Finding communities in networks in the strong and almost-strong sense

Sonia Cafieri, Gilles Caporossi, Pierre Hansen, Sylvain Perron, Alberto Costa

Les Cahiers du GERAD ISSN: 0711–2440

Finding Communities in Networks in the Strong and Almost-Strong Sense

S. Cafieri, G. Caporossi,
P. Hansen, S. Perron,
A. Costa

G–2011–74

December 2011

Finding Communities in Networks in the Strong and Almost-Strong Sense

Sonia Cafieri

Laboratoire MAIAA, École Nationale de l'Aviation Civile
F-31055 Toulouse, France
sonia.cafieri@enac.fr

Gilles Caporossi
Pierre Hansen*
Sylvain Perron

GERAD & HEC Montréal
Montréal (Québec) Canada, H3T 2A7
* *Also at LIX, École Polytechnique, F-91128 Palaiseau, France*
gilles.caporossi@gerad.ca
pierre.hansen@gerad.ca
sylvain.perron@gerad.ca

Alberto Costa

LIX, École Polytechnique
F-91128 Palaiseau, France
costa@lix.polytechnique.fr

December 2011

Les Cahiers du GERAD
G–2011–74

Abstract

Finding communities, or clusters, or modules, in networks can be done by optimizing an objective function defined globally and/or by specifying conditions which must be satisfied by all communities. Radicchi et al. [Proc. Natl. Acad. Sci. U.S.A. **101**, 2658 (2004)] define a susbset of vertices of a network to be a community in the strong sense if each vertex of that subset has a larger inner degree than its outer degree. A partition in the strong sense has only strong communities.

In this paper we first define an enumerative algorithm to list all partitions in the strong sense of a network of moderate size. The results of this algorithm are given for the Zachary karate club dataset, which is solved by hand, as well as for several well-known real-world problems of the literature. Moreover, this algorithm is slightly modified in order to apply it to larger networks keeping only partitions with the largest number of communities. It is shown that some of the partitions obtained are informative, although they often have only a few communities, while they fail to give any information in other cases, having only one community. It appears that degree two vertices play a big role in forcing large inhomogeneous communities. Therefore, a weakening of the strong condition is proposed and explored: we define a partition in the almost-strong sense by substituting a non-strict inequality to a strict one in the definition of strong community for all vertices of degree two. Results, for the same set of problems as before, then give partitions with a larger number of communities and are more informative.

Acknowledgments: Financial support by Grants Digiteo 2009-14D "RMNCCO" and Digiteo 2009-55D "ARM" is gratefully acknowledged. G.C., P.H., and S.P. were partially supported by FQRNT (Fonds de recherche du Québec – Nature et technologies) team grant PR-131365. S.P. has been partially supported by NSERC (Natural Sciences and Engineering Research Council of Canada) grant 327435-06.

1 Introduction

Networks, or graphs, are composed of a set of vertices and of a set of edges which join pairs of vertices [1, 2, 3, 4, 5, 6]. As vertices can be associated with entities and edges with relations defined on all pairs of vertices, networks are extensively and increasingly used in many areas of science and its applications.

An ubiquitous phenomenon in networks is the presence of communities (also called clusters or modules) where inner edges, joining two vertices of the same community, are dense and cut edges, joining two vertices of different communities, are sparse. As shown below, there are many ways to define and/or evaluate communities as well as partitions of the vertex set into communities. There are three main approaches to the evaluation of a partition of the vertex set of a network:

(i) one can specify a heuristic and evaluate informally the result obtained. For instance, Girvan and Newman [7] present a hierarchical divisive heuristic in which the edge with largest betweenness is iteratively removed and communities correspond to connected components obtained each time a community is split into two. This heuristic therefore proceeds from an initial (trivial) partition in a single community containing all vertices to a final partition in which each community contains a single vertex. Divisions of communities are graphically represented by a dendrogram;

(ii) a criterion function, to be maximized or minimized, is chosen. There are many examples, the best known of which is *modularity*, initially proposed as a stopping rule for the divisive heuristic mentioned above and later considered as an independent criterion. Other well-known criteria are the k-way cut [8, 9], the normalized cut [10, 11], the ratio cut [9], the modularity density and its variants [12, 13], and strength maximization subject to strong or weak constraints on the communities (see Section 2) [14, 15]. More recently, several promising criteria have been put forward: information compression [16], maximum likelihood and the expectation maximization (EM) algorithm [17] and the constant Potts model [18].

(iii) conditions to be satisfied by a community are specified. Several such conditions have been proposed; the early ones are reviewed in the book [2]. They include the cliques, in which every pair of vertices must be joined by an edge, the k-cliques in which the indegree of each vertex must be at least k, and LS (Luccio-Sami) set [19], i.e., a set of vertices S such that each of its proper subsets has more ties to its complement within S than to the outside of S. These three criteria tend to be too stringent and/or too difficult to compute, except on the smallest networks. Two others intuitive and well-known criteria have been proposed by Radicchi et al. [20]: a subset S of vertices of a network forms a *community in the strong sense* if the number of neighbors of each vertex within S is larger than the number of neighbors outside S. A set of vertices S forms a *community in the weak sense* if the sum, for all of its vertices, of the difference between the number of neighbors within S and the number of neighbors outside S is positive. As these concepts capture the intuitive idea of a community, they are very often cited. Recently, weakened versions have been proposed, in which instead of comparing the numbers of neighbors within and outside the community, one compares the numbers of neighbors within a community, and outside that community but within another specific community [13].

In this paper we focus on communities in the strong sense. We first give an exact algorithm for finding all partitions of a network into communities in the strong sense. There are several reasons to consider exact algorithms instead of heuristics: while exact algorithms are limited to problems of moderate size (in modularity maximization the largest instance solved exactly up to now is the S838 dataset which has 512 vertices and 819 edges, solved by the column generation algorithm [21]). Smaller networks are of interest in several domains, e.g., sociology, mathematical psychology and ecology. Then, using an exact algorithm allows discrimination of errors due to an inadequate model or to a non-optimal resolution by heuristics. Moreover, exactly solved problems provide a benchmark for evaluating and comparing heuristics. Comparing the exact solution with those obtained by some heuristic focuses on parts of the problem which are badly solved and can suggest how to improve the heuristic. Conversely, heuristics can be used in order to accelerate one or several steps of the exact algorithm, e.g., by providing a hot start. A more complete discussion of merits and demerits of exact algorithms and heuristics is given in the first section of [21].

The paper is organized as follows: definitions and notation are specified in the next section. The main algorithm, called Strong Communities (SC), is presented in Section 3. Its application is illustrated step-by-

step by hand resolution of the classical Zachary karate club network [22]. A variant of SC, called Almost-Strong Communities (ASC), and in which the definition of strong community is weakened, is presented in Section 4. Results obtained by both algorithms for several real-world networks, described in the Pajek database (`http://vlado.fmf.uni-lj.si/pub/networks/data/`), are presented and discussed in Section 5. Conclusions are drawn in Section 6.

2 Definitions and notation

We denote a network, or graph, by $G = (V, E)$ where V is the set of n vertices, and E is the set of m edges which join pairs of vertices. A vertex v_j is represented by a point and an edge $e_{i,j} = \{v_i, v_j\}$ by a line joining its two end vertices v_i and v_j. The shape of this line does not matter, only the presence or absence of an edge is important. A loop $e_{i,i} = \{v_i, v_i\}$ is an edge for which both end vertices coincide. In a simple graph, there is at most one edge between any pair of vertices, and no loops. The degree k_i of a vertex $v_i \in V$ is the number of edges incident with v_i. In this paper, we focus on unweighted simple graphs.

The adjacency matrix $A = (a_{i,j})$ of G is a square n by n matrix such that $a_{i,j} = 1$ if vertices v_i and v_j are joined by an edge, and equal to 0 otherwise.

A subgraph $G_S = (S, E_S)$ of a graph $G = (V, E)$ induced by a set of vertices $S \subseteq V$ is a graph with vertex set S and edge set E_S equal to all edges with both vertices in S. Such a subgraph corresponds to a community and many heuristics, as well as exact algorithms, aim at finding a partition of V into pairwise disjoint nonempty subsets $S_1, S_2, \ldots, S_N$ inducing subgraphs of G. Various objective functions have been proposed for evaluating such a partition. Roughly speaking, one seeks modules which contain more inner edges (with both vertices in the same module) than cut edges (with vertices in different modules). The degree k_i of the vertex v_i can be split in two parts: the indegree k_i^{in} or number of neighbors of v_i within its community and the outdegree k_i^{out} or number of neighbors of v_i outside its community. Several concepts of community follow.

A set S of vertices is a clique if all pairs of vertices of S are joined by an edge, i.e., $k_i = |S| - 1$, $\forall v_i \in S$. A set S is a k-clique if every vertex of S has at least k neighbors within S, where k is a parameter.

In 2004, Radicchi et al. [20] defined a *community in the strong sense* as a subgraph all vertices of which have larger indegree than outdegree: $k_i^{in} > k_i^{out}$, $\forall v_i \in S$ or equivalently $\sum_{j \in S} a_{i,j} > \sum_{j \in V \setminus S} a_{i,j}$, $\forall v_i \in S$. Note that the concept of defensive alliance, studied in graph theory (see the thesis [23] and references therein), is very close to that of community in the strong sense and is obtained by substituting non-strict inequalities to strict ones.

In the same paper [20], a *community in the weak sense* is defined as a subgraph for which the sum of vertex indegrees is larger than the sum of vertex outdegrees: $\sum_{v_i \in S} k_i^{in} > \sum_{v_i \in S} k_i^{out}$ or equivalently $\sum_{v_i \in S} \sum_{v_j \in S} a_{i,j} > \sum_{v_i \in S} \sum_{v_j \in V \setminus S} a_{i,j}$. As an inner edge contributes by two to the sum of the in-degrees and a cut edge contributes by one to the sum of outdegrees, the number of inner edges in a community in the weak sense must be at least as large as half the number of cut edges. As cut edges contribute by one to the sum of degrees of two communities, this definition entails that for the network as a whole the number of inner edges is larger than the number of cut edges. Recently, several extensions of the definition of the *community in the strong and weak sense* have been proposed [13]. In these extensions one considers all outgoing edges from the set S to another subset S_t of E instead of the sum for all such sets: $k_i^{in}(S) > \max_{t=1,\ldots,M | S_t \neq S} \sum_{v_j \in S_t} a_{i,j}$, $\forall v_i \in S$, where M is the number of communities. The second extension is similar but uses sums of degrees instead of individual degrees: $\sum_{v_i \in S} k_i^{in}(S) > \max_{t=1,\ldots,M | S_t \neq S} \sum_{v_i \in S} \sum_{v_j \in S_t} a_{i,j}$.

One may also consider the difference for each community of the sum of indegrees and the sum of outdegrees. Then summing these contributions for all communities gives a k-way cut problem. Another approach is to normalize the contribution of each community by dividing it by its number of vertices [12]. The resulting function, to be maximized, is called *modularity density*. Alternatively, contributions of communities may be divided by their number of edges [15]. Finally, one may consider maximizing, in a divisive hierarchical method, the minimum ratio of the number of edges in a community divided by the number of cut edges [24].

3 Algorithm SC

We next present the rules of an algorithm, called SC (*Strong Communities*), for enumerating all the partitions in the strong sense of a given network $G = (V, E)$. Note that this problem always has a solution, i.e., the trivial partition consisting in a single community containing all the vertices. The algorithm will make use of two types of labels associated with the vertices and the edges of G respectively: label l_i associated with vertex v_i, $i = 1, \ldots, n$ (initially $l_i = i$ for all vertices, and at the current iteration the label of the vertex v_i is equal to the smallest label of a vertex of the community to which v_i belongs); the label $t_{i,j}$ associated with edge (v_i, v_j) can take three values $(-1, 0, 1)$. It is equal to -1 if it has already been decided that the vertices v_i and v_j belong to different communities; it is equal to 1 if it has already been decided that vertices v_i and v_j belong to the same community. If no decision has been taken, $t_{i,j} = 0$.

It turns out that the rules of our algorithm are simple and effective. To illustrate, we next list these rules and show at the same time that all partitions in the strong sense of the Zachary karate club network [22] can be found by hand.

Rules of the algorithm SC are the following:

- Rule 1 (*pending edges*): if the edge (v_i, v_j) is a pending one, set its label $t_{i,j}$ to 1 and set both l_i and l_j to $\min(l_i, l_j)$. In words, both vertices of a pending edge must belong to the same community.
  - Example 1 (Karate club): this network has a single pending edge, i.e., $(1, 12)$, so we set $t_{1,12} = 1$ and $l_{12} = 1$.
- Rule 2 (*degree two vertices*): if vertex v_i has degree $k_i = 2$, and its neighbors are v_j and v_k, set $t_{i,j} = 1$, $t_{i,k} = 1$ and $l_i = l_j = l_k = \min(l_i, l_j, l_k)$. In words, if a vertex v_i has degree 2 and neighbors v_j and v_k, it follows from the strong condition that all three vertices v_i, v_j, v_k must belong to the same community.
  - Example 1 (continued): this network has eleven degree two vertices; applying Rule 2 we get the following labels (see Fig. 1):
    - $t_{3,10} = t_{10,34} = 1$; $l_3 = l_{10} = l_{34} = 3$;
    - $t_{1,13} = t_{4,13} = 1$; $l_1 = l_4 = l_{13} = 1$;
    - $t_{15,33} = t_{15,34} = 1$; $l_{15} = l_{33} = l_{34} = 3$;
    - $t_{16,33} = t_{16,34} = 1$; $l_{16} = l_{33} = l_{34} = 3$;
    - $t_{6,17} = t_{7,17} = 1$; $l_6 = l_7 = l_{17} = 6$;
    - $t_{1,18} = t_{2,18} = 1$; $l_1 = l_2 = l_{18} = 1$;
    - $t_{19,33} = t_{19,34} = 1$; $l_{19} = l_{33} = l_{34} = 3$;
    - $t_{21,33} = t_{21,34} = 1$; $l_{21} = l_{33} = l_{34} = 3$;
    - $t_{1,22} = t_{2,22} = 1$; $l_1 = l_2 = l_{22} = 1$;
    - $t_{23,33} = t_{23,34} = 1$; $l_{23} = l_{33} = l_{34} = 3$;
    - $t_{27,30} = t_{27,34} = 1$; $l_{27} = l_{30} = l_{34} = 3$.

    Note that the Rules 1 and 2 should be applied only once, at the beginning of the resolution.
- Rule 3.a (*positive transitivity*): if $l_i = l_j$ and $t_{i,j} = 0$, set $t_{i,j} = 1$. In words, if two vertices v_i and v_j belong to the same community, and are joined by an edge which does not specify that, set the label of this edge as positive.
  - Example 1 (continued): after application of Rule 3.a we get the following edge labels:
    - $t_{1,2} = 1$ due to $t_{1,22} = t_{2,22} = 1$;
    - $t_{1,4} = 1$ due to $t_{1,13} = t_{4,13} = 1$;
    - $t_{6,7} = 1$ due to $t_{6,17} = t_{7,17} = 1$;
    - $t_{3,33} = 1$ due to $t_{3,10} = t_{10,34} = t_{19,34} = t_{19,33} = 1$;
    - $t_{33,34} = 1$ due to $t_{19,34} = t_{19,33} = 1$;
    - $t_{30,34} = 1$ due to $t_{27,30} = t_{27,34} = 1$;
    - $t_{30,33} = 1$ due to $t_{33,34} = t_{30,34} = 1$.

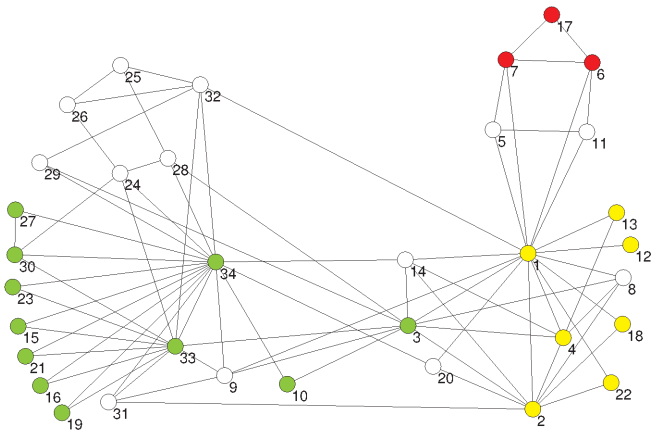


Figure 1: Zackary karate club communities after application of Rule 2; empty vertices are isolated at the current stage of resolution (color online).

- Rule 3.b (*negative transitivity*): if $l_i \neq l_j$ and $t_{i,j} = -1$, set $t_{a,b} = -1 \forall (v_a, v_b) : l_a = l_i, l_b = l_j$, and $t_{a,b} = 0$. In words, if two vertices belong to different communities and are joined by a negative edge, all the edges with label 0 joining two vertices of these communities are set to -1.
  - Example 1 (continued): this Rule does not apply yet because no edges have a negative label.
- Rule 4.a (*majority 1*): if the majority of neighbors of the vertex v_i belong to the same community and the label of the vertices belonging to this community is l, set $l_i = l = \min(l, l_i)$ and apply the *positive transitivity* Rule. In words, if half or more of the neighbors of v_i have the same label l, the only way to satisfy the strict inequality of the strong condition is to add the vertex v_i to the community where its vertices have label l.
  - Example 1 (continued): after application of Rule 4.a we get the following labels (see Fig. 2):
    - $l_8 = 1$, $t_{1,8} = t_{2,8} = t_{4,8} = 1$ due to $l_1 = l_2 = l_4 = 1$ (3 out of 4 labels of vertex v_8);
    - $l_{14} = 1$, $t_{1,14} = t_{2,14} = t_{4,14} = 1$ due to $l_1 = l_2 = l_4 = 1$ (3 out of 4 labels of vertex v_{14});
    - $l_{20} = 1$, $t_{1,20} = t_{2,20} = 1$ due to $l_1 = l_2 = 1$ (2 out of 3 labels of vertex v_{20});
    - $l_9 = 3$, $t_{3,9} = t_{9,33} = t_{9,34} = 1$ due to $l_3 = l_{33} = l_{34} = 3$ (3 out of 5 labels of vertex v_9);
    - $l_{31} = 3$, $t_{9,31} = t_{31,33} = t_{31,34} = 1$ due to $l_9 = l_{33} = l_{34} = 3$ (3 out of 4 labels of vertex v_{31});
    - $l_{24} = 3$, $t_{24,30} = t_{24,33} = t_{24,34} = 1$ due to $l_{30} = l_{33} = l_{34} = 3$ (3 out of 5 labels of vertex v_{24});
    - $l_{28} = 3$, $t_{3,28} = t_{24,28} = t_{28,34} = 1$ due to $l_3 = l_{24} = l_{34} = 3$ (3 out of 4 labels of vertex v_{28});
    - $l_{29} = 3$, $t_{3,29} = t_{29,34} = 1$ due to $l_3 = l_{34} = 3$ (2 out of 3 labels of vertex v_{29});
    - $l_{32} = 3$, $t_{29,32} = t_{29,33} = t_{29,34} = 1$ due to $l_{29} = l_{33} = l_{34} = 3$ (3 out of 6 labels of vertex v_{32});
    - $l_{25} = 3$, $t_{25,38} = t_{25,32} = 1$ due to $l_{28} = l_{32} = 3$ (2 out of 3 labels of vertex v_{25});
    - $l_{26} = 3$, $t_{24,26} = t_{25,26} = t_{26,32} = 1$ due to $l_{24} = l_{25} = l_{32} = 3$ (3 out of 3 labels of vertex v_{26}).
- Rule 4.b (*majority 2*): if a vertex v_i has degree $k_i = 2d$, and there are d neighbors with label l_1 and d neighbors with label l_2, set $l_i = \min(l_1, l_2)$, and for all the vertices v_k with $l_k = l_1$ or $l_k = l_2$, set $l_k = l_i$. Then, apply the *positive transitivity* Rule. In other words, we merge the communities with labels l_1 and l_2, and we put in this new community the vertex v_i, too.
  - Example 1 (continued): after application of Rule 4.b we get the following labels (see Fig. 3):
    - v_3 has 10 neighbors, 5 of which have label 1, and the 5 others have label 3. So, to have coherent labels, these two communities must be merged. All labels of vertices are equal to 1, except for v_5, v_6, v_7, v_{11}, v_{17}. All labels of edges are equal to 1 except for $t_{1,5}$, $t_{1,6}$, $t_{1,7}$, $t_{1,11}$, $t_{5,6}$, $t_{5,11}$, $t_{6,11}$ which are equal to 0.

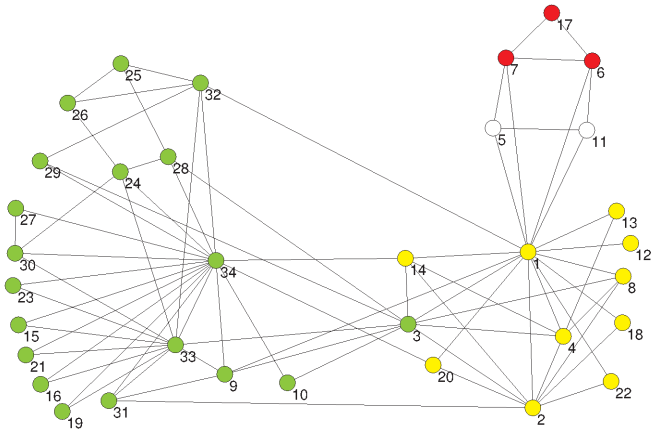


Figure 2: Zackary karate club communities after application of Rule 4.a (color online).

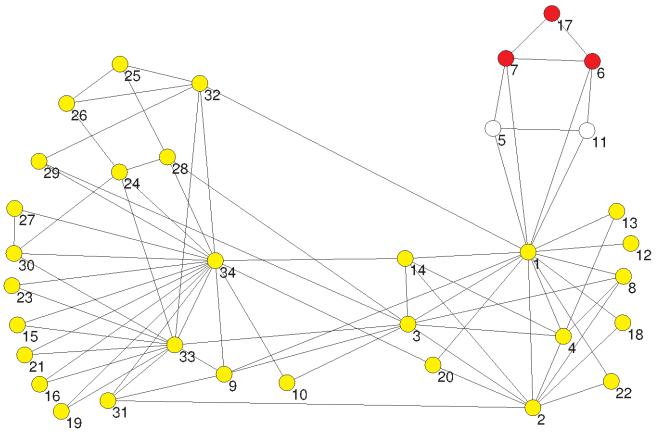


Figure 3: Zackary karate club communities after Rule 4.b (color online).

- Rule 4.c (*majority 3*): if the number of negative edges, i.e., edges labeled with -1, incident with the vertex v_i is equal to $\lceil \frac{k_i}{2} \rceil - 1$, for all the neighbor vertices v_j of v_i having $t_{i,j} = 0$, set $t_{i,j} = 1$, and for all the vertices v_k with $l_k = l_j$, set $l_k = l_i$. In words, when the number of negative edges incident to v_i is almost the majority there is only one way to satisfy the strong condition at vertex v_i, i.e., set the label associated to all other incident edges to 1.
  - Example 1 (continued): this Rule does not apply yet because no edges have a negative label.

  Rules 3 and 4 must be repeated as long as there is at least one change of label.
- Rule 5 (*branching*): if no more labels of edges can be modified according to the previous rules, one selects an edge with label 0 and, subject to that condition, joining the two largest communities. Set the label of this edge to -1 (left branch). Then set separately this label to 1 (right branch), store the current subproblem and return to Rule 3.a.
- Rule 6.a (*no majority*): if the number of negative edges incident with the vertex v_i is larger than $\lceil \frac{k_i}{2} \rceil - 1$, apply Rule 8 below.
- Rule 6.b (*no coherent labels*): if there exists two vertices v_i and v_j with $l_i = l_j$ and $t_{i,j} = -1$, apply Rule 8 below.
- Rule 7 (*feasible solution*): if all edges have a label -1 or 1, store the corresponding partition, then apply Rule 8.

- Rule 8 (*backtracking*): return to the latest application of the branching rule and consider the right hand-side branch as current subproblem.
  - Example 1 (continued): the branching is done on edge $(1, 6)$ which first gets a label equal to -1, and later to 1.
    - $t_{1,6} = -1$. For the *negative transitivity* also $t_{1,7} = -1$. For the *majority 3* Rule applied to vertex v_6 we can set $t_{6,11} = 1$ and $l_{11} = 6$. Again, for the *negative transitivity* $t_{1,11} = -1$. Then, for the *majority 1* Rule applied to vertex v_5, $l_6 = l_7 = l_{11} = l_{17} = 5$ and $t_{5,7} = t_{5,11} = 1$ and at the end for the *negative transitivity* $t_{1,5} = -1$. Since all the labels are now fixed, we apply Rule 7, and save the solution, whose labels are: $l_5 = l_6 = l_7 = l_{11} = l_{17} = 5$, and all the other vertices have label 1; $t_{1,5} = t_{1,6} = t_{1,11} = -1$ and all the other edges have label 1. The obtained partition is presented in Fig. 4. After storing this partition we apply Rule 8, and we continue with the right branch below.
    - $t_{1,6} = 1$. For the *positive transitivity* also $t_{1,7} = 1$ and $l_6 = l_7 = l_{17} = 1$. For the *majority 1* Rule applied to vertex v_5 we can set $l_5 = 1$ and $t_{5,7} = t_{1,5} = 1$. Then, we apply again the *majority 1* Rule to vertex v_{11} and we can set $l_{11} = 1$, $t_{1,11} = t_{5,11} = t_{6,11} = 1$. Since all the labels are now fixed, we apply Rule 7, and save the solution, where all the vertices have label 1 and all the edges have label 1. After storing this partition we apply Rule 8, and the algorithm stops.

  So, only two partitions have been found, including the trivial partition with a single community. The other one isolates a small and dense community connected to the remaining part of the network by a cut vertex, i.e. v_1 whose removal disconnects the network. This partition is far from that one observed by Zachary in which there were two clusters of equal size. Note however that the small cluster is contained in one of the clusters of Zachary's partition. A similar result was obtained by Medus and Dorso in [15], Fig. 1, when optimizing their strong figure of merit subject to the constraint that communities are strong ones.

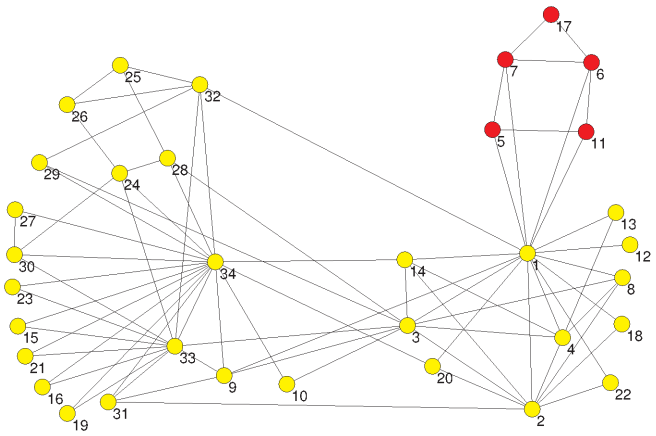


Figure 4: Solution with two strong communities found by algorithm SC for Zachary karate club network (color online).

4 A variant of the algorithm SC: ASC

From the discussion of the real-word example of community detection in the strong sense presented in the previous section it appears that: (i) results are often satisfactory in terms of individual communities, as the strong conditions are not easily satisfied; (ii) communities are however often too large and correspond in fact to the union of several distinct communities; (iii) in the limit one finds a single community regrouping all vertices, which of course is not informative; (iv) the main reason for having large heterogeneous communities

appears to be due to the degree two vertices. Indeed, the strong condition imposes that them and both of their two neighbors belong to the same community, while these neighbors can be very different in terms of their own neighbors.

This suggests to weaken the strong condition when it appears to be unjustified, i.e., for the degree two vertices. We call a community *almost-strong* if it satisfies the condition $k_i^{in} \geq k_i^{out}$ for all vertices of degree two, and the strong condition $k_i^{in} > k_i^{out}$ for the remaining vertices.

Modifications to be brought to the algorithm of the previous section in order for it to find partitions in the almost-strong sense are minimum. They lead to a modified algorithm called ASC (*Almost-Strong Communities*). Indeed, it sufficies to remove Rule 2 and update Rules 4.a, 4.b, 4.c and 6.a by specifying that in the case of a vertex v_i of degree 2, one uses majority and not strict majority.

Rules of the modified algorithm ASC are the following:

- Rule 1 (*pending edges*): if the edge (v_i, v_j) is a pending one, set its label $t_{i,j}$ to 1 and set both l_i and l_j to $\min(l_i, l_j)$. In words, both vertices of pending edge must belong to the same community.
- Rule 3.a (*positive transitivity*): if $l_i = l_j$ and $t_{i,j} = 0$, set $t_{i,j} = 1$. In words, if two vertices v_i and v_j belong to the same community, and are joined by an edge which does not specify that, set the label of this edge as positive.
- Rule 3.b (*negative transitivity*): if $l_i \neq l_j$ and $t_{i,j} = -1$, set $t_{a,b} = -1 \, \forall (v_a, v_b) : l_a = l_i, l_b = l_j$, and $t_{a,b} = 0$. In words, if two vertices belong to different communities and are joined by a negative edge, all the edges with label 0 joining two vertices of these communities are set to -1.
- Rule 4.a.1 (*majority 1*): if the majority of neighbors of the vertex v_i with $k_i \neq 2$ belong to the same community and the label of the vertices belonging to this community is l, set $l_i = l = \min(l, l_1)$ and apply the *positive transitivity* Rule.
- Rule 4.a.2 (*majority 1'*): if both neighbors of the vertex v_i with $k_i = 2$ belong to the same community and the label of the vertices belonging to this community is l, set $l_i = l = \min(l, l_1)$ and apply the *positive transitivity* Rule.
- Rule 4.b (*majority 2*): if a vertex v_i has degree $k_i = 2d \neq 2$, and there are d neighbors with label l_1 and d neighbors with label l_2, set $l_i = \min(l_1, l_2)$, and for all the vertices v_k with $l_k = l_1$ or l_2, set $l_k = l_i$. Then, apply the *positive transitivity* Rule. In other words, merge the communities with labels l_1 and l_2, and put in this new community the vertex v_i, too.
- Rule 4.c.1 (*majority 3*): if the number of negative edges incident with the vertex v_i with degree $k_i \neq 2$ is equal to $\lceil \frac{k_i}{2} \rceil - 1$, for all the neighbor vertices v_j of v_i having $t_{i,j} = 0$, set $t_{i,j} = 1$, and for all the vertices v_k with $l_k = l_j$, set $l_k = l_i$.
- Rule 4.c.2 (*majority 3'*): if the number of negative edges incident with the vertex v_i with degree $k_i = 2$ is equal to 1, for all the neighbor vertices v_j of v_i having $t_{i,j} = 0$, set $t_{i,j} = 1$, and for all the vertices v_k with $l_k = l_j$, set $l_k = l_i$.
  Rules 3 and 4 must be repeated as long as there is at least one change of label.
- Rule 5 (*branching*): if no more labels of edges can be modified according to the previous rules, one selects an edge with label 0 and subject to that condition joining the two largest communities. Set the label of this edge to -1 (left branch). Then set separately this label to 1 (right branch), store the current subproblem and return to Rule 3.a.
- Rule 6.a.1 (*no majority*): if the number of negative edges incident with a vertex v_i with degree $k_i \neq 2$ is larger than $\lceil \frac{k_i}{2} \rceil - 1$, apply Rule 8 below.
- Rule 6.a.2 (*no majority'*): if the number of negative edges incident with a vertex v_i with degree $k_i = 2$ is larger than 1, apply Rule 8 below.
- Rule 6.b (*no coherent labels*): if there exists two vertices v_i and v_j with $l_i = l_j$ and $t_{i,j} = -1$, apply Rule 8 below.
- Rule 7 (*feasible solution*): if all edges have a label -1 or 1, store the corresponding partition, then apply Rule 8.

- Rule 8 (*backtracking*): return to the latest application of the branching rule and consider the right hand-side branch as current subproblem.

5 Results

In this section we compare the results obtained for finding partitions in the strong and the almost-strong sense. As partitions in the almost-strong case may be very numerous, we only store those which have the largest number of communities. Further discriminations between them could be obtained according to the values they give to some objective function. This topic will not be discussed here.

Let us first consider again Zachary's karate club. The largest number of communities in almost-strong partition is three. This partition is unique and it is represented in Fig. 5. There are also 22 partitions into two clusters, including one where the yellow and green clusters communities are joined into a single one. This partition coincides with the partition between followers of the karate club administrator and the instructor in the split observed by Zachary except for member 10 (which is often misclassified). Another partition into two clusters is of course the strong partition of Fig. 4 and finally there is also the trivial partition.

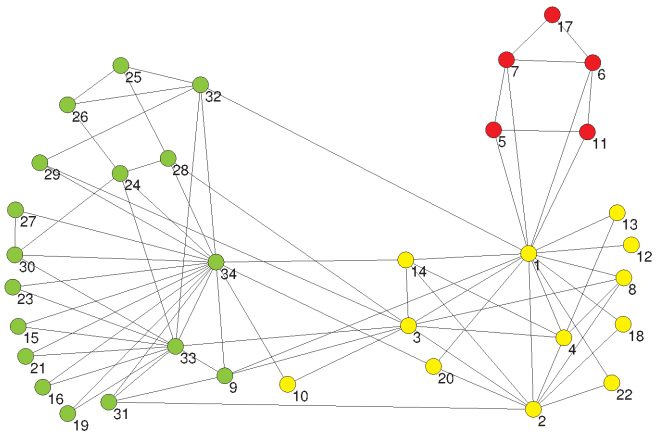
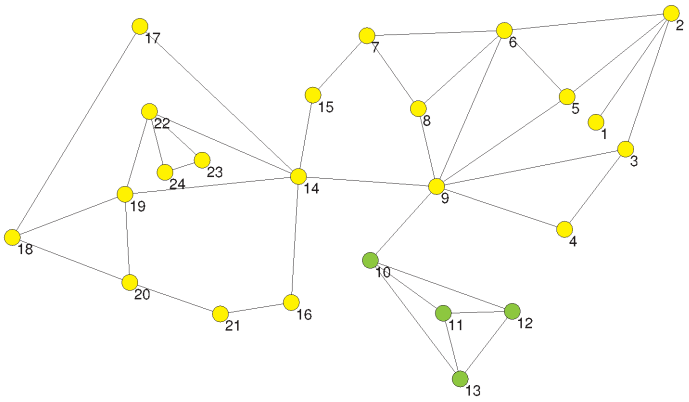
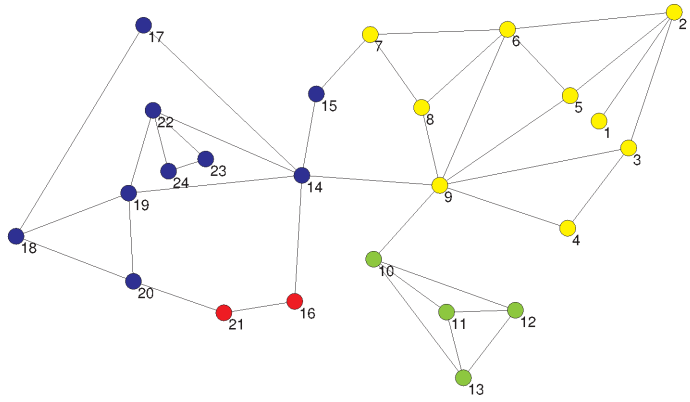


Figure 5: Partition into three almost-strong communities obtained by algorithm ASC for Zachary karate club network (color online).

A second real-world dataset concerns informal communications within a sawmill on strike [25]. Vertices are associated with the 24 employees of a wood processing facility where a new management team proposes changes to the compensation package. The workers refuse and a strike ensues. Facing a stalemate, the management asks a consultant to analyze the communications among the employees. Edges of the network correspond to frequent discussions on the strike between pairs of colleagues. Two partitions into strong communities were obtained with the SC algorithm; one of them is a clique on 4 vertices $(10, 11, 12, 13)$ corresponding to all spanish-speaking employees, the remaining 20 employees are english-speaking; see Fig. 6(a). The ASC algorithm gives 20 partitions. A single one of them has 4 communities (see Fig. 6(b)), and none had more. The small community with 4 spanish-speaking employees remains the same. The second community of 20 employees is split into 3 communities: a first one corresponds to 9 english-speaking employees with age less than or equal to 30. The second community with 9 employees and the third one with 2 employees correspond to older english-speaking workers. The partition of the 24 employees in three communities, i.e., spanish-speaking, young english-speaking and older english speaking employees obtained by joining the two last communities corresponds exactly to the partition obtained by the consultant. As the strong conditions and the almost-strong conditions remain satisfied when communities are merged, the ASC algorithm did also find the optimal three community partition. Detection of the small community with employees 16 and 21 may be interpreted in that these employees are less talkative, or less concerned by the strike, than most of the others.



(a)



(b)

Figure 6: Partitions into strong communities and into almost-strong communities obtained by algorithms SC and ASC respectively for the strike dataset (color online).

It thus appears that algorithm SC recognises well a small, almost isolated community but groups unduly the others. Algorithm ASC finds the optimal partition and it perhaps provides a little more information.

A next example purports to graph theory. Mr. Cherowitzo compiled a glossary of graphs and digraphs.[1] The version of 2003, in Pajek format, contains 72 vertices associated with terms. An arc from v_i to v_j means that the concept associated with v_i is used in the definition of v_j. We neglected orientation of arcs and considered only the largest connected component, which has 60 vertices and 114 edges. Applying algorithm SC only the trivial partition was found. Turning to algorithm ASC, many partitions were obtained, 5 of which have the largest number of communities, i.e., 6. The most intuitively appealing of them is presented on Fig. 7.

We next comment on these communities going from the smallest to the largest. The first community corresponds to two terms, i.e., {complete, clique}. They are clearly close, as a complete graph is a clique. The second community also has two terms, pertained to computer searh, i.e., {child, ordered tree}. These two communities appeared unchanged in all 5 partitions into 6 communities. A third community contains 7 terms, i.e., {decision tree, binary search tree, m-ary tree, rooted tree, offspring, level, height}. All those terms correspond, as did those of community 2, to computer search. Community three is similar in the 4 other partitions into 6 communities except for that the term decision tree is assigned to another community. A fourth community contains 8 terms, i.e., {diameter, distance, hamiltonian, walk, trail, path, aciclyc graph, cycle}.

[1]The glossary can be found at `http://vlado.fmf.uni-lj.si/pub/networks/data/DIC/TG/glossTG.pdf`.

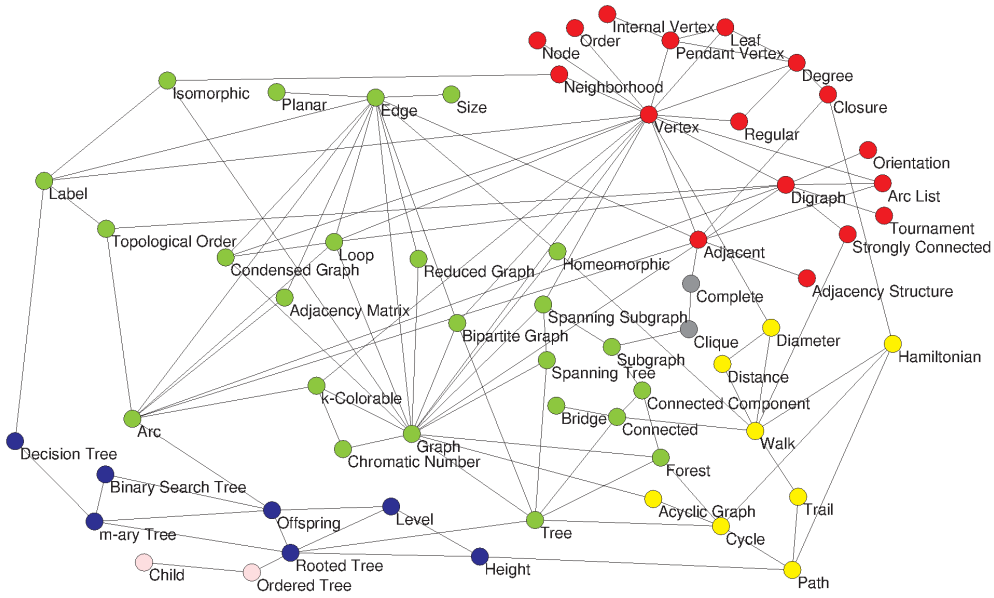


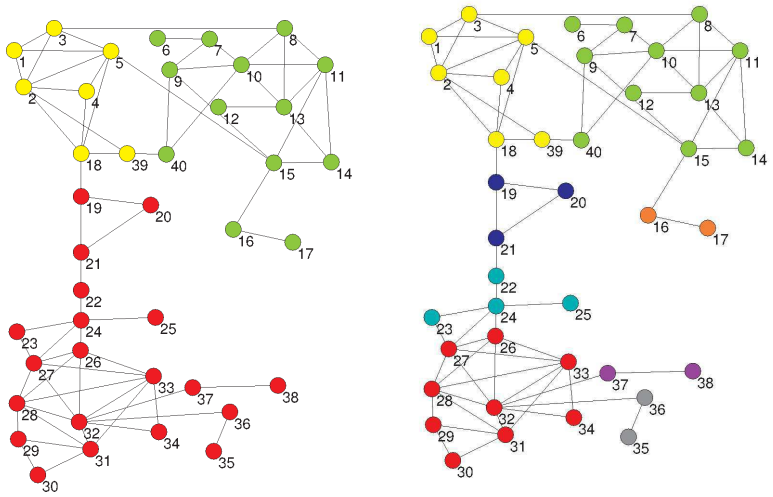
Figure 7: Partition into almost-strong communities obtained by algorithm ASC for the graph and digraph glossary dataset (color online).

These terms correspond to concepts related to paths and cycles. A fifth community contains 17 terms, i.e., {strongly connected, tournament, digraph, orientation, arc list, neighborhood, node, order, internal vertex, vertex, pendant vertex, leaf, degree, regular, adjacency structure, adjacent, closure}. It seems difficult to find a concept encompassing all of these terms. The five first, i.e., {strongly connected, tournament, digraph, orientation, arc list} correspond to oriented graphs. The remainder corresponds to vertices and adjacency. Note that this community contains several pairs of synonims, i.e., {node} and {vertex}, and {pendant vertex} and {leaf}. The sixth community contains 24 terms, i.e., {label, isomorphic, planar, edge, size, topological order, adjacency matrix, loop, reduced graph, condensed graph, homeomorphic, bipartite graph, spanning subgraph, subgraph, spanning tree, connected component, bridge, connected, forest, tree, graph, cromatic number, k-colorable, arc}. This community appears to be less homogeneous than the others. Some concepts are related to edges, i.e., {edge, loop, size, label}. Others correspond to properties or families of graphs: {isomorphic, homeomorphic, condensed graph, reduced graph, bipartite graph, spanning subgraph, spanning tree, subgraph, connected component, connected, bridge, tree, k-colorable, cromatic number, graph}. Although this partition appears to be quite informative, it is not perfect, e.g. because {forest} and {acyclic graph} are synonyms but attributed to different communities. Or yet close terms such as {adjacency matrix} and {adjacency structure} are also attributed to different communities.

A fourth example comes from the well-known paper on dolphins due to Lusseau et al. [27]. However, it does not concern the set of all 62 dolphins, but another network giving the sociogram of the community for groups followed between 1995 and 2001. This network has 40 vertices and 70 edges. When trying to find communities in the strong sense four partitions were obtained, one with three communities, represented on Fig. 8(a), two obtained by merging pairs of adjacent communities, and the trivial partition.

Finding communities in the almost-strong sense gives a partition in eight communities, which is represented in Fig. 8(b). It refines one of the communities by isolating a small two vertices community with one vertex of degree two. It also refines more drastically the largest community by isolating four subnetworks with two, two, three and four entities. Each of these subnetworks contains a vertex of degree two. It appears clearly that this almost-strong partition is more informative than any other strong one.

We next studied the full set of 62 dolphins, for which a real-world partition into two communities has been observed by Lusseau et al. [27]. Algorithm SC found four strong partitions, all of which have two communities, plus the trivial partition. None of these bipartitions is close to the optimal one, observed by



(a) (b)

Figure 8: Partitions into strong communities and into almost-strong communities obtained by algorithms SC and ASC respectively for the small dolphin dataset (color online).

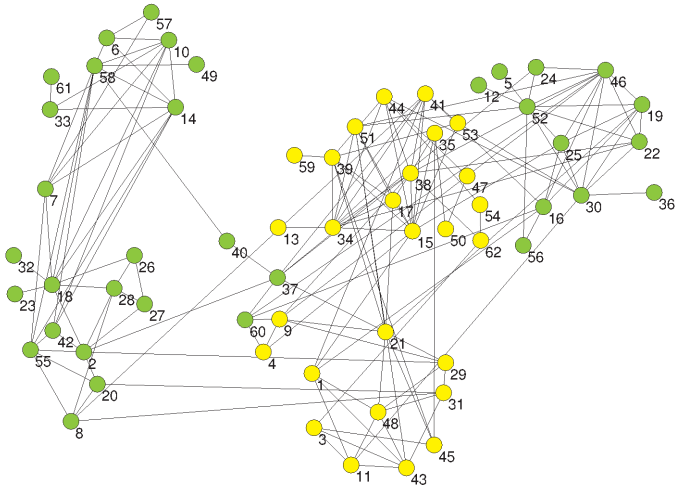
Lusseau. The closest bipartition is presented in Fig. 9(a) and has 14 misclassified dolphins. Turning to the ASC algorithm it was found that it became quite time consuming. A partial resolution gave 7 partitions into 4 communities. One of them (see Fig. 9(b)) has a community which is the same as the 21 dolphins community of Lusseau, and consequently the union of the other three communities is equal to the 41 dolphins community of Lusseau.[2] We also use a mixed-integer program [29] with the almost-strong condition on communities and as objective to find a partition in as many communities as possible. This gives us a partition very close to that one obtained by partial resolution with ASC. It has 4 communities which are equal to those obtained previously, except for the fact that dolphins 16 and 56 now belong to a community adjacent to their own (yellow instead of green color online).

Our last dataset is the set of characters of Victor Hugo's *Les Misérables* [30, 31]. 77 vertices correspond to these characters and 254 edges indicate direct interaction i.e., an edge joins vertices v_i and v_j if and only if the corresponding characters appear jointly in at least one of the many, usually short, chapters of the novel. Algorithm SC gives a partition into 4 strong communities, and it is represented in Fig. 10(a). Algorithm ASC was too time and space consuming and did not give useful results for this dataset. So we used again the mixed-integer program mentioned above. We then obtained an almost-strong partition with 7 communities presented in Fig. 10(b).

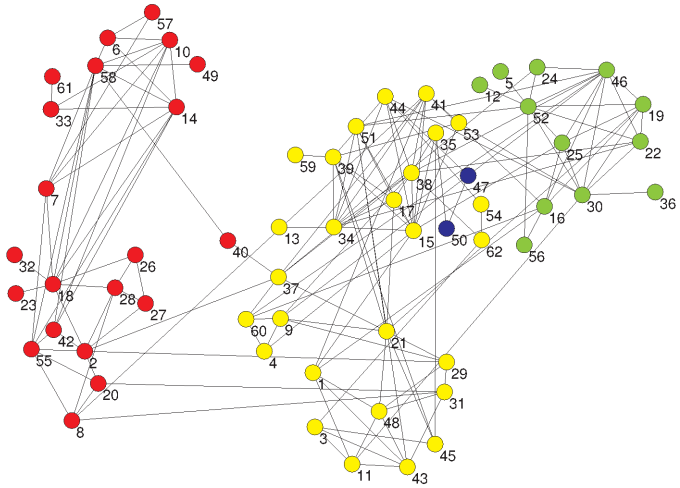
Comparing results of SC and ASC we observe that the partition of ASC is clearly more discriminant than that of SC. The largest community is broken up into 2 communities and slightly mixed with the second largest community. This second largest community is broken up into 3 communities. One of the remaining communities is unchanged, and the second one slightly modified by moving vertices 3 and 4 to the largest community. This is arbitrary as in both cases the ASC condition is satisfied.

Moreover, in absence of a generally accepted substantively justified partition for *les misérables*, we compare both partitions obtained by SC and the mixed-integer version of ASC, with the partition obtained by the edge ratio (ER) algorithm [24], which has an excellent intuitive interpretation. The ER partition contains 10 communities, represented in Fig. 11. They can be described as follows:

[2]Medus and Dorso present in [15], Fig. 4, a partitioning of the 62 dolphin dataset into five weak communities. The union of two of them and of the three others gives a bipartition of the set of dolphins into communities in the strong sense. It appears however that the dataset used by Medus and Dorso differs from the standard one of Lusseau (i.e., there are 157 edges instead of 159 and the number of vertices of equal degree differs in several places). A. Medus kindly applied his algorithm for strenght maximization subject to Radicchi strong condition to Lusseau standard network and obtained the same partition as we do.



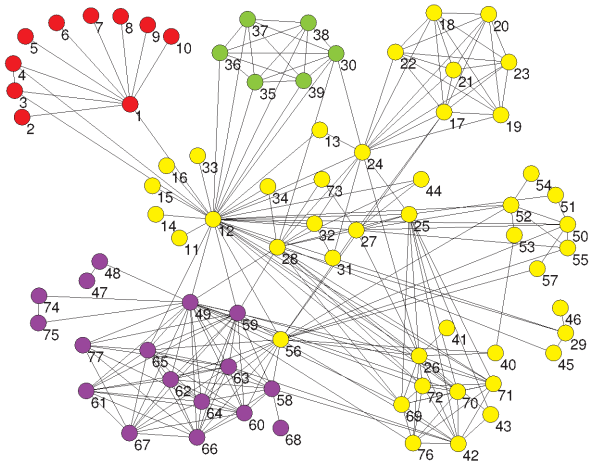
(a)



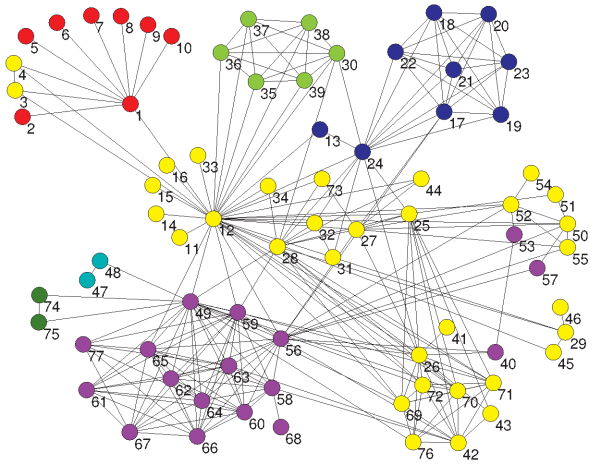
(b)

Figure 9: Partitions into strong communities and into almost-strong communities obtained by algorithms SC and ASC respectively for the dolphin dataset (color online).

- C_1: two young children; $C_1 = \{74, 75\}$.
- C_2: Gavroche, Marius and the revolutionaries, members of the friends of the ABC association; $C_2 = \{49, 56, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 77\}$.
- C_3: the evil innkeeper Thénardier, his family and accomplices; $C_3 = \{26, 40, 41, 42, 43, 69, 70, 71, 72, 76\}$.
- C_4: an alias of Thénardier and the wife of his accomplice Brujon; $C_4 = \{47, 48\}$.
- C_5: Bishop Myriel and the characters he met during his long life; $C_5 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$.
- C_6: protagonists of the Champmathieu affair; $C_6 = \{30, 35, 36, 37, 38, 39\}$.
- C_7: students and their *grisettes* (except Fantine who pertains to the main plot) whom they loved and abandoned; $C_7 = \{17, 18, 19, 20, 21, 22, 23\}$.
- C_8: Fauchelevent, saved from an accident by Jean Valjean, who provides sanctuary at the convent of which Mother Innocent is priores, and gardener Gribier also living there; $C_8 = \{29, 45, 46\}$.



(a)



(b)

Figure 10: Partitions into strong communities and into almost-strong communities obtained by algorithms SC and ASC respectively for *Les misérables* dataset (color online).

- C_9: the family of Marius;
  $C_9 = \{50, 51, 52, 53, 54, 55, 57\}$.
- C_{10}: the central plot with the main hero Jean Valjean (12), his nemesis inspector Javert (28), as well as Fantine (24) and Cosette (27); $C_{10} = \{11, 12, 13, 14, 15, 16, 24, 25, 27, 28, 31, 32, 33, 34, 44, 73\}$.

As can be seen on Fig. 10, algorithm SC finds exactly the two communities C_5 and C_6. However, it groups several other communities. Indeed, it groups the community C_2 but omitting Marius (vertex 56), together with the two small communities C_1 and C_4. These two mergings are less damaging than omitting Marius as the four protagonists of these two communities are minor characters connected only to Gavroche (vertex 49) in community C_2. All other characters are joined into a big communities of 44 characters. So once again algorithm SC determines exactly some of the communities, but gives a fuzzy description of another one, and merges all the remaining ones.

Comparing now the partitions obtained for ASC and ER we find that they are much closer than those of SC and ASC, as well as of those of SC and ER. Indeed, the ASC partition contains without errors 3 of the communities of the ER partition, i.e., C_1, C_4 and C_6. Three other communities, i.e., C_2, C_5 and C_7,

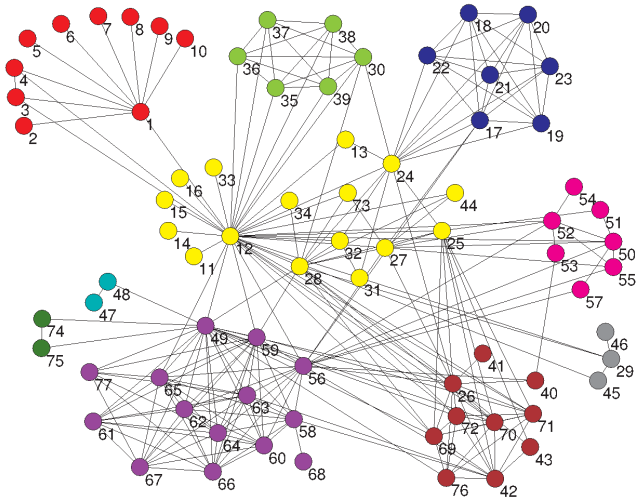


Figure 11: Partition obtained by algorithm ER for *Les misérables* dataset (color online).

are slightly different in the two partitions. As mentioned above, the sister and the maid of Bishop Myriel (associated respectively to vertices 4 and 3) can be arbitrarily assigned to community C_5 or C_{10}. Community C_7 is larger in the ASC partition than in the ER one, Fantine (vertex 24) and Marguerite (vertex 13) being added. Finally, the ASC partition groups the communities C_3, C_8 and C_9 with the remaining part of the main plot community, except for vertices 40, 53 and 57 which belong to community C_2. It thus appears that the ASC criterion recognises precisely several communities but may still unduly merge some of them.

6 Conclusions

Several conditions which should be satisfied by communities in a network have been proposed by various authors. In this paper we studied one of the most intuitively appealing, i.e., the condition to have a community in the strong sense proposed by Radicchi et al. [20]. It states that to be a community in the strong sense, a subset $S \subseteq V$ of vertices should be such that the indegree must be larger than the outdegree for each vertex $v_i \in S$.

A too stringent condition will not have a high degree of resolution, i.e., it may have a tendency to regroup several natural communities into large ones. Conversely, a too lax condition can generate a large number of small and not necessary relevant communities. We have investigated the question of whether the definition of strong community is too stringent or too lax. To that effect we have proposed an exact enumeration algorithm which guarantees finding all partitions of a network into strong communities. This algorithm is quite efficient, as illustrated by the fact that it could be applied by hand to the well-known Zachary karate club network. Actually, it has solved problems with up to 77 vertices. The results are the following: while some not too large communities are found in a crisp way, it is not uncommon for several network communities to be joined in larger, heterogeneous ones. In the limit, one can obtain only the trivial partition in which all vertices belong to the same community. The reason for the formation of too large communities appears to be the presence of degree two vertices. Indeed, when a vertex has degree two, the strong condition imposes that itself and both of its neighbors belong to the same community.

To test this simple hypothesis, we have modified the definition of community in the strong sense in a minimal but sufficient way. In the definition of community in the strong sense we modify the strict inequality between indegree and outdegree to a non-strict inequality for all vertices of degree two and only them. We say that this new condition defines a community in the almost-strong sense. Then we have modified, slightly but sufficiently, the algorithm SC to conform to this new condition. Indeed, the resulting algorithm ASC

usually gives partitions into more communities than algorithm SC. Moreover, they are usually more intuitively appealing.

Combining a criterion for community evaluation with constraints on each community has been pioneered by Medus and Dorso [15]. This work can be pursued in two ways: (i) modifying the models and the corresponding algorithms or heuristics to incorporate the constraints on the communities from the outset, or (ii) using a standard algorithm or heuristic then checking if the obtained communities satisfy the given constraints; if not, apply some specific heuristic to restore feasibility. Those two approaches are the topic of a paper in preparation.

References

[1] M. E. J. Newman, *Networks: An Introduction* (Oxford University Press, Oxford, 2010).

[2] S. Wasserman and K. Faust, *Social Network Analysis. Methods and Applications.* (Cambridge University Pres, New York, USA, 1994).

[3] C. Berge, *Graphs and Hypergraphs* (Elsevier Science Ltd, 1985).

[4] J. Gross and J. Yellen, *Handbook of graph theory*, edited by J. Gross and J. Yellen, Discrete mathematics and its applications (CRC Press, 2004).

[5] X. S. Zhang, R. S. Wang, Y. Wang, J. Wang, Y. Qiu, L. Wang, and L. Chen, Europhysics Letters **87**, 38002 (2009).

[6] J. G. Wang, L. Wang, Y. Q. Qiu, Y. Wang, and X. S. Zhang, in *Proceedings of the 3rd International Symposium on Optimization and Systems Biology*, Lecture Notes in Operations Research 11 (2009) pp. 142–150.

[7] M. Girvan and M. E. J. Newman, Proceedings of the National Academy of Sciences of the U.S.A. **99**, 7821 (2002).

[8] P. Chan, M. Schlag, and J. Zien, IEEE Trans. on CAD of Integrated Circuits and Systems **13**, 1088 (1994).

[9] L. W. Hagen and A. B. Kahng, IEEE Trans. on CAD of Integrated Circuits and Systems **11**, 1074 (1992).

[10] J. Shi and J. Malik, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 888 (2000).

[11] M. Bolla, Phys. Rev. E **84**, 016108 (2011).

[12] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, Phys. Rev. E **77**, 036109 (2008).

[13] Y. Hu, H. Chen, P. Zhang, M. Li, Z. Di, and Y. Fan, Phys. Rev. E **78**, 026121 (2008).

[14] C. O. Dorso and A. D. Medus, International Journal of Bifurcation and Chaos **20**, 361 (2010).

[15] A. D. Medus and C. O. Dorso, Phys. Rev. E **79**, 066111 (2009).

[16] M. Rosvall and C. T. Bergstrom, Proceedings of the National Academy of Sciences of the U.S.A. **104**, 7327 (2007).

[17] B. Ball, B. Karrer, and M. E. J. Newman, Phys. Rev. E **84**, 036103 (2011).

[18] V. A. Traag, P. Van Dooren, and Y. Nesterov, Phys. Rev. E **84**, 016114 (2011).

[19] F. Luccio and M. Sami, IEEE Transactions on Circuit Theory **16**, 184 (1969).

[20] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, Proceedings of the National Academy of Sciences of the U.S.A. **101**, 2658 (2004).

[21] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti, Phys. Rev. E **82**, 046112 (2010).

[22] W. W. Zachary, Journal of Anthropological Research **33** (1977).

[23] K. H. Shafique, *Partitioning a graph in alliances and its application to data clustering*, Ph.D. thesis, University of Central Florida Orlando, Florida (2004).

[24] S. Cafieri, P. Hansen, and L. Liberti, Phys. Rev. E **81**, 026105 (2010).

[25] J. H. Michael, Forest Products Journal **47**, 41 (1997).

[26] The glossary can be found at `http://vlado.fmf.uni-lj.si/pub/networks/data/DIC/TG/glossTG.pdf`.

[27] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, Behavioral Ecology and Sociobiology **54**, 396 (2003).

[28] Medus and Dorso present in [15], Fig. 4, a partitioning of the 62 dolphin dataset into five weak communities. The union of two of them and of the three others gives a bipartition of the set of dolphins into communities in the strong sense. It appears however that the dataset used by Medus and Dorso differs from the standard one of Lusseau (i.e., there are 157 edges instead of 159 and the number of vertices of equal degree differs in several places). A. Medus kindly applied his algorithm for strenght maximization subject to Radicchi strong condition to Lusseau standard network and obtained the same partition as we do.

[29] V. Chvátal, *Linear programming* (W.H. Freeman, 1983).

[30] V. Hugo, *Les Misérables* (Gallimard, Bibliothèque de la Pleiade, Paris, 1951).

[31] D. E. Knuth, *The Stanford GraphBase: a platform for combinatorial computing* (Addison-Wesley, 1993).