Community detection and its use in Real Graphs

András Bóta
Department of Computer Science
University of Szeged, Hungary
bandras@inf.u-szeged.hu

László Csizmadia
Department of Computer Science
University of Szeged, Hungary
cheeseme@freemail.hu

András Pluhár
Department of Computer Science
University of Szeged, Hungary
pluhar@inf.u-szeged.hu

ABSTRACT
We survey and unify the methods developed for finding overlapping communities in Small World graphs in the recent years. The results have impact on graph mining; we give some demonstration of this.

General Terms
Theory

Keywords
data mining, graphs, communities

1. INTRODUCTION
The discovery of Small World graphs has changed the direction of interest in graph theory profoundly. These graphs are different from those that were studied before, and also the questions that were asked about those. It is not easy to collect the information to build such a graph, or give models to generate it. The sheer size of the real problems prohibits most of the time consuming algorithms, so the researcher has to fall back on simpler heuristics, sometimes derived from physical intuition [3, 5, 20]. Following the usual notation, a graph G has vertex set $V(G)$, edge set $E(G)$. If the later one consists of ordered pairs, then G is directed, and an edge might be also weighted.

An intriguing question is the classification of vertices of a graph. One can consider the usual clusters and also overlapping sets, that we call communities. Here we concentrate on the possible definitions, search and use of communities. While for clustering both the top down and bottom up algorithms are used for defining and finding the classes, all known algorithms for communities are bottom up.

2. SOME ALGORITHMS
Here we consider only three algorithms. The selection is arbitrary, although has some justification. Maybe the first algorithm that was used for finding communities is the N^{++}. However, since we could get no permission to use the data set it was designed for, it has not been published yet in English. After that several similar algorithms were proposed; unfortunately the qualities of implementations differ so it is not easy to compare them. The k-Clique percolation method was the first widely known algorithm, which was also applied to real world problems. Edge clustering is the third algorithm we mention; it has mainly theoretical interest.

2.1 The N^{++} algorithm.
[23, 11] It is a generic algorithm, with arbitrary functions

$$f : 2^{V(G)} \times V(G) \to \mathbb{R}$$

and $c : \mathbb{N} \to \mathbb{R}$. Here $f(A, x)$ describes the strength of a community A with a vertex x. Then the algorithm joins x to A if $f(A, x) \geq c(|A|)$. The **Build** routine gets the first approximation of communities $\mathcal{K}$ in a bottom up way.

The pseudo-code of Build

begin(Build)

Input G, k, c (`max k-size c-communities`)

Let $\mathcal{K} := V(G)$ (`nodes are communities.`)

For $i = 1$ to k

$\forall A \in \mathcal{K}$, $x \in V(G)$ if $f(A, x) \geq c(|A|)$ then put $A \cup \{x\}$ into $\mathcal{K}$.

Remove all $A \in \mathcal{K}$, for which $A \subset B \in \mathcal{K}$, and $A \neq B$.

Print $\mathcal{K}$, "c-communities of G up to size k."

end(Build)

After running Build, we use **Merge** to glue communities that are almost identical. Let C be a graph, where $V(C) = \mathcal{K}$, and $(A, B) \in E(C)$ if $A \cap B$ is "big" then changes $\mathcal{K}$ to $(\mathcal{K} \setminus \{A, B\}) \cup \{A \cup B\}$. Then the components of C are declared to be the communities. The practice suggested the following set-ups. The big means the 60% of the smaller set. The function $f(A, x)$ depends on the number of paths with length one and two from x to A. That is to get the communities containing x, it is enough to search $N^{++}(x) := N(N(x))$.

Some similar methods are listed in [13].

2.2 k-Clique percolation.

[21] Here a $k \in \mathbb{N}$ is fixed. After finding all k-size clique in G, the graph Q_k is considered such that the vertices of Q_k are these cliques, and $(A, B) \in E(Q_k)$ iff $|A \cap B| = k - 1$. Finally a k-community is the unions of cliques of a connected components.

2.3 Edge clustering.

[22, 25] One chooses an arbitrary clustering on the set of edges. Then the communities are defined as the set of endpoints of the clusters.

These methods differ in output, i. e. in the type of communities, and in the computing costs. Although the edge clustering is easy to compute, it has serious drawbacks in use. (First of all is that the overlap among communities is maximum one vertex.) The N^{++} and Clique percolation are more promising; here the implementation issues are crucial. For small world graphs both can perform almost in linear time, which is a natural requirement if one wants to deal with real problems.[1]

2.4 A unified view

These algorithms, and those that were mentioned but not listed, has a common core. Their execution consists of two steps. In the first a hypergraph $\mathcal{F} = (V, \mathcal{H})$ is defined (and computed), where $V = V(G)$, the original point set of the graph G, and $\mathcal{H} \subset 2^V$. The elements of $\mathcal{H}$ can be considered as the building blocks of the communities. In the second step one endows the set $\mathcal{H}$ with an appropriate d distance function and thereby establishes a metric space $\mathcal{M} = (\mathcal{H}, d)$. Then a chosen clustering algorithm is executed on $\mathcal{M}$, yielding a set of clusters $\mathcal{C}$. Finally, the arising clusters are associated to the subsets of V such that $K_i = \cup_{H \in C_i \in \mathcal{C}} H$, where K_i, the ith community corresponds to C_i, the ith cluster and K_i is just the union of those hyperedges that belong to C_i.

In the case of the mentioned algorithms $\mathcal{H}$ consist of vertex sets of the small dense subgraph, k-cliques and the edges, respectively. The distance functions are represented by an appropriate graph $\mathcal{D}$, take the value one if there is an edge, infinity otherwise. In the first case $(K_i, K_j) \in \mathcal{D}$ if $|K_i \cap K_j|$ is big enough, in the second if $|K_i \cap K_j| = k - 1$, while we left this as a parameter in the third case.

3. EVALUATION

Since more or less all community (or cluster) definitions are arbitrary [18], there are several ideas to measure their goodness. This is a crucial point and naturally the viewpoint of researches differ. There are direct and indirect methods to assess the usefulness of communities, the following list is far from being complete.

3.1 Appearance, parametrization

First of all, one has to run the algorithms, get the outputs and possible make mathematical predictions for certain graph classes. That is an important factor is the speed of these algorithms. However, it is not easy to compare the real speed of these algorithms since it depends strongly on the implementations and test graphs (being real or theoretical). Definitely all three algorithms, and perhaps most algorithms in that family we described in subsection 2.4 are fast, and designed to solve huge problems. In subsection 3.4 we recur to this problem, and report some date on time and a goodness measure (modularity) of the solutions.

The clique percolation method is appealing from both theoretical and practical view. For Erdős-Rényi random graphs the clique percolation process is thoroughly studied and well understood, [6]. It was reported to be useful also in practice, [1]. However, it sometimes gives too large communities and the parametrization is elusive, since one has to decide for which value k to be chosen?

The N^{++} algorithm looks arbitrary, and do not yield for theoretical investigations. Its main advantages are the speed, the small diameter of the communities and its robustness. The edge clustering methods are not well studied or tested in practice. Their inherent problem is that communities derived this way may have only one common element, what is too restrictive in real graphs.

We tested on these algorithms on some benchmark graphs, let us illustrate our findings on the famous Zachary graph, see [26]. This is a friendship graph of a karate club that split into two parts, A and B. Part A is centered around their Japanese master, while part B is led by the club administrator. The the clique percolation method gives three communities for $k = 3$ with sizes 3, 6 and 24. For $k = 4$ there are also three communities, the sizes are 4, 4 and 7, while for $k = 5$ there is one community of size six. Here a blend of $k = 3$ and $k = 4$ seems to be appropriate, and the communities are on the two sides of border where the split occurred. The N^{++} algorithm results in twelve communities, four of size three, five of size four, one of size six, and two of size seven. All but one communities are entirely either in A or in B. One might argue that the club was always one the verge of demise that happened at the end.

3.2 Graphical.

Another way is to compare the communities with some visualized form of the graph; this was the most common approach in the early publications. Indeed, the clustering methods provide classes that conform the eye. Assessing communities (permitting overlapping) are harder, since visualization is not an obvious task anymore. Some ideas, like showing the intersection graph of communities can help. However, this approach has certain limits; it works only for small graphs and it is always subjective.[2]

Another possibility is to draw some derived graphs. Among these the intersection graph H of the communities performed best. Here the vertices are the communities of G, and an edge is drawn if the communities associated to the vertices

[1] This means millions of vertices. The N^{++} available in the Sixtep software, while the Clique percolation in the CFinder.

[2] For graph visualization the so-called *force directed* algorithms performed best. However, these usually take $O(n^2)$ time that prohibit the use when n is several thousand or million.

has a non-empty overlap. That is $I(G) = (V(H), E(H))$, where $V(H) = \mathcal{K}$ and $(C_i, C_j) \in E(H)$ if $|C_i \cap C_j| > 0$.

Again, for the Zachary graph the clique percolation method gives an unconnected graph H. The intersection graph H based on the N^{++} algorithm is more delicate. It consist of two dense subgraph with one common vertex x. A four element community corresponds to vertex x, and this community contains the master (1), the administrator (33) and the vertices labeled by 3 and 9. The community was almost a clique, except that the master and the administrator were not friends. When the split occurred, 3 and 9 ended up in different parts destroying completely the only community that connected the two parts. One might speculate that the friendship of 3 and 9 was responsible for the cohesion of the club, and when it could not take more pressure they took parts which meant the end of the club, too.

3.3 Random Small World Graphs

There are several ways to generate random graphs having similar properties that of real Small World graphs, [2, 8]. From those we tried out the Preferential Attachment (PA) and the Vertex Copy (VC) models. In both of these models the graph is build step by step, while the neighborhood of the newly arrived vertex x is chosen differently. In the PA model the new vertex x brings k new edges, and the other end of these edges are at an old vertex y with probability proportional to $d(y)$ and taken independently from each other. In the VC model an old vertex s is selected uniformly, and the new vertex x takes vertices independently from $N(s)$ with a prescribed probability p.

The results are far from being conclusive, and indeed tell more about the models (PA and VC) than the community algorithms (CPC, N^{++}). Note, that a different approach, using random intersection graphs, is investigated in [24].[3] Here we illustrate it on two sets of graphs that approximately belong to the same category. For all these the number of vertices is 100, G_1 and H_1 were generated by the PA model, $|E(G_1)| = 192$, $|E(H_1)| = 358$ while G_2 and H_2 come from the VC model with $|E(G_2)| = 151$ and $|E(H_2)| = 378$. The #C and #CO mean the number of clusters and communities, while the column with head k contains the number of communities of size k. At the case of CPM the column k refers to the parameter of the algorithm instead, that is the algorithm was run for $k = 3, 4, \ldots$ The number of clusters were determined by a modularity maximization algorithm (a version of Newman), see the next subsection.

Graph and Method	#C	#CO	3	4	5	6	7	> 7
G_1 / CPM	10	7	7					
G_1 / N^{++}	10	9	5	0	0	2	1	1
G_2 / CPM	9	17	13	4				
G_2 / N^{++}	9	22	8	7	2	4	1	0
H_1 / CPM	6	10	7	3				
H_1 / N^{++}	6	37	5	2	3	9	7	12
H_2 / CPM	6	24	4	8	6	6		
H_2 / N^{++}	6	26	8	3	2	5	1	7

[3]For intersection graphs the CPM gives too large communities sometimes. A possible remedy is to fix the diameter, like in N^{++}.

3.4 Modularity.

The Newman modularity [20] is the following function of a graph G and its partition:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j),$$

where $m = |E(G)|$, A_{ij} is the adjacency matrix of G, k_i is the degree, c_i is the cluster of the ith vertex, and $\delta(c_i, c_j)$ is the Kronecker symbol. The clustering algorithms may be based on some mathematical/physical heuristics like edge-betweenness (EB), eigenvectors (EV), label propagation (LP), spin glass (SG), walk trap (WT), or try to maximize the modularity function itself on the set of all partitions with a greedy algorithm (Gr). The formula can be generalized to communities [19]. One write s_{ij} instead of $\delta(c_i, c_j)$, where s_{ij} is an arbitrary similarity measure between vertices i and j. (In [19] the u_i is a probability distribution of i over the communities, and $s_{ij} = \langle u_i, u_j \rangle$, but it could be $\|u_i - u_j\|$ form any norm.) On the other hand, it is possible to get communities by maximizing the modularity function. The findings of [16] show the cluster and community structure cannot be measured on the same scale, some additional weighting must be introduced to solve this. The algorithms were tested for some graphs, we illustrate the results on the already mentioned Zachary graph. The clusterings are followed the Clique Percolation (CPM) with clique sizes $k = 3$ and $k = 4$, and N^{++} with its default parameters. The running time is in seconds, $\#C$ stands for the number of clusters or communities, whatever it applies.

Method	Modularity	Running time	#C
EB	0.4013	0.0100	5
EV	0.3727	0.0000	3
Gr	0.3807	0.0000	3
LP	0.4020	0.0000	3
SP	0.4063	1.1500	6
WT	0.4198	0.0000	4
CPM 3	0.2438	0.012	3
CPM 4	0.2557		3
N^{++}	0.1947	0.6690	12

One can evaluate cluster/community algorithms in indirect ways. That is by taking a problem in which the communities might have predictive value, and check the usefulness of these. We have observed dependencies among functions in some social graphs (telecommunication, friendship, Erasmus contracts etc.), and practically all methods provided useful hints. However, here the use of communities greatly outperforms the methods which use only clusters.

3.5 Refinements, time and orders.

One can conduct similar studies like the graphical method if have some functions that are defined on the vertices or the edges. Again, we have seen some highly subjective but still robust phenomena that might deserve to be mentioned.

First of all, the clusters are usually much bigger than the communities, and their number is less.

The number of communities might follow power law, although even to test this is impossible.

The communities are usually within the clusters, and give a fine structure of those larger classes. However, the reverse direction is also detected, the clusters might give information on communities. To be more precise, the most interesting communities are those ones in which elements belong to several clusters.

In social graphs we confirmed the role of the weak links described in [14], and also tested the different algorithms. The communities given by N^{++} are containing strong edges almost exclusively, while most of the weak edges are among communities. On the other type of small world graphs, the so-called technical graphs[4] there are no such effects. We used data from [17]. (The CPM does not give good results with any k, perhaps its performance is too sensitive to the measurement errors, missing data.)

The social graphs might have natural vertex attribute, the time when a vertex has been joined to the net. This order may not be manifested in the clusters if one considers the whole graph, but shows remarkable coincidence when restricting the graph to the neighborhood of a fixed vertex. In that case the clusters usually can be interpreted with some interval of time or spatial restrain. Note, that communities may cross the borders of clusters.

3.6 Weights.

Dealing with weighted graphs is difficult. It turns out that for the indirect methods the numerical results are more reliable. While all these methods can be extended to weighted graphs, the performance of them is little known [7].

In the rest we outline a model which is an example for indirect evaluation. The infection models are central in applications of real graphs [4], but to build appropriate ones is far from being trivial. The main points are (i) which model to choose, (ii) what are the significant variables and (iii) how to decide the values of the parameters. Our investigations concentrated on two problems in corporate banking, default (failing in paying debt) [9] and delay (in paying debt) [10]. We have to stress, although the two problems look similar, there are subtle differences. The main similarity is that these processes can be considered as some kind of "infectious disease." However, one has to be careful since financial difficulties may come from intrinsic reasons. (The rise or fall of the economic might be accounted by taking a fictitious node.) So the task is to devise a methodology that, given the *a priori* probabilities of some problem (say the default), estimates the *a posteriori* probabilities. The difference of these probabilities is recognized as *network effect* in the certain problem. The characteristics of the problem (e. g. no recovery, the probability of transmission is not constant) exclude the SIR or SIS models that play central role in Epidemiology. The best suited model is the Independent Cascade.

[4]In social graphs the presence of edges (x, y) and (x, z) increases the conditional probability of the the edge (y, z), while in the technical graphs this probability is decreased in that case.

3.7 Independent Cascade Model (IC)

This model is due to Domingos and Richardson [12], but an equivalent is in [15]. Here an edge weighted graph G is given, where to the edge (v, w) a probability $p_{v,w}$ is associated. The process of infection goes as follows. In the 1st step the set of infected vertices F_1 considered active, that is $F_1 = A_1$. In general for a vertex $w \in V(G) \subset F_{i-1}$ gets infected with probability $p = \prod_{v \in A_{i-1}} p_{v,w}$, and in that case $w \in F_i$. Note that the infected vertices may transmit the disease only in the very next step, that is $A_i = F_i \setminus F_{i-1}$. If for an i $F_i = F_{i-1}$, then the process halts.

3.8 Weighting and optimization.

First of all, one has to modify the IC model for effective use. Since the probabilities are assesses by simulation, it is natural to subject the a priori infection to this, too [9]. While the modified IC model provides extreme flexibility for modeling complex system, it is also very difficult to find appropriate transmission probabilities, or even an measure that tells from the better from the worse. The weights are assigned by a standard AI method, making a training set and a test set on the past data. A possible solution for the measurement is the use of the *gain curve*.

The vertices of the graph are ordered monotone decreasing way by their infection computed on the training set. Let $w_1, \ldots, w_n$ be the values of the same vertices given in the test set. Then the function $\text{gain}(x) = \sum_{i \le x} w_i / \sum_{i=1}^{n} w_i$; and $\int_{x=1}^{n} \text{gain}(x)dx$ should be maximized.

An estimation for an edge probability $p_{v,w}$ is based on the vertex and edge attributes that are available in the data. To maximize the performance measured by the gain function, a systematic search was done to try out the possible combinations of the reasonable functions of the considered variables. This included linear, quadratic, logarithm, exponential and sigmoid functions. The final aggregation of these transformed values was also treated this way. To find the best parameters of this function, a grid search was used.

3.9 Results.

Here we single out only one experiment out of several ones. A thorough study was executed on the data of one of the largest Hungarian bank (OTP), and the findings published in [10]. Here the estimation of default probabilities of certain clients (small and middle enterprise sector) was the goal. The OTP Bank Corporate transaction database was used, where the graph building period was from August 2008 to April 2009 (6 months) and the infection period was from February 2009 to April 2009 (the last 3 months from it). For default event observation two periods were chosen: a longer one from May 2009 to April 2010 (12 months), and a shorter one from May 2009 to April 2010 (3 months).

I. It turned out that shorter periods (3 month) gave better models than those were based on longer periods.

II. The direction of the edges counts, it should be taken as buyer - provider, i. e. if x sends money and y receives it than (x, y).[5]

[5]There is some effect even when the edges are taken indi-

III. The variables and findings worth considering follow.

(i) Community information. (If the edge belongs to a community?)

(ii) The edge (x, y) inherits the variables of x (but not y).

(iii) Relative traffic, that is the transfer of the edge divided by the sum of all incoming transfer.

(iv) The age of the client. (How old is the company?)

(v) Behavioral types. (queuing on the account, overdraft etc.)

Even though, the most significant variables are the ones listed in (i) and (iii).

Based on this, we found an expected 3-4 to even 10-12 times lift in the different segments [10]. The fact that a vertex x is in a same community with an infected increases the chance of x's infection by a factor three. Note that there were similar findings in [9] on different data. However, in [9] the parameter values for the IC based model were set by using trial and error, while in [10] a more sophisticated search was done. The computations were carried out by the use of Sixtep software.

4. ACKNOWLEDGMENTS

The first author was partially supported by the the joint grant of Hungarian Government and the European Union in the framework of the Social Renewal Operational Programme, project no. TÁMOP 4.2.2-08/1-2008-006, while the third author was partially supported by the Hungarian National Science Foundation Grants OTKA K76099 and also by the grants TÁMOP-4.2.2/08/1/2008-0008 and TÁMOP-4.2.1/B-09/1/KONV-2010-0005.

5. REFERENCES

[1] B. Adamcsek, G. Palla, I. J. Farkas, I. Derényi, and T. Vicsek. Cfinder: Locating cliques and overlapping modules in biological networks. *Bioinformatics*, 22(8):1021–1023, February 2006.

[2] R. Albert and A. L. Barabási. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[3] R. Albert and A. L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, January 2002.

[4] M. Boguná, R. Pastor-Satorras, and A. Vespignani. Absence of epidemic threshold in scale-free networks with connectivity correlations. *arXiv:cond-mat/0208163v1 [cond-mat.stat-mech]*, 2002.

[5] B. Bollobás. *Modern Graph Theory*. Springer, New York, New York, 1998.

[6] B. Bollobás and O. Riordan. Clique percolation. *Random Structures and Algorithms*, 35(3):294–322, October 2009.

[7] A. Bóta. Applications of overlapping community detection. In $(CS)^2$ - *Conference of PhD Students in Computer Science*, page . , June 2010.

[8] A. Cami and N. Deo. Techniques for analyzing dynamic random graph models of web-like networks: An overview. *Networks*, 51(4):211–255, July 2008.

[9] A. Csernenszky, G. Kovács, M. Krész, A. Pluhár, and T. Tóth. The use of infection models in accounting and crediting. In *Challenges for Analysis of the Economy, the Businesses, and Social Progress*, pages 617–623. , November 2009.

[10] A. Csernenszky, G. Kovács, M. Krész, A. Pluhár, and T. Tóth. Parameter optimization of infection models. In $(CS)^2$ - *Conference of PhD Students in Computer Science*, June 2010.

[11] L. Csizmadia. *Recognizing communities in social graphs*. MSc thesis, University of Szeged, Hungary, 2003.

[12] P. Domingos and M. Richardson. Mining the network value of costumers. In 7th Intl. Conf. on Knowledge Discovery and Data Mining, pages 57–66. ACM, August 2010.

[13] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, February 2010.

[14] M. Granovetter. The strength of weak ties. *American Journal of Sociology*, 78(6):1360–1380, May 1973.

[15] M. Granovetter. Threshold models of collective behavior. *American Journal of Sociology*, 83(6):1420–1443, May 1978.

[16] E. Griechisch. Comparison of clustering and community detection algorithms, the extension of the modularity. In $(CS)^2$ - *Conference of PhD Students in Computer Science*, June 2010.

[17] C. A. Hidalgo, B. Klinger, A. L. Barabási, and R. Hausmann. The product space conditions the development of nations. *Science*, 317(5837):482–487, July 2007.

[18] J. Kleinberg. An impossibility theorem for clustering. In *Advances in Neural Information Processing Systems (NIPS)* **15**, 2002.

[19] T. Népusz, A. Petróczi, L. Négyessy, and F. Bazsó. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 77(1):016107, January 2008.

[20] M. Newman. The structure and function of complex networks. *arXiv:cond-mat/0303516 v1*, 2003.

[21] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(9):814–818, June 2005.

[22] A. Pluhár. On the clusters of social graphs based on telephone log-files. *Research Report*, 2001.

[23] A. Pluhár. Determination of communities in social graphs by fast algorithms. *Research Report*, 2002.

[24] D. Stark. The vertex degree distribution of random intersection graphs. *Random Structures and Algorithms*, 24(3):249–258, May 2004.

[25] T.S.Evans and R.Lambiotte. Line graphs of weighted networks for overlapping communities. *arXiv:0912.4389v2 [physics.data-an]*.

[26] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.

rected. This is due to another network effect, since the edges are describing an economic interdependence/same sector.