Delta(S_2, S) < \gamma t/8$. Given S_2 we can output S with probability $> f^{-8\log n/\gamma^2}/2$: guess a set U_2 of $m = 8\log n/\gamma^2$ points in S_2; guess a function ψ_2 on U_2; output $S =$ the set of points that receive at least $(\alpha - \gamma/2)t$ votes according to (U_2, ψ_2). Moreover, a facet structure ψ' can be recovered on S so that (S, ψ') is an $(\alpha - \gamma/4, \beta + \gamma/4, \theta)$-multifaceted community using a combination of linear programming and sampling. Details appear in Appendix A.3.

Theorem 5 *Let S be an f-faceted (α, β, θ)-community. Then there is an algorithm that runs in $O(n^2)$ time and outputs S, as well as a facet structure ψ' on S such that (S, ψ') is an $(\alpha - \gamma/4, \beta + \gamma/4, \theta)$-multifaceted community with probability $\geq (f \cdot n)^{-O(\log(1/\gamma)/\alpha)} \left(\frac{f\cdot\theta\log(1/\gamma)}{\alpha}\right)^{-O\left(\frac{1}{\gamma^2}\log\left(\frac{\theta\log(1/\gamma)}{\alpha\gamma}\right)\right)} f^{-O(\log n/\gamma^2)}$.*

4 A Local Algorithm for Finding Self-determined Communities

In this section we describe a local algorithm for finding a community. Given a single element v and the target community size t, the goal of the algorithm is to output a community S of size t containing v. Let us

fix a target community S that we are trying to uncover this way.

We note that we need $\alpha > 1/2$ for a local algorithm that uses only one seed to succeed. If $\alpha \leq 1/2$ then one may have a valid (θ, α, β)-community that is comprised of two disjoint cliques of vertices. In this case, no local algorithm that starts with just one vertex as a seed may uncover *both* cliques, however we can extend the construction below if we start with $O(1/\alpha)$ seeds. Below, we focus on providing a local algorithm for $\alpha > 1/2$. Our local algorithm will follow the structure of the generic Algorithm 1. The main technical challenge is to provide a local procedure for producing rough approximations. In general, it is not possible to do so starting from *any* seed vertex $v \in S$. For example, if v is a super-popular vertex that is voted first by *everyone* in V, then v will belong to all communities including S, but v would contain no "special information" that would allow one to identify S. However, we will show that *a constant fraction* of the nodes in S are sufficiently "representative" of S to enable one to recover S.

Let us fix t and θ. For an element v, we let $R(v)$ be a uniformly random element which receives v's vote with these parameters. In other words, $R(v) :=$ uniform element of $\pi_v(1 : \theta \cdot t)$. We start with the main technical claim that enables a local procedure for producing rough approximations.

Lemma 3 *Let S be any (θ, α, β)-community of size t. Let $\eta := 2\alpha - 1 > 0$. Then there is a subset $T \subseteq S$ such that $|T| \geq \eta t$ and for each pair $v \in T$ and $u \in S$, we have $\mathbf{Pr}[R(R(v)) = u] \geq \frac{(\alpha - 1/2)/\theta^2}{t}$.*

Proof: For each element $v \in S$ denote by $O_S(v) := \pi_v(1 : \theta \cdot t) \cap S$ – the elements of S that v votes for, and by $I_S(v) := \{u \in S : \ v \in \pi_u(1 : \theta \cdot t)\}$ – the elements of S that vote for v. By the community property we know that $|I_S(v)| \geq \alpha t$ for all $v \in S$. Observe that

$$\sum_{v \in S} |O_S(v)| = \sum_{v \in S} |I_S(v)| \geq \alpha t^2.$$

Hence at least an η-fraction of v's in S must satisfy $|O_S(v)| \geq t/2$, where $\eta = 2\alpha - 1$. Let $T := \{v : \ |O_S(v)| \geq t/2\} \subseteq S$. For any $v \in T$ and any $u \in S$, we have

$$|O_S(v) \cap I_S(u)| \geq |O_S(v)| + |I_S(u)| - t \geq (\alpha - 1/2) \cdot t.$$

To finish the proof note that

$$\mathbf{Pr}[R(R(v)) = u] \geq \mathbf{Pr}[R(v) \in O_S(v) \cap I_S(u)] \cdot \frac{1}{\theta \cdot t} \geq \frac{(\alpha - 1/2) \cdot t}{\theta \cdot t} \cdot \frac{1}{\theta \cdot t} = \frac{(\alpha - 1/2)/\theta^2}{t}.$$

■

We call any vertex v in the set T in Lemma 3 a "good seed vertex" for S. Lemma 3 suggests a natural procedure (Algorithm 4) for generating a rough approximation in a local way given a good seed vertex.

Algorithm 4 Generate rough approximations

Input: Preference system (V, Π), information I (parameters θ, α, β, γ, vertex v, size t).

- Set $S_1 = \left\{u : \ \mathbf{Pr}[u = R(R(v))] \geq \frac{(\alpha - 1/2)/\theta^2}{t}\right\}$.

Output: List of sets $\mathcal{L} = \{S_1\}$.

Theorem 6 *Assume $\alpha > 1/2$. Let $k_2(\theta,\alpha,\gamma) = O\left(\frac{\log(\theta/\delta\gamma(\alpha-1/2))}{\gamma^2}\right)$, $N_2(\theta,\alpha,\gamma) = \left(\frac{\theta^2}{\alpha-1/2}\right)^{k_2(\theta,\alpha,\gamma)} \log(1/\delta)$. Assuming v is a good seed element for a community S, then by using Algorithm 4 together with Algorithm 3 for steps (1) and (2) of Algorithm 1, we have that with probability $\geq 1-\delta$ we will output S.*

Proof: It is enough to show that each iteration of the purification algorithm (Algorithm 3) has a probability $\geq \left(\frac{\alpha-1/2}{\theta^2}\right)^{k_2}$ to output S. Since v is a good seed element of S, the set S_1 produced by Algorithm 4 must contain S. It is easy to see that $|S_1| \leq t\theta^2/(\alpha-1/2)$. Thus, applying Lemma 2 with $M = \theta/(\alpha-1/2)$ we see that if the points of U_2 are drawn uniformly from S, then with high probability S_2 is $\gamma/8$-close to S, and $S_3 = S$. Since conditioned on $U_2 \subseteq S$, U_2 is uniform in S, our probability of success is given by the probability that $U_2 \subseteq S$, which is equal to $\left(\frac{|S|}{|S_1|}\right)^{k_2} \geq \left(\frac{\alpha-1/2}{\theta^2}\right)^{k_2}$, which completes the proof. ■

Note that when $\alpha > 1/2$, β, and θ are constants, the purification procedure will run in a constant number of iterations. Our main result of this section is the following:

Theorem 7 *Suppose $\alpha > 1/2$. Assume α, β, θ, and δ are constants. If v is chosen uniformly at random from S, then with probability at least $(2\alpha-1)(1-\delta)$ we can find S in time $O(t\log t)$.*

Proof: First, by Lemma 3, with probability at least $2\alpha-1$, element v is such that for all $u \in S$, we have $\mathbf{Pr}[R(R(v)) = u] \geq \frac{\alpha-1/2}{\theta^2 t}$. We now implement Algorithm 4 by performing $\left(\frac{8\theta^2 t}{\alpha-1/2}\right)\log(2t/\delta)$ random draws from $R(R(v))$ and letting S_1 be the set of points u hit at least $4\log(2t/\delta)$ times. By Chernoff bounds, for each $u \in S$, we have included u in S_1 with probability at least $1 - e^{-8\log(2t/\delta)/8} = 1 - \delta/(2t)$, so with probability at least $1-\delta/2$ we have $S_1 \supseteq S$. Furthermore, since we only include points hit at least $4\log(2t/\delta)$ times, we have $|S_1| \leq \left(\frac{2\theta^2 t}{\alpha-1/2}\right)$. Thus, the analysis in Theorem 3 implies that the purification step (Algorithm 3) will succeed with probability at least $1-\delta/2$ for a choice of $N_2 = \left(\frac{2\theta^2}{\alpha-1/2}\right)^{k_2(\theta,\alpha,\gamma)} \log(2/\delta)$. Putting these together yields the desired success probability. Furthermore, since α, β, θ, δ are constants, the overall time is $O(t\log t)$. ■

It is not hard to see that the algorithm in Theorem 6 will work even if t is given to it only up to some small multiplicative error. As a corollary of Theorem 6, we see that the number of communities is actually linear and we can find all of them in quasilinear time.

Theorem 8 *Suppose that $\alpha > 1/2$. The total number of (θ,α,β)-self-determined communities is bounded by $O\left(n \cdot \frac{1}{\min(\gamma,1/2-\alpha)} \cdot \left(\frac{\theta^2}{\alpha-1/2}\right)^{O\left(\frac{\log(\theta/\delta\gamma(\alpha-1/2))}{\gamma^2}\right)}\right)$, which is $O(n)$ if α, β, and θ are constants.*

Proof: It is easy to see that executing the Algorithm in Theorem 7 where we only do one iteration of the purification step (i.e., of Algorithm 3) with inputs $t' \in ((1-\varepsilon)t,(1+\varepsilon)t)$, $\alpha' = \alpha - 4\varepsilon$, $\beta' = \beta + 4\varepsilon$, $\theta' = \theta(1+\varepsilon)$, and an appropriate seed vertex $v \in S$ will lead to a discovery of an (θ,α,β)-community of size $|S| = t$ with probability $\geq p := \left(\frac{\alpha-1/2}{\theta^2}\right)^{k_2(\theta,\alpha,\gamma)}$, as long as ε is sufficiently small. Here it is enough to take $\varepsilon = \min(\gamma,\alpha-1/2)/100$. Thus a pair (v,t'), where v is a vertex and t' is the target size corresponds to at most $1/p$ distinct communities. Moreover, each community S of size t corresponds to more than $t(2\alpha-1)/2$ such pairs. Since t' needs only to be within a multiplicative $(1+\varepsilon)$ from t, we can always select t' from the set of values $\{(1+\varepsilon)^i : i = 0,1,\ldots,\lceil\log_{1+\varepsilon} n\rceil\}$. For each value t', the number of

communities of size between t' and $t'(1+\varepsilon)$ is thus bounded by the number of possible pairs (t', v) $(= n)$, times $1/p$ and divided by $t'(2\alpha - 1)/2$:

$$\#\{\text{communities of size between } t' \text{ and } t'(1+\varepsilon)\} \leq \frac{n}{t'} \cdot \frac{1/p}{(2\alpha-1)/2}.$$

Summing over the possible values of t' we obtain the upper bound:

$$n \cdot \frac{2}{\varepsilon(2\alpha-1)} \cdot \left(\frac{\theta^2}{\alpha - 1/2}\right)^{k_2(\theta,\alpha,\gamma)},$$

which leads to the bound in the statement of the theorem. ■

Note: We can extend our local approach to weighted and multi-faceted affinity systems. See Appendix A.4.

4.1 An Alternative Non-local Algorithm

The analysis in this section suggests an alternative way for generating rough approximations in the non-local model which leads to an algorithm that provides asymptotically better bounds than Theorem 1 in interesting cases, in particular when θ, α, and γ are constants and there is a large gap between α and γ. This leads to an improved polynomial bound of $n^{O(\log(1/\alpha)/\alpha)}$ on the number of (θ, α, β)-self-determined communities when θ, α, and γ are constants using Algorithm 5:

Algorithm 5 Generate rough approximations

Input: Preference system (V, Π), information I (parameters θ, α, β, size t).

- Set $\mathcal{L} = \emptyset$; $\gamma = \beta - \alpha$.
- Exhaustively search over all subsets U_0 of V of size $\lceil(\log 1/\alpha)/\alpha\rceil + 1$; for each U_0 to the $\mathcal{L}$ the set $S_1 := \left\{x : \sum_{y \in U_0} \mathbf{Pr}[x = R(R(y))] \geq \frac{\alpha}{2\theta^2 t}\right\}$.

Output: List of sets $\mathcal{L}$.

Theorem 9 *Fix a (θ, α, β)-self-determined community S. Let $\gamma = \alpha - \beta$, $k_1(\theta, \alpha, \gamma) = O(\log(1/\alpha)/\alpha)$, $k_2(\theta, \alpha, \gamma) = O\left(\frac{1}{\gamma^2}\log\left(\frac{\theta k_1}{\gamma\delta}\right)\right)$, $N_2(\theta, \alpha, \gamma) = O((\theta^2/\alpha^3)^{k_2}\log(1/\delta))$. Using Algorithm 5 together with Algorithm 3 for steps (1) and (2) of Algorithm 1, then with probability $\geq 1 - \delta$ one of the elements in the list $\mathcal{L}$ we output is* identical *to S.*

PROOF SKETCH: By using a reasoning similar to the one in Lemma 2 we can show that there exist a set U_0 of $\lceil(\log 1/\alpha)/\alpha\rceil + 1$ points such that the subset U_1 of points voted by at least a member in U_0 contains $\geq 1 - \alpha/2$ fraction of S. We show in the following that the corresponding set S_1 indeed covers S. Fix a vertex $x \in S$. We need to show that

$$\sum_{y \in U_0} \mathbf{Pr}[x = R(R(y))] \geq \frac{\alpha}{2\theta^2 t}.$$

Let $Q \subseteq S$ be the set of elements that vote for x. We know that $|Q| \geq \alpha t$, since $x \in S$. Thus

$$|U_1 \cap Q| \geq |U_1| + |Q| - |S| > \alpha t/2.$$

Each $z \in U_1 \cap Q$ contributes at least $1/\theta^2 t^2$ to the sum $\sum_{y \in U_0} \mathbf{Pr}[x = R(R(y))]$. Thus this sum is at least $(\alpha t/2) \cdot (1/\theta^2 t^2) = \alpha/(2\theta^2 t)$. Hence $x \in S_1$, as required. Moreover, by observing that

$$\sum_x \sum_{y \in U_0} \mathbf{Pr}[x = R(R(y))] = \sum_{y \in U_0} \sum_x \mathbf{Pr}[x = R(R(y))] < 1/\alpha^2,$$

we obtain $|S_1| < \frac{2\theta^2 t}{\alpha^3}$.

Since when running Algorithm 5 we exhaustively search over all subsets of U_1 of V of size $k_1(\theta, \alpha, \gamma)$, in one of the rounds we find a set U_1 s.t. $|S_1| < \frac{2\theta^2 t}{\alpha^3}$, $S \subseteq S_1$. So, $\mathcal{L}_1$ contains a rough approximation to S. Finally, using a reasoning similar to the one in Theorem 1 we get the desired conclusion. ■

Theorem 9 gives asymptotically better bounds than Theorem 1 when $N_1 = n^{k_1(\theta,\alpha,\gamma)}$ is the dominant term in the bound (e.g., when θ, α, and γ are constants) and especially when there is a large gap between α and γ – since k_1 is reduced from $\log(16/\gamma)/\alpha$ to $\lceil \log(1/\alpha)/\alpha \rceil + 1$. On the other hand, Theorem 9 has worse dependence on θ and α in N_2, so for certain parameter settings, Theorem 1 can be preferable especially if one optimizes the constants in Lemmas 1 and 2 based on the given parameters.

5 Self-determined Communities in Social Networks

In this section we present a natural notion of self-determined communities in social networks and discuss how our analysis sheds light on the notion of (α, β)-clusters [16, 17, 12]. We assume that the input is a directed graph $G = (V, E)$ and for a vertex i we denote by d_i its out-degree. As discussed in Section 1, given a social network we can consider the affinity system induced by direct lifting and then consider self-determined communities in that affinity system. This leads to the following very natural notion:

Definition 3 *Let $G = (V, E)$ be a directed graph and let $\theta, \alpha, \beta \geq 0$ with $\beta < \alpha \leq 1$. Consider the affinity system $(V, a_1, \ldots, a_n)$ where $a_{i,j} = w_{i,j}$ if $(i, j) \in E$ and $a_{i,j} = 0$ otherwise. A subset $S \subseteq V$ is a (θ, α, β) self-determined community in G if it is a (θ, α, β) weighted self-determined community in $(V, a_1, \ldots, a_n)$.*

Note that when evaluating a community of size t each node i is allowed a total vote of at most θt. One natural way to achieve this is to only fractionally count edges from high-degree nodes i, giving them weight $\min(\theta t/d_i, 1)$ when evaluating a community of size t in the induced weighted affinity system.

The community notion introduced in [16, 17] is as follows:

Definition 4 *Let α, β with $\beta < \alpha \leq 1$ be two positive parameters. Given an undirected graph, $G = (V, E)$, where every vertex has a self-loop, a subset $S \subseteq V$ is an (α, β)-cluster if S is:*

(1) Internally Dense: $\forall i \in S$, $|E(i, S)| \geq \alpha |S|$.

(2) Externally Sparse: $\forall i \notin S$, $|E(i, S)| \leq \beta |S|$.

The (α, β)-cluster notion resembles our community notion in Definition 3. In particular, in the case where the graph is undirected, Definition 3 is similar to Definition 4, except that in the case of our Definition 3 each node i is allowed a total vote of at most θt. As discussed above one way to achieve this is to only fractionally count edges from high-degree nodes i, giving them weight $\min(\theta |S|/d_i, 1)$. This distinction is

crucial for getting polynomial time algorithms. From our results in the previous sections we have that every graph has only a polynomial number of communities satisfying Definition 3 and moreover, we can find all of them in polynomial time. In contrast, as we show, there exist graphs with a superpolynomial number of (α, β)-clusters.

Theorem 10 *For any constant ϵ, $\alpha = 1$, $\alpha - \beta = 1/2 - \epsilon$, there exist instances with $n^{\Omega(\log n)}$ (α, β)-clusters.*

Proof: Consider the graph $G_{n,p}$ with $p = 1/2^l$. Consider all $\binom{n}{k}$ sets of size $k = \frac{2\log n}{l}(1-\delta)$, where δ is a constant (determined later). For each such set S, the probability it is a clique is

$$p^{\binom{k}{2}} \geq (1/2)^{\ell k^2/2} = (1/2)^{2\log^2 n(1-\delta)^2/\ell} = n^{-k(1-\delta)}.$$

We now want to show that conditioned on S being a clique, it is also an (α, β)-cluster with probability at least $1/2$. This will imply that the *expected* number of (α, β)-clusters is at least

$$0.5\binom{n}{k} n^{-k(1-\delta)} = n^{\Omega(\log n)}.$$

Fix such set of size $k = \frac{2\log n}{l}(1-\delta)$. The probability that a node outside is connected to more than a $(1/2+\epsilon)$-fraction of the set is upper bounded by

$$2^k \left(\frac{1}{2^l}\right)^{\frac{k}{2}(1+\epsilon)} \leq n^{\frac{2}{l}} 2^{-\frac{lk}{2}(1+\epsilon)} = n^{\frac{2}{l}} n^{-(1+\epsilon)(1-\delta)}.$$

By imposing $\frac{2}{l} - (1+\epsilon)(1-\delta) < -1 + \log_n(2)$, we get that this probability is upper bounded by $1/(2n)$. So by union bound over all nodes we then get the desired result. We need to impose $(1+\epsilon)(1-\delta) - \frac{2}{l} > 1 + \log_n(2)$. This is true for $\delta \leq \epsilon/4$ and $l > 12/\epsilon$ and n large enough. ■

We note that for certain range of parameters our bounds in Theorem 13 this improves over the general upper bound given in [16, 17]. Moreover, we show that even in graphs with only one (α, β)-cluster, we show that finding this cluster is at least as hard solving the *planted clique problem* for planted cliques of size $O(\log n)$, which is believed to be hard (see, e.g., Hazan and Krauthgamer [11]).

The Hidden Clique Problem: In this problem, the input is a graph on n vertices drawn at random from the following distribution $G_{n,1/2,k}$ pick a random graph from $G_{n,1/2}$ and plant in it a clique of size $k = k(n)$. The goal is to recover the planted clique (in polynomial time), with probability at least (say) $1/2$ over the input distribution. The clique is hidden in the sense that its location is adversarial and not known to the algorithm. The hidden clique problem becomes only easier as k gets larger, and the best polynomial-time algorithm to date [1], solves the problem whenever $k = \Omega(\sqrt{n})$. Finding a hidden clique for $k = c\log n$ for any c is believed to be hard. The decision version of this problem is also believed to be hard.

We begin with a simpler result that finding the *approximately-largest* (α, β)-cluster is at least as hard as the hidden clique problem.

Theorem 11 *Suppose that for $\alpha = 1$ and $\beta - \alpha = 1/4$, there was an algorithm that for some constant c could find an (α, β)-cluster of size at least MAX/c, where MAX is size of the largest community with those parameters. Then, that algorithm could be used to distinguish (1) a random graph $G_{n,1/2}$ from (2) a random graph $G_{n,1/2}$ in which a clique of size $2c\log_2(n)$ has been planted.*

Proof: We can show that with probability at least $1 - 1/n$ the largest clique in $G_{n,1/2}$ largest clique has size $2\log(n)$, which implies the largest (α, β) cluster (with $\alpha = 1$ and $\beta - \alpha = 1/4$) has size at *most* $2\log(n)$. On the other hand we can also show that with probability at least $1 - 1/n$, for $c \geq 8\ln 2$ the planted clique of size $2c\log_2(n)$ *is* a cluster with these parameters. Thus, under the assumption that distinguishing these two cases is hard, the problem of finding the approximately-largest (α, β)-cluster is hard. ■

We now show that in fact, even finding a single (α, β)-cluster is as hard as the hidden clique problem. Here, instead of $G_{n,1/2}$ we will use $G_{n,p}$ for constant $p > 1/2$. Note that the hidden clique problem remains hard in this setting as well.[3]

Theorem 12 *For sufficiently small (constant) γ and ϵ, with probability at least $1 - 3/n$, we have that: (1) the graph $G_{n,1-\gamma-\epsilon}$ has no $(1, 1-\gamma)$ clusters; and (2) a hidden clique of size $\frac{1}{\epsilon^2}\log n$ is an $(1, 1-\gamma)$ cluster. Therefore, finding even one such cluster is as hard as the hidden clique problem.*

Proof: Consider $G_{n,p}$ for $p = 1 - \gamma - \epsilon$. We start by showing that with probability at least $1 - 1/n$ the size of the largest clique is at most $\frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$. For any k, the probability that there exists a clique of size k is at most

$$\binom{n}{k} p^{\binom{k}{2}} \leq \frac{n^k}{k!} p^{k^2/2} p^{-k/2}.$$

For $k = \frac{-2\ln n}{\ln(1-\gamma-\epsilon)} = -2\log_p n$, this is

$$\frac{p^{-k/2}}{k!} n^{-2\log_p n} p^{2(\log_p n)^2} = \frac{p^{-k/2}}{k!} = \frac{n}{k!} = o(\frac{1}{n}).$$

This immediately implies that with probability at least $1-1/n$, $G_{n,p}$ does not contains any $(1, 1-\gamma)$ clusters of size greater than $\frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$.

We now show that with probability at least $1 - 1/n$, $G_{n,p}$ does not contain any $(1, 1-\gamma)$ clusters of size $\leq \frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$. For this, we will show that for any set S of size $\leq \frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$ and any node v not in S, the probability that v connects to at least $(1-\gamma)|S|$ nodes inside S is at least $1/\sqrt{n}$. Because these events are independent over the different nodes v, this implies that the probability that no node v outside S connects to at least $(1-\gamma)|S|$ nodes inside S is at most $\left(1 - \frac{1}{\sqrt{n}}\right)^{n-k} \leq e^{-\sqrt{n}/2}$. By union bound over all sets S of size at most $\frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$, this will imply that the probability there exits a $(1, 1-\gamma)$ cluster of size at most $\frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$ is at most $1/n$.

Consider a set S of size k and a node v outside S. The probability that v connects to more than $(1-\gamma)k$ nodes inside S is at least

$$\binom{k}{\gamma k}(1-\gamma-\epsilon)^{(1-\gamma)k}(\gamma+\epsilon)^{\gamma k} \geq \frac{1}{k}\left(\frac{(1-\gamma)ke}{\gamma k}\right)^{\gamma k}(1-\gamma-\epsilon)^{(1-\gamma)k}(\gamma+\epsilon)^{\gamma k}.$$

This follows from the fact that

$$\binom{k}{\gamma k} = \frac{k(k-1)\ldots(k-\gamma k+1)}{(\gamma k)!} \geq \frac{((1-\gamma)k)^{\gamma k}}{k(\gamma k/e)^{\gamma k}} = \frac{1}{k}\left(\frac{(1-\gamma)ke}{\gamma k}\right)^{\gamma k},$$

[3] In particular, if it were easy, then one could solve the decision version of the hidden clique problem for $G_{n,1/2}$ by first adding additional random edges and then solving the problem for $G_{n,p}$. We assume here that the planted clique has size greater than the largest clique that would be found in $G(n,p)$.

where we use the fact that $(\gamma k)! < 2\sqrt{2\pi\gamma k}(\gamma k/e)^{\gamma k} < k(\gamma k/e)^{\gamma k}$.

So, the probability that v connects to more than $(1-\gamma)k$ nodes inside S is at least

$$\frac{1}{k}(1-\gamma-\epsilon)^k \left[\frac{1-\gamma}{\gamma}\cdot\frac{\gamma+\epsilon}{1-\gamma-\epsilon}e\right]^{\gamma k} \geq [(1-\gamma-\epsilon)e^{\gamma}]^k \frac{1}{k}.$$

This is decreasing with k and thus it suffices to consider $k = \frac{-2\ln n}{\ln(1-\gamma-\epsilon)}$. For this k, we get that the probability that v connects to more than $(1-\gamma)k$ nodes inside S is at least

$$\frac{1}{k}e^{-2\ln n}e^{-\frac{2\gamma\ln n}{\ln(1-\gamma-\epsilon)}} = \frac{1}{k}n^{-2-\frac{2\gamma}{\ln(1-\gamma-\epsilon)}}.$$

We want this to be greater than $1/\sqrt{n}$, and thus it suffices to have $-2-\frac{2\gamma}{\ln(1-\gamma)} > -0.4$. This holds for $\gamma = 0.1, \epsilon = 0.01$.

Finally, it is easy to show that with probability at least $1-1/n$, a hidden clique of size $k = \frac{1}{\epsilon^2}\ln n$ is a $(1, 1-\gamma)$ cluster. This follows by noticing that every vertex outside the clique has in expectation $k(1-\gamma-\epsilon)$ connections insides the clique, so by Hoeffding bounds, the probability it has more than $k(1-\gamma-\epsilon)+\epsilon k = k(1-\gamma)$ neighbors inside the clique is at most $1/n^2$. By union bound, we get that with probability at least $1-1/n$ every vertex outside the clique has at most $k(1-\gamma)$ neighbors inside the clique so the planted clique is a community as desired. ■

References

[1] N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. *Random Structures Algorithms*, 1998.

[2] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *47th Annual Symposium on Foundations of Computer Science*, 2006.

[3] A. Arora, T. Ge, S. Sachdeva, and G. Schoenebeck. Finding overlapping communities in social networks: Toward a rigorous approach. Manuscript, 2011.

[4] M.-F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the 40th ACM Symposium on Theory of Computing*, 2008.

[5] L. M. Busse, P. Orbanz, and J. M. Buhmann. Cluster analysis of heterogeneous rank data. In *Proceedings of the 24th Annual International Conference on Machine Learning*, 2007.

[6] M. Charikar, S. Guha, E. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, 1999.

[7] W. Chen, Z. Liu, X. Sun, and Y. Wang. A game-theoretic framework to identify overlapping communities in social networks. *Data Mining and Knowledge Discovery Journal, special issue on ECML PKDD*, 2010.

[8] D. Cheng, R. Kannan, S. Vempala, and G. Wang. A divide-and-merge methodology for clustering. In *Proceedings of the ACM Symposium on Principles of Database Systems*, 2005.

[9] P. Doyle and J. Snell. *Random walks and electric networks*. Mathematical Assoc. of America, 1984.

[10] R. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley, 2001.

[11] E. Hazan and R. Krauthgamer. How hard is it to approximate the best nash equilibrium. *SIAM Journal on Computing*, 2011.

[12] J. He, J. E. Hopcroft, H. Liang, S. Suwajanakorn, and L. Wang. Detecting the structure of social networks using (α, β)-communities. In *8th International Conference on Algorithms and Models for the Web-graph*, 2011.

[13] K. Jain, M. Mahdian, and A. Saberi. A new greedy approach for facility location problems. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, 2002.

[14] J. Leskovec, K. Lang, A. Dasgupta, and M. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 2009.

[15] J. Leskovec, K. Lang, and M. Mahoney. Empirical comparison of algorithms for network community detection. In *ACM WWW International conference on World Wide Web*, 2010.

[16] N. Mishra, R. Schreiber, I. Stanton, and R. Tarjan. Clustering social networks. In *5th International Conference on Algorithms and Models for the Web-graph*, 2007.

[17] N. Mishra, R. Schreiber, I. Stanton, and R. Tarjan. Finding strongly-knit clusters in social networks. *Internet Mathematics*, 2009.

[18] G. Narasimhan and M. Smid. *Geometric Spanning Networks*. Cambridge University Press, 2007.

[19] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 2006.

[20] D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM Symposium on Theory of Computing*, 2004.

[21] D. A. Spielman and S.-H. Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. *CoRR*, abs/0809.3232, 2008. Available at `http://arxiv.org/abs/0809.3232`. Submitted to SICOMP.

[22] K. Voevodski, S. Teng, and Y. Xia. Finding local communities in protein networks. *BMC Bioinformatics*, 2009.

A Additional Proofs

A.1 Finding Self-determined Communities in Quasi-Polynomial Time

We present here a simple quasi-polynomial algorithm for enumerating all the self-determined communities.

Theorem 13 *For any θ, α, β, $\gamma = \alpha - \beta$, there are $n^{O(\log n/\gamma^2)}$ sets which are (θ, α, β) (weighted) self-determined communities. All such communities can be found by using Algorithm 6 with parameters θ, α, β, $\gamma = \alpha - \beta$ and $k(\gamma) = 2\log(4n)/\gamma^2$.*

Proof: Fix a (θ, α, β) (weighted) self-determined community S. We show that there exists a multiset U of size $k(\gamma) = 2\log(4n)/\gamma^2$ such that the set S_U of points in V that receive at least $(\alpha - \gamma/2)|U|$ amount of vote from points in U is identical to S. The proof follows simply by the probabilistic method. Let us fix a point $i \in V$. By Hoeffding, if we draw a set U of $2\log(4n)/\gamma^2$ uniformly at random from S, then with probability $1 - 1/(2n)$, the average amount of vote that i receives from points in U is within $\gamma/2$ of the average amount of vote that i receives from points in S. By union bound, we get that with probability at least $1/2$, for all points in V the average amount of vote that they receive from points in U is within $\gamma/2$ of the average amount of vote that they receive from points in S. Using this together with the definition of a self-determined community, we get that with probability $1/2$ we obtain $S_U = S$ for U of size $2\log(4n)/\gamma^2$ drawn uniformly at random from S. This then implies that there must exist a multiset U of size $k(\gamma)$ such that $S_U = S$.

Since in Algorithm 6 we exhaustively search over all multisets U (of point from V) of size $k(\gamma)$, we clearly get the list L we output contains all the (θ, α, β) (weighted) self-determined communities. Moreover, clearly, $n^{O(\log n/\gamma^2)}$ is an upper bound on the number of (θ, α, β) (weighted) self-determined communities. ■

Algorithm 6 Algorithm for enumerating self-determined communities

Input: Affinity system (V, Π), parameters θ, α, β, γ; $k(\gamma)$;

- Set $L = \emptyset$.
- Exhaustively search over all multisets U with elements from V of size $k(\gamma)$.
  - For $t = 1$ to n (determining the meaning of "vote for") do:
    - Let S_U be the subset of points in V that receive at least $(\alpha - \gamma/2)|U|$ amount of vote from points in U. Add S_U to the list $\mathcal{L}$.
- Remove from the list $\mathcal{L}$ all the sets that are not (θ, α, β) weighted self-determined communities.

Output: List of self-determined communities L.

A.2 Additional Proofs in Section 3

THEOREM 2 For any constant $\theta \geq 1$, for any $\alpha \geq 2\sqrt{\theta}/n^{1/4}$, there exists an instance such that the number of (θ, α, β)-self-determined communities with $\beta - \alpha = \gamma = \alpha/2$ is $n^{\Omega(1/\alpha)}$.

Proof: Consider $L = \sqrt{n}$ blobs B_1, ..., B_L each of size $\sqrt{n}$. Assume that each point ranks the points inside its blob first (in an arbitrary order) and it then ranks the points outside its blob randomly. We claim that with non-zero probability for $l \leq n^{1/4}/(2\sqrt{\theta})$ any union of l blobs satisfies the (θ, α, β)-self-stability property with parameters $\alpha = 1/l$ and $\gamma = \alpha/2$.

Let us fix a set S which is a union of l blobs. Note that for each point i in S, the expected number of points

in S voting for i is

$$\sqrt{n} + (l\sqrt{n} - \sqrt{n})\frac{\theta l\sqrt{n} - \sqrt{n}}{n - \sqrt{n}}.$$

Also, for a point j not in S the expected number of points in S voting for j is

$$l\sqrt{n}\frac{\theta l\sqrt{n} - \sqrt{n}}{n - \sqrt{n}} \leq l\sqrt{n}\frac{\theta l\sqrt{n}}{n} \leq \sqrt{n}/4,$$

for $l \leq n^{1/4}/(2\sqrt{\theta})$. By Chernoff, we have that the probability that j is voted by more than $\sqrt{n}/2$ is at most $e^{-\sqrt{n}/48}$.

By union bound, we get that the probability that there exists a set S which is a union of l blobs that does not satisfy the (θ, α, β)-self-stability property with $\alpha = 1/l$, $\gamma = \theta/2$ is at most

$$n \cdot n^{l/2} \cdot e^{-\sqrt{n}/48} < 1,$$

for $l \leq n^{1/4}/(2\sqrt{\theta})$. ■

COROLLARY 1 The number of (θ, α, β)-self-determined communities in an affinity system (V, Π) satisfies $B(n) = n^{O(\log(1/\gamma)/\alpha)} \left(\frac{\theta \log(1/\gamma)}{\alpha}\right)^{O\left(\frac{1}{\gamma^2}\log\left(\frac{\theta\log(1/\gamma)}{\alpha\gamma}\right)\right)}$ and with probability $\geq 1 - 1/n$ we can find all of them in time $B(n)\text{poly}(n)$.

Proof: Consider a community size t. For any (θ, α, β)-self-determined community S and let p_S be the probability that S is in the list output by Algorithm in Theorem 1 with parameters θ, α, β, t. By Theorem 1 we have that $p_S \geq 1 - \delta$. By linearly of expectation we have that $\sum_S p_S$ is the expected number of (θ, α, β)-self-determined communities in the list output by our algorithm. Combining these, we obtain that $B(n)(1 - \delta) \leq \sum_S p_S \leq N_1(\delta)N_2(\delta)$ where $k_1 = \log(16/\gamma)/\alpha$, $k_2(\delta) = \frac{8}{\gamma^2}\log\left(\frac{32\theta k_1}{\gamma\delta}\right)$, $N_1(\delta) = n^{k_1}$ and $N_2(\delta) = O((2\theta k_1)^{k_2(\delta)}\log(1/\delta))$. By setting $\delta = 1/2$, we get the desired bound,

$$B(n) = n^{O(\log(1/\gamma)/\alpha)} \left(\frac{\theta \log(1/\gamma)}{\alpha}\right)^{O\left(\frac{1}{\gamma^2}\log\left(\frac{\theta\log(1/\gamma)}{\alpha\gamma}\right)\right)}.$$

Let $N = N_1(1/2)N_2(1/2)n$. By running the algorithm in Theorem 1 $2\log[N]$ times we have that for each (θ, α, β)-self-determined community S, the probability that S is not output in any of the runs is at most $(1/2)^{2\log(N)} \leq 1/N^2$. By union bound, with probability at least $1 - 1/n$, we output all of them. ■

A.3 Self-determined Communities in Multi-faceted Affinity Systems

Recall that a multi-faceted affinity system is a system where each node may have more than one rankings of other nodes. This may reflect, for example, that a person may have two rankings of other people, one corresponding to personal friends (in descending order of affinity), and one of co-workers. Suppose that each element i is allowed to have at most f different rankings $\pi_i^1, \ldots, \pi_i^f$. We say that the pair (S, ψ) is a multi-faceted community where $\psi : S \to \{1, \ldots, f\}$, if S is a community where $\psi(i)$ specifies which ranking facet should be used by element i. In other words, as before, let $\phi^\theta_{S,\psi}(i) := |\{s \in S | i \in \pi_s^{\psi(s)}(1 : \lceil\theta|S|\rceil)\}|$. Then (S, ψ) is an (α, β, θ)-multifaceted community if for all $i \in S$, $\phi^\theta_{S,\psi}(i) \geq \alpha|S|$, and for all $j \notin S$, $\phi^\theta_{S,\psi}(j) < \beta|S|$.

For a bounded f, it is not harder to find multifaceted communities than to find regular communities. Note that in all our sampling algorithms can be adapted as follows. Once a representative sample $\{i_1, \ldots, i_k\}$ of the community S is obtained, we can guess the facets $\psi(i_1), \ldots, \psi(i_k)$ while adding a multiplicative f^k factor to the running time. We can thus get the set S_2 approximating S in the same way as it is found in Algorithms 2 and 3 while adding a multiplicative factor of $f^{k_1+k_2}$ to the running time. We thus obtain a list $\mathcal{L}$ that for each multi-faceted community (S, ψ) contains set S_2 such that $\Delta(S_2, S) < \gamma t/8$:

Claim 1 *We can output a list $\mathcal{L}$ of $(f \cdot n)^{O(\log(1/\gamma)/\alpha)} \left(\frac{f \cdot \theta \log(1/\gamma)}{\alpha}\right)^{O\left(\frac{1}{\gamma^2} \log\left(\frac{\theta \log(1/\gamma)}{\alpha\gamma}\right)\right)}$ sets, such that for each multi-faceted community S there is an $S_2 \in \mathcal{L}$ such that $\Delta(S_2, S) < \gamma t/8$.*

It remains to show that:

Lemma 4 *Suppose that (S, ψ) is a valid (α, β, θ)-multifaceted community of size t. Given t and a set S_2 such that $\Delta(S_2, S) < \gamma t/8$, there is an algorithm that outputs S with probability $> f^{-8 \log n/\gamma^2}/2$.*

Moreover, a facet structure ψ' can be recovered on S so that (S, ψ') is an $(\alpha - \gamma/4, \beta + \gamma/4, \theta)$-multifaceted community.

Together with Claim 1, Lemma 4 shows that multifaceted communities can indeed be recovered in polynomial time.

THEOREM 5 Let S be an f-faceted (α, β, θ)-community. Then there is an algorithm that runs in $O(n^2)$ time and outputs S, as well as a facet structure ψ' on S such that (S, ψ') is an $(\alpha - \gamma/4, \beta + \gamma/4, \theta)$-multifaceted community with probability at least

$$(f \cdot n)^{-O(\log(1/\gamma)/\alpha)} \left(\frac{f \cdot \theta \log(1/\gamma)}{\alpha}\right)^{-O\left(\frac{1}{\gamma^2} \log\left(\frac{\theta \log(1/\gamma)}{\alpha\gamma}\right)\right)} f^{-O(\log n/\gamma^2)}.$$

Proof:(of Lemma 4). The algorithm is very simple. Guess a set U_2 of $m = 8 \log n/\gamma^2$ points in S_2; guess a function ψ_2 on U_2; output $S =$ the set of points that receive at least $(\alpha - \gamma/2)t$ votes according to (U_2, ψ_2).

Note that in the non-faceted case, by Hoeffding's inequality, with probability $> 1/2$ selecting a set U_2 as above and then selecting those points that receive at least $(\alpha - \gamma/2)t$ votes from U_2 would have yielded S. This is because each element of S receives at least $(\alpha - \gamma/8)t$ votes from elements of S_2, while each element of the complement S^c receives at most $(\beta + \gamma/8)t$ votes from elements of S_2. This reasoning extends to the multifaceted setting, *provided*, the function ψ_2 coincides with the function ψ on the elements of $U_2 \cap S$. This indeed happens with probability $\geq f^{-|U_2|} = f^{-8 \log n/\gamma^2}$, completing the proof of the first part of the lemma.

For the second part of the lemma we assume that the set S is known and we need to recover the facets ψ' that make S a community. Note that this step is necessary in order to verify that S is indeed a multifaceted community. There are two cases to consider.

Case 1: $t \leq 8 \log n/\gamma^2$. In this case we can find ψ by exhaustively checking all possibilities in time $O(q^{8 \log n/\gamma^2})$, which is the same as the probability of success of the first step.

Case 2: $t > 8 \log n/\gamma^2$. In this case we use linear programming to find a fractional version ψ_f of the function ψ first. In other words, we find a function $\psi_f : S \times \{1, \ldots, q\} \to [0, 1]$ such that (S, ψ_f) is a "community" on average:

1. for all $s \in S$, $\sum_{i=1}^{f} \psi_f(s,i) = 1$;
2. for all $x \in S$, $\sum_{s \in S} \sum_{i=1}^{f} \psi_f(s,i) \cdot \chi_{x \in \pi_s^i(1:\theta t)} \geq \alpha t$;
3. for all $y \notin S$, $\sum_{s \in S} \sum_{i=1}^{f} \psi_f(s,i) \cdot \chi_{y \in \pi_s^i(1:\theta t)} < \beta t$;

This linear program is feasible, since the original ψ is an integral solution to it. As a result, we obtain a fractional solution ψ_f satisfying the three conditions. To obtain ψ' we round ψ_f by sampling. In other words, we set $\psi'(s) = i$ with probability $\psi_f(s,i)$. By Hoeffding's inequality, since $t > 8 \log n/\gamma^2$, the sampling will preserve conditions 2 and 3 that were imposed on ψ_f up to an additive error of $\gamma/4$. Thus, by definition, (S, ψ') will be an $(\alpha - \gamma/4, \beta + \gamma/4, \theta)$-multifaceted community. ■

A.4 Extensions to weighted affinity systems and to the local model

We note that that Algorithm 4 can be combined with our reduction from weighted to unweighted communities to obtain a local algorithm for finding communities in the weighted case.

Extending the local approach to the multi-faceted setting is more involved, since the definition of $R(v)$ would need to be adapted to this setting. Indeed, the multi-faceted version $R_f(v)$ of $R(v)$ can be taken to be a random element voted by a random facet i of v. Then Algorithm 4 can be adapted by taking the threshold to be $\frac{(\alpha - 1/2)/(\theta^2 f^2)}{t}$, where f is the number of facets. Note that while an approximation to any community S can be found locally in near-linear time, finding the exact community S as well as the facet structure on S as in Lemma 4 will still take $f^{O(\log n/\gamma^2)}$ time.