

CS599: Structure and Dynamics of Networked Information (Spring 2005)
01/31/2005: Finding Communities in an Arbitrary Graph
Scribes: Karthik Dantu and Farnoush Banaei-Kashani

In a network, a “community” is vaguely defined as a cohesive set of interconnected nodes; for example, a community of friends (or a community of terrorists, to make it more interesting!) in a social network, or a web community in the WWW. Finding one or more interesting communities in the graph modeling a network is an intriguing problem to be discussed here. Besides the mere sociological interest, the application to the web may also help in structuring and differentiating search results.

As another example application, Granovetter [1] classifies ties between individuals in social networks based on their “strength”. A *weak* tie represents less interactions between the individuals as compared to a *strong* tie. He shows that social networks are community structures, where groups of individuals that are interconnected via strong ties form the communities and weak ties define the relations between the communities. In a particular experiment, he studied the process of job hunt through the ties in the social network and observed that more jobs are found through weak ties rather than strong ties (i.e., weak ties that represent less interactions are more effective). Considering the community structure of the social networks, he speculated that strong ties expose job seekers to the restricted number of positions available within their own community, whereas weak ties give them a chance to seek positions in many other communities probably with numerous other positions. In a separate experiment, a friendship network was studied in a high-school. In this experiment, it was observed that the graph representing only the strong ties, where each person is linked only to 2 best friends, is a fragmented graph containing many small, isolated communities. On the other hand, if we also consider weaker ties by including links to top 8 friends of each person, then the resulting graph has a giant connected component, and only few isolated smaller clusters. Defining a meaningful notion of community structure would allow us to make phenomena such as the effect of “weak ties” more precise, which in turn enables rigorous analysis and explanation of the observations such as those mentioned above.

There are various definitions for the notion of community, each emphasizing a particular property of the community in a specific application. Any such definition can be potentially valid and useful. For example, to capture the extent to which each node contributes to the community, one can define a community as a set of nodes each having at least d links, or at least a certain fraction of its links, to other nodes in the community. On the other hand, one can relax this per-node constrain and capture the connectedness of the community as a whole by constraining the total number of internal links among all nodes. Here, we study one particular definition for the notion of community, based on the edge density of a node set

Definition 1 Let $G = (V, E)$ be a graph. Denote by $e(S, T) = E \cap (S \times T)$ the set of edges with exactly one endpoint in S and one in T , and by $e(S) = e(S, S)$ the set of edges with both endpoints in S .

The edge density of a node-set S is defined as $\frac{|e(S)|}{|S|}$. We then define a community as a set S with high edge density.

In Section 1, we analyze a mincut-based algorithm to find communities in an arbitrary graph based on this definition. In Section 2 we introduce a more efficient 1/2-approximation algorithm for this problem. Other definitions for the notion of community that capture other properties of a community-like entity are discussed in successive sessions of this class.

1 A Mincut Based Algorithm

Finding the densest subgraph of exactly k nodes is an NP-hard problem, as can be seen by reducing from the k -clique problem, setting the density to be $k-1$. More strongly, Khot [2] recently rules out that existence of a

PTAS for densest k -subgraph, by showing that unless a certain complexity-theoretic assumption is violated, there is some $\alpha < 1$ such that no α -approximation to the densest subgraph can be found in polynomial time. Here, we consider the densest subgraph problem without the size constraint.

Problem 1 *In an arbitrary graph G , find the densest subgraph S , i.e., the set S maximizing $\frac{|e(S)|}{|S|}$.*

We begin with the decision problem: “Is there a subgraph S with $\frac{|e(S)|}{|S|} > \alpha$?” This constraint can be re-written as:

$$|e(S)| - \alpha|S| \geq 0 \quad (1)$$

Each internal edge of S contributes 2 to the total degree in S , and each edge leaving S contributes 1, so the number of internal edges $|e(S)|$ in S is $|e(S)| = 1/2(\sum_{v \in S} d(v) - |e(S, \bar{S})|)$. Substituting this in Equation 1 gives us

$$\sum_{v \in S} d(v) - |e(S, \bar{S})| - 2\alpha|S| \geq 0, \quad (2)$$

or equivalently:

$$\underbrace{\sum_{v \in V} d(v)}_{2|E|} - \underbrace{\left(\sum_{v \in \bar{S}} d(v) + |e(S, \bar{S})| + 2\alpha|S|\right)}_{\beta(S)} \geq 0 \quad (3)$$

As $2|E|$ is a constant (independent of S), this constraint is satisfiable iff it is satisfied by the set S with minimum $\beta(S)$. To find such a set, we formulate a mincut problem as follows. Consider the graph G' with $V' = V \cup \{s, t\}$, where s is connected to all vertices in V with a link of capacity 2α , and t is connected to all vertices in V with a link of capacity $d(v)$. The cost of the cut $(S + s, \bar{S} + t)$ in G' is exactly $2|E| - \beta(S)$. Therefore, one can solve this mincut problem with one of the classic mincut algorithms, obtaining a set S . If S satisfies the constraint in Equation 3, then it is a set of density at least α , if not, then no such set exists.

To summarize, one can formulate the solution to the Problem 1 as follows: If $\beta(S) > 2|E|$ then no α -community exists, else at least one α -community exists. Thus, by a binary search on all α , we can find the densest α -community. In fact, one can use *parametric max flow computation* [3] to compute mincuts for all values of α in one computation, thus avoiding the cost of repeated MinCut invocations in the binary search.

We can generalize the MinCut based algorithm to solve the following, slightly more general, problem:

Problem 2 *Given a graph G , find the densest subgraph S containing a specific vertex-set X (i.e., S maximizes $\frac{|e(S)|}{|S|}$ over all sets with $S \supseteq X$).*

One can solve this problem similarly by increasing the cost of the links from the vertices in X to s to ∞ . This will keep the nodes X in S , as otherwise the cost of the cut is infinity. The rest of the analysis stays the same.

2 A 1/2-Approximation

In some real-world graphs, such as the WWW or the graph of all friendships among people in the US, the running time of $O(mn^2)$ for MinCut computations is still prohibitively large. Thus, we are interested in faster, linear-time algorithms. As it is now known how to compute MinCut faster, we will have to settle for an approximation algorithm for now. The following greedy algorithm was first analyzed by Charikar [4].

Algorithm 1 A Greedy 1/2-Approximation Algorithm for finding dense subgraphs

Let $G_n \leftarrow G$.
for $k = n$ downto $|X| + 1$ **do**
 Let $v \notin X$ be the lowest degree node in $G_k - X$.
 Let $G_{k-1} \leftarrow G_k - v$.
end for
Output the densest subgraph among $G_n, \dots, G_{|X|}$

Claim 1 *This algorithm is a 1/2-approximation.*

Proof. Let S be the optimal subgraph for the given graph $G = (V, E)$. If our algorithm outputs S , then it is clearly optimal. If not, then at some point, we must have deleted a node $v \in S$. Let G_k be the graph right before the first $v \in S$ was removed. Because S is optimal, removing v from it would only make it worse, so

$$\frac{|e(S)|}{|S|} \geq \frac{e(S-v)}{|S|-1} \geq \frac{|e(S)|-d(v)}{|S|-1}.$$

Multiplying through with $|S|(|S| - 1)$ and rearranging gives us $d(v) \geq \frac{|e(S)|}{|S|}$.

Because G_k is a supergraph of S , the degree of v in G_k must be at least as large as in S , so $d_{G_k}(v) \geq d_S(v) \geq \frac{|e(S)|}{|S|}$. The algorithm chose v because it had minimum degree, so we know that for each $u \in G_k \setminus X$, we have $d_{G_k}(u) \geq d_{G_k}(v) \geq \frac{|e(S)|}{|S|}$. We thus obtain the following bound on the density of the graph G_k :

$$\begin{aligned} \frac{|e(G_k)|}{|G_k|} &\geq \frac{\sum_{u \in S} d_S(u) + \sum_{u \in G_k \setminus S} \frac{|e(S)|}{|S|}}{2|G_k|} \\ &= \frac{2|e(S)| + |G_k \setminus S| \frac{|e(S)|}{|S|}}{2|G_k|} \\ &\geq \frac{|e(S)|}{|S|} \cdot \frac{|S| + |G_k \setminus S|}{2|G_k|} \\ &= \frac{|e(S)|}{2|S|} \end{aligned}$$

The graph that the algorithm outputs is certainly no worse than G_k , as G_k was available as a potential solution. Hence, the algorithm is a 1/2-approximation. ■

References

- [1] M. Granovetter, *The strength of weak ties*. American Journal of Sociology, 78(6):1360-1380, 1973.
- [2] S. Khot, *Ruling Out PTAS for Graph Min-Bisection, Densest Subgraph and Bipartite Clique*. Proceedings of FOCS 2004.
- [3] G. Gallo, M. Grigoriadis, R. Tarjan, *A fast parametric maximum flow algorithm and applications*. SIAM Journal on Computing 18/1, 2/1989, pp. 30-55.
- [4] M. Charikar, *Greedy Approximation Algorithms for Finding Dense Components in Graphs*. Proceedings of APPROX 2000.