Math. Comput. Sci.
1 (2008), no. 3, 441-457

Community Structures of Networks

William Y. C. Chen[1], Andreas W. M. Dress[2] and Winking Q. Yu[3]

[1,3]Center for Combinatorics, LPMC
Nankai University, Tianjin 300071, P. R. China

[2]CAS-MPG Partner Institute for Computational Biology
Shanghai Institutes for Biological Sciences, Chinese Academy of Sciences
Shanghai 200031, P. R. China
Max-Planck-Institut fuer Mathematik in den Naturwissenschaften
Inselstrasse 22-26, D-04103 Leipzig, Germany

emails: [1]chen@nankai.edu.cn, [2]dress@sibs.ac.cn,
[3]yuqiang@mail.nankai.edu.cn

Abstract. We present an approach to studying the community structures of networks by using linear programming (LP). Starting with a network in terms of (a) a collection of nodes and (b) a collection of edges connecting some of these nodes, we use a new LP-based method for simultaneously (i) finding, at minimal cost, a second edge set by deleting existing and inserting additional edges so that the network becomes a disjoint union of cliques and (ii) appropriately calibrating the costs for doing so. We provide examples that suggest that, in practice, this approach provides a surprisingly good strategy for detecting community structures in given networks.

Keywords: Networks, graphs, community structures, clique partitioning problem, graph partitioning problem, linear programming, integer linear programming, food webs, Zachary's karate club

1 Introduction

In recent years, the study of large networks has attracted a lot of attention in the natural and the social sciences. In both areas, networks of all sorts play an important role, from the World-Wide Web to scientific-collaboration and citation networks to regulatory, protein, metabolic, and ecological networks. In particular, since "scale-free" and

"small-world" networks were proclaimed as constituting new and universally applicable paradigms of interaction schemes observed in real-world systems suggesting fundamentally new basic laws governing important processes addressed, network-oriented research intensified drastically (cf. [1] to [36]).

What we are concerned with here is the *community structure* of networks. Loosely speaking, given a network consisting of a collection V of *nodes* together with some information about the *degree of relatedness* between any two of its nodes, the term *community* is meant to refer to those subsets C of the node set V whose nodes are more closely related to one another than to the nodes outside C. The term *community structure* is meant to refer to a partition of V into a disjoint union of such subsets.

Starting with B.W. Kernighan and S. Lin's paper [23] from 1970, such subsets — and algorithms for computing them — were studied in quite a number of interesting papers and books in the last decades, see e.g. [15], [27], and [33].

However, this study took a decidedly new turn with the publication of M. Girvan and M. Newman's PNAS paper [16] in 2002. Their algorithm, later dubbed "GN algorithm" by F. Radicchi et al [28], begins with the entire original network, computes each edge's "betweenness", and proceeds by deleting one edge at a time according to its (continuously updated) "betweenness", starting with the (or, rather, an) edge of highest "betweennes". It works quite well for many networks. However, it does not unambiguously determine a community structure (i.e., a *partition* of the network's vertex set), but rather a *hierarchy*, and its complexity is relatively high, i.e., it is $O(m^2n)$ where m is the number of its edges and n is that of its vertices.

J.R. Tyler et al [31] introduced a faster variant of that algorithm using a Monte Carlo method to estimate the relevant parameters. F. Wu and B.A. Huberman [35] proposed an algorithm of complexity $O(n^3 \lg n)$ that is motivated by properties of resistor networks and avoids edge cutting. Next, F. Radicchi et al [28] introduced "weak" and "strong" communities, thus motivating a slightly different edge parameter resulting in an algorithm of complexity $O(m^4/n^2)$. Then, M. Newman and M. Girvan [25] introduced "modularity" to quantify how well a community structure "fits" a given network that led to an algorithm whose complexity is approximately $O(mn)$ [24], later to be improved by A. Clauset et al in [14]. J. Reichardt et al [29] found a method for the identification of fuzzy communities. Very recently, J.P. Bagrow and E.M. Bollt [3] developed a method of complexity $O(n^3)$ while A. Clauset [13] developed a method for finding "local" community structures.

A related problem is the *graph-partitioning problem* that has been much discussed in computer science over the last years. Here, the goal is to group the vertices in a graph into a given number of disjoint subsets of roughly equal size while minimizing the number of edges not fully contained in any one of these subsets.

The graph-partitioning problem is motivated by parallel processing. Some pertinent algorithms require one to input the number of communities in advance (which is a rather natural requirement in the context of parallel computing where this number represents the number of available processors), others propose to just iteratively bisect the given graph [23, 27].

The graph-partitioning differs from the community-detection problem in at least two aspects: In real-world applications, "natural" communities can neither be expected to be of roughly equal size nor can their number be fixed in advance. We will compare the community-detection and graph-partitioning problem in more detail in Section 4.

2 A Linear Programming Approach

Remarkably, a very straightforward approach published by M. Grötschel and Y. Wakabayashi in 1989 [17] and 1990 [18] was completely ignored in this context: Observe that

- identifying a "community structure" in a network is nothing but inserting and deleting edges in a somehow "most parsimonious" way so that the network becomes a *target network*, i.e., a disjoint union of complete subgraphs (or "cliques"),
- and that such networks are characterized by the (strictly local) property that any two distinct *incident* edges are part of a (necessarily uniquely determined) triangle.

Thus, describing the edges of a graph $G = (V, E)$ with vertex set V and edge set $E \subseteq \binom{V}{2}$ in terms of the associated *indicator function*

$$\chi_E : \binom{V}{2} \to \{0,1\} : \{u,v\} \mapsto \chi_E(uv) := \begin{cases} 1, & \text{if } \{u,v\} \in E, \\ 0, & \text{otherwise.} \end{cases}$$

A network $G = (V, F)$ is easily seen to be a target network if and only if

$$\chi_F(uv) + \chi_F(vw) - \chi_F(uw) \leq 1$$

holds for all $u, v, w \in V$. Indeed, denoting by $\mathbb{R}^{\binom{V}{2}}$ the $\mathbb{R}$-vector space consisting of all maps

$$\chi : \binom{V}{2} \to \mathbb{R} : \{u,v\} \mapsto \chi(uv)$$

from $\binom{V}{2}$ into $\mathbb{R}$, and by $P = P(V)$ the convex polytope consisting of all maps $x \in [0,1]^{\binom{V}{2}} \subseteq \mathbb{R}^{\binom{V}{2}}$ for which

$$\chi(uv) + \chi(vw) - \chi(uw) \leq 1 \tag{2.1}$$

holds for any three distinct elements $u, v, w \in V$, there is a canonical one-to-one correspondence between

(i) partitions Π of V into a disjoint union of subsets of V and

(ii) integer-valued maps $\chi \in P(V) \cap \mathbb{Z}^{\binom{V}{2}} = P(V) \cap \{0,1\}^{\binom{V}{2}}$ in $P(V)$ (all of which must be extremals — or vertices — of $P(V)$ because they are extremals already in the larger convex polytope $[0,1]^{\binom{V}{2}}$).

This correspondence is easily defined by associating, to each such map χ, the partition Π_χ of V into the equivalence classes relative to the equivalence relation $\sim_\chi$ defined on V by

$$u \sim_\chi v \iff u = v \text{ or } \chi(uv) = 1 \ \ (u, v \in V). \tag{2.2}$$

This binary relation is indeed an equivalence relation, even for any map $\chi \in P(V)$, because $\chi \in P(V)$, $u, v, w \in V$, $\#\{u, v, w\} = 3$, and $\chi(uv) = \chi(vw) = 1$ implies

$$1 \geq \chi(uw) \geq \chi(uv) + \chi(vw) - 1 = 1$$

and, therefore, $\chi(uw) = 1$ as required.

Consequently, all we need to do is to define what the term "most parsimonious" should mean in this context. The most simple way to measure the deviation of the original network $G = (V, E)$ from a target network $N = (V, F)$ with the same vertex set V is, of course, the total number $\#(E \triangle F)$ of inserted or deleted edges. This number can be expressed as

$$\begin{aligned}
\#(E \triangle F) &= \#(E - F) + \#(F - E) \\
&= \#E - \#(E \cap F) + \#\Big(F \cap \big(\binom{V}{2} - E\big)\Big) \\
&= \#E - \sum_{\{u,v\} \in E} \chi_F(uv) + \sum_{\{u,v\} \in \binom{V}{2} - E} \chi_F(uv) \\
&= \#E + \sum_{\{u,v\} \in \binom{V}{2}} \big(1 - 2\chi_E(uv)\big)\, \chi_F(uv)
\end{aligned}$$

and, thus, gives rise to a *penalty function* that is apparently an affine bilinear function of the two indicator functions χ_E and χ_F considers as vectors in the linear space $\mathbb{R}^{\binom{V}{2}}$. So, following the approach worked out so excellently in [18], we can use integer linear programming (ILP) to find an optimal target network relative to that penalty function — see, e.g., http://www.princeton.edu/~rvdb/LPbook for a freely available and very carefully worked out introduction into Linear and Integer Linear Programming.

However, ILP can easily accommodate also much more complex penalty functions: We are allowed to specify, for every 2-subset $\{u, v\} \in \binom{V}{2}$ of V, an arbitrary positive number $L_{apr}(uv)$ recording an *a priori* measure of the likelihood of the pair $\{u, v\}$ being contained (if in E) — or not being contained (if not in E) — in the same community within the community structure we want to detect, and then use ILP to determine that target network for which the resulting penalty function

$$-\sum_{\{u,v\}\in E} L_{apr}(uv)\,\chi_F(uv) \;+\sum_{\{u,v\}\in\binom{V}{2}-E} L_{apr}(uv)\,\chi_F(uv)$$

is minimized. Note that the numbers $L_{apr}(uv)$ could be derived from the overall graph structure as well as from any additional information we may have gathered. In particular, it may be tempting to experiment with the various "betweenness" parameters discussed in the literature quoted above.

Currently, we are using the "CPLEX" software package to investigate this approach, experimenting just for a start with the parameterized linear programming problem $LP(G, \mathbf{s})$ of searching for a vertex χ in $P(V)$ or in $P(V) \cap \{0, 1\}^{\binom{V}{2}}$ minimizing the *a priori* likelihood function

$$L_{G,\mathbf{s}}(uv) := \begin{cases} \mathbf{s}\big(\deg_G(u) + \deg_G(v)\big), & \text{if } \{u, v\} \in E, \\ (2\#V - 2 - \deg_G(u) - \deg_G(v)), & \text{otherwise,} \end{cases}$$

where $\deg_G(x)$ is, for any vertex x in a graph $G = (V, E)$, of course the number of edges in E incident with x, and $\mathbf{s}$ is a positive real number that we use as a *control parameter*for appropriately *calibrating* our penalty function.

3 A Surprising Discovery and an Accompanying Theorem

Clearly, while M. Grötschel' and Y. Wakabayashi's work addressed the problem of solving a specific single integer linear programming problem, i.e., the *clique-partitioning*

problem, it was not designed to be used for deriving plausible community structures. Thus, to using it for this purpose, we introduced — and experimented with — the control parameter $\mathbf{s}$ searching for ways to flexibly and adaptively identifying those values of $\mathbf{s}$ that would help to unravel the "proper" community structure (if any) underlying a given network under consideration.

To our own surprise, we quickly discovered a way for just doing this. More specifically, we observed that, increasing the control parameter $\mathbf{s}$ from 1 to larger and larger values, the running times of the ILP problem became shorter and shorter until a value $\mathbf{s}^*$ was reached for which

(i) the running time was approximately that of the associated *relaxed* linear programming (LP) problem,

(ii) the solutions of both problems coincided (i.e., the relaxed problem had an integral solution), and

(iii) the resulting community structure was approximately that one which, by other researchers, was considered to be a "plausible" one.

This finding suggested the following strategy for detecting community structures associated to a given graph G:

- Start with $\mathbf{s} := 1$.
- Use CPLEX 9.1 (or any other good software tool for solving LP problems) to find vertices in $P(V)$ that solve the linear programming problem $LP_{G,\mathbf{s}}$.
- Increase $\mathbf{s}$ continuously in sufficiently small steps until the smallest value $\mathbf{s}^* = \mathbf{s}^*(G) \in [1, +\infty)$ is found for which this problem has an integer solution $\chi^*(G)$.
- And then stop and consider the associated partition $\Pi(G) := \Pi_{\chi^*(G)}$ as a hopefully reasonably good solution of the original problem.
- Finally, as the resulting primary output may exhibit isolated vertices and edges, we may, in a final packing step, join an isolated vertex or edge with the vertices of that neighboring group (in the primary output) that has the highest number of connections to it (in the original graph). If two or more neighboring groups have the same number of connections, we choose that group for which the degree sequence of the vertices connecting to the isolated vertex or edge is maximal relative to dominance order (and give up if that does not break the deadlock).

In this paper, we report some first results that were obtained applying this procedure. The first problem to address is, of course, the question whether such a value $\mathbf{s}^* = \mathbf{s}^*(G) \in [1, +\infty)$ will always exist?

It is easy to show that this is indeed the case. More generally and more precisely, denoting by $\pi(G)$ the partition of V into the set of connected components of G, we will show here that, provided $\mathbf{s}$ is sufficiently large so that eliminating just a single edge only becomes much more expensive than to insert all missing ones, the above LP-problem has, for any given map

$$L : \binom{V}{2} \to \mathbb{R}_{>0} : \{u, v\} \mapsto L(uv)$$

from $\binom{V}{2}$ into the set $\mathbb{R}_{>0}$ of positive real numbers, a unique integer solution corresponding to exactly this partition. Performing the required explicit estimations easily leads to

Theorem 3.1. *Given a simple finite graph $G = (V, E)$, there exists a unique vertex and, therefore, only one point $\chi = \chi^G$ in $P(V)$,* viz.*, the map for which the corresponding partition Π_{χ^G} of V coincides with the partition of V into the set of connected components of G, such that, for every map $L : \binom{V}{2} \to \mathbb{R}_{>0} : \{u, v\} \mapsto L(uv)$, there exists some positive real number $\mathbf{s}_{G,L} \geq 1$ for which $L^{G,\mathbf{s}}(\chi^G) \leq L^{G,\mathbf{s}}(\eta)$ holds, for all real numbers $\mathbf{s} \geq \mathbf{s}_{G,L}$ and all $\eta \in P(V)$, for the the associated linear form*

$$L^{G,\mathbf{s}} : \mathbb{R}^{\binom{V}{2}} \to \mathbb{R} : \eta \mapsto -\mathbf{s} \sum_{\{u,v\} \in E} L(uv)\, \eta(uv) + \sum_{\{u,v\} \in \mathcal{P}_2(V) - E} L(uv)\, \eta(uv).$$

More precisely, denoting by $\epsilon(G, L)$ the minimum, over all vertices η of $P(V)$ with $\sum_{\{u,v\} \in E} L(uv)\,(1 - \eta(uv)) \neq 0$, of exactly all of these (finitely many) non-zero terms, we have $L^{G,\mathbf{s}}(\eta) \geq L^{G,\mathbf{s}}(\chi^G)$ for all $\mathbf{s}$ with

$$\mathbf{s} > \frac{\sum_{\{u,v\} \in \binom{V}{2} - E} L(uv)}{\epsilon(G, L)}$$

and for all maps $\eta \in P$.

Proof: Note first that, given any map $\eta \in P$, we must have

$$\sum_{\{u,v\} \in E} L(uv)\, \eta(uv) \leq \sum_{\{u,v\} \in E} L(uv),$$

and that equality holds for some $\eta \in P$ if and only if $\eta(uv) = 1$ holds for all $\{u, v\} \in E$ and, therefore, for all $\{u, v\} \in \binom{V}{2}$ with $\chi^G(uv) = 1$. In consequence, we have

$\sum_{\{u,v\}\in E} L(uv)\,\eta(uv) = \sum_{\{u,v\}\in E} L(uv)$ for some map $\eta \in P$ if and only if $\chi^G(uv) \leq \eta(uv)$ holds for all $\{u,v\} \in \binom{V}{2}$. In turn, this implies that

$$\begin{aligned}
&L^{G,\mathbf{s}}(\eta) - L^{G,\mathbf{s}}(\chi^G)\\
&= \mathbf{s} \sum_{\{u,v\}\in E} L(uv)\,(\chi^G(uv) - \eta(uv)) - \sum_{\{u,v\}\in\binom{V}{2}-E} L(uv)(\chi^G(uv) - \eta(uv))\\
&= - \sum_{\{u,v\}\in\binom{V}{2}-E,\,\chi^G(uv)\neq 1} L(uv)\,(\chi^G(uv) - \eta(uv))\\
&= \sum_{\{u,v\}\in\binom{V}{2}-E,\,\chi^G(uv)=0} L(uv)\,\eta(uv)\\
&\geq 0
\end{aligned}$$

must hold for any such η, with equality holding if and only if $\eta = \chi^G$ holds. Thus, if a vertex η of $P(V)$ with $\sum_{\{u,v\}\in E} L(uv)\,\eta(uv) = \sum_{\{u,v\}\in E} L(uv)$ minimizes $L^{G,\mathbf{s}}$ over $P(V)$, it must coincide with χ^G.

Moreover, if a vertex η of $P(V)$ minimizes $L^{G,\mathbf{s}}$ over $P(V)$ and if $\mathbf{s}$ is very large, we must have $\sum_{\{u,v\}\in E} L(uv)\,\eta(uv) = \sum_{\{u,v\}\in E} L(uv)$ and, therefore $\eta = \chi^G$: Indeed, let $\epsilon = \epsilon(G,L)$ denote the minimum, over all vertices η of $P(V)$, of the non-zero terms of the form $\sum_{\{u,v\}\in E} L(uv)\,(1-\eta(uv))$, so that $\epsilon > 0$ holds by definition (as there are only finite vertices of $P(V)$), then we have

$$\begin{aligned}
L^{\mathbf{s}}(\eta) &= -\mathbf{s} \sum_{\{u,v\}\in E} L(uv)\,\eta(uv) + \sum_{\{u,v\}\in\binom{V}{2}-E} L(uv)\,\eta(uv)\\
&\geq -\mathbf{s}\,\big(-\epsilon + \sum_{\{u,v\}\in E} L(uv)\big) + \sum_{\{u,v\}\in\binom{V}{2}-E} L(uv)\,\eta(uv)\\
&\geq -\mathbf{s}\,\big(-\epsilon + \sum_{\{u,v\}\in E} L(uv)\big)\\
&> -\mathbf{s} \sum_{\{u,v\}\in E} L(uv) + \sum_{\{u,v\}\in\binom{V}{2}-E} L(uv)\\
&\geq L^{G,\mathbf{s}}(\chi^G)
\end{aligned}$$

for every such $\eta \in P$ with $\sum_{\{u,v\}\in E} L(uv)\,(1-\eta(uv)) > 0$ whenever

$$\mathbf{s}\,\epsilon > \sum_{\{u,v\}\in\binom{V}{2}-E} L(uv)$$

or, equivalently,

$$\mathbf{s} > \frac{\sum_{\{u,v\}\in\binom{V}{2}-E} L(uv)}{\epsilon}$$

holds. ∎

This shows that one can always find some real number $\mathbf{s} \geq \mathbf{1}$ for which the LP problem has an integer solution and, hence, a smallest such number for which, using the estimates above, one could also derive explicit upper bounds. So, we will never search in vain when trying to determine $\mathbf{s}^*$. We will now present some results we have obtained in this way.

4 Experimental Results

We consider first the well known data regarding *Zachary's karate club* [36] that describes a simple graph with 34 vertices. As explained in [36], this club broke in two due to some internal strive, and W.W. Zachary investigated which members were on good terms with each other before that happened.

In this case, the partition associated with the first integer-valued solution of the associated LP-problem illustrated in Fig. 1 by the partition line, was obtained for $\mathbf{s} = \mathbf{s}^* \approx 38.8$. Remarkably, it coincides exactly with the real-world situation. Denoting the time ILP needs to find a solution by $t(\mathbf{s})$, the map $t \mapsto t(\mathbf{s})$ is plotted in Fig. 2, the proportion between the optimal value of the ILP problem and the associated relaxed LP problem is, as a function of $\mathbf{s}$, plotted in Fig. 3, and the proportion between the respective computation times is plotted in Fig. 4.

The second example is the *Chesapeake-Bay Food Web* [4]. This network representing the 33 most prominent marine organisms living in this large estuary on the east coast of the United States was first compiled by D. Baird and R.E. Ulanowicz. The edges indicate *trophic* relationships (i.e., who eats whom). The network together with the result of our algorithm are shown in Fig. 5, and related data in Figs. 6 and 7. We also compare, in Fig. 8, our result with that of the GN-algorithm [16].

For obtaining an expert's comment, we sent our files to R. Ulanowicz. He found both groupings quite good and noted that there is only one transposition distinguishing our grouping and that by M. Girvan and M. Newman. Namely, we group Blue Crab (19) in the benthic group (organisms that find their food on the sea bottom) and Bluefish (30) in the pelagic group (organisms that find their food in the open water). The GN-algorithm does the opposite.

R. Ulanowicz also remarked that, from a biologist's point of view, the groupings are according to where the organisms "feed", not where they are located — shedding also some light on the question raised by M. Girwan and M. Newman in this context in [16] who wondered whether "the simple traditional division of taxa into pelagic or benthic may not be an ideal classification in this case".

He writes: "That clams (12), oysters (13), and other suspension feeders (11) live on the bottom is only incidental. They are all filter feeders and take their nourishment from the water column. In terms of *feeding*, they belong with the pelagic organisms." Following his advice, we present the "real" community structure and the comparison between LP method and GN method in Fig. 8 (which,taking this into account, differs a little from that in [16]).

As for the single discrepancy between the two methods, he would judge that Blue Crab (19) belongs decidedly to the benthic feeding group as detected by our method. Bluefish (30) feed mostly on other nekton, but ultimately derive most of their sustenance from the benthos. In fact, R. Ulanowicz notes in [32] how the indirect diet of Striped Bass (33) differs from that of Bluefish (30) in that the former derives most of its sustenance from the pelagic domain whereas the latter derives it ultimately (but not directly) from the benthos. Hence, he told us that "Bluefish is a borderline species and the GN-algorithm does not err gravely by placing it among the benthic feeders" — while placing Blue Crab (19) among the pelagic feeders is simply wrong.

The third example is the food web of *Marshes and Sloughs*. This is an ecosystem in Florida Bay of the United States. We got the data from the project's webpage [19]. As above, the edges indicate trophic relations, here involving altogether 63 species. We again get a plausible result using our LP-based method. Among the two communities we get, one seems to merge almost all of the organisms living in, or being directly related to, water (like fishes, water plants, some amphibians, etc.) as well as some birds that feed on such species. The second community contains all the mammals all of which are not related to water. We again asked for expert comments, and they confirmed [6] that, basically speaking, the classification according to "wetland" and "upland" habitats was right; the only error is "Lizards" as these are not so directly related to water and would better be put into the "upland" group. So, there is essentially only one error among those 63 organisms under investigation.

We also applied various graph-partition algorithms to these data. As the Kernighan-Lin [23] and the spectral bisection [27] algorithm clearly do not pursue the goal of detecting community structures, we restricted our attention to the rather successful and fast "METIS" software [20] developed by G. Karypis and V. Kumar [21, 22] that has become very popular for dealing with the graph-partitioning problems. There are

two variants, "kmetis" and "pmetis" both of which are known to produce very good results. We applied both of them to the three examples above.

To this end, we had to predefine the number N of communities and, using our *a priori* knowledge, we put $N := 2$ for all three examples. For the first example, the two programs produced identical communities with 17 vertices in each of them, moving Node **10** from the left-hand side in Fig. 1 over to the right-hand side.

For the Chesapeake-Bay food web with its 33 species, the two programs also produced identical results, one 16-species and one 17-species community. According to R. Ulanowicz, there is only one error in this result: Bay anchovy (22) was mistakenly put into the benthic group. So, for this example, METIS produced a better result than both, the GN- and the LP-based procedure, provided you put $N := 2$. However, if we put $N := 3$ (as the latter two methods really suggest that there should be three communities), the METIS-algorithms produce, not unexpectedly, more mistakes. Measuring the number of mistakes using the "single-element transfer distance" introduced by Charon et al [11] (also discussed in [12]), for kmetis the distance is 12, and for pmetis the distance is 9.

Finally, METIS produces two communities containing 31 and 32 species, respectively, for *Marshes and Sloughs*. The two programs again give different results, but the sizes of their communities and the number of their errors are the same. We evaluated the result produced by METIS (comparing with R. Ulanowicz's judgement) and found that the number of errors (distance) is 23 (recall that LP method produces a result with just 1 error). As the "real" community sizes for *Marshes and Sloughs* are highly asymmetric, one can, of course, not expect the METIS software to produce a reasonable result.

From the above tests, we conclude that it is not advisable to simply use graph-partitioning algorithms for community detection.

We also applied our approach to yet another real-world example: a protein network involving 101 proteins [26]. Our algorithm reveals their relationships quite well, but we skip the results here (to be published — together with results obtained for simulated data — elsewhere [12] and available upon request) and, instead, shortly discuss the time complexity of our approach:

First note that, after our initial experiments that led to the discovery communicated above, our goal soon became finding the smallest value of $\mathbf{s}$ for which the LP problem $LP(G, \mathbf{s})$ has an integer solution. In this note, we discussed and compared both, the ILP problems and their LP relaxations, only to motivate the approach that we outlined above.

In this approach, the main task is solving a sequence of LP — and not of ILP

— problems. So, it does not involve intrinsically NP-hard problems. However, although there are polynomial algorithms for solving LP problems in theory (such as the Inner-Point Method), in most practical cases (and provably in the average, cf. [7]-[10] and [30]), the dual simplex method performs best though it can need exponential time in worst-case examples. And it is this algorithm that is also used in CPLEX. Consequently, one cannot give a rigorous analysis of the complexity of our algorithm yielding sensible polynomial bounds unless, following the papers by Borgwardt and Smale quoted above, one addresses the *average* complexity of our algorithm which, however, is clearly beyond the scope of the present paper.

However, as suggested by these results, the actual total computation time needed for the four examples (including the variation of $\mathbf{s}$) is surprisingly small:

Computation Time for the Four Examples

Example	Computation Time
Karate Club	5.22 seconds
Chesapeake Bay	3.35 seconds
Marshes and Sloughs	6 minutes
Protein Network	183 minutes

5 Directions for Future Work

Regarding possible usages of the LP-based approach to community detection, note first that users can easily — and in a rather transparent manner — play with the algorithm to investigate the consequences of specific requirements by varying and experimenting with the respective penalty functions. So, we expect it to be particularly useful as a feasible alternative to the currently popular network-clustering algorithms for a first exploration of network communities, their stability, and hypotheses regarding their structure.

To make this possible, various questions deserve to be investigated and several tasks to be pursued in the future:

1. Studying a larger range of penalty functions and trying to determine those that seem to be particularly *appropriate* for a specific task, including penalty functions related to edge-weighted networks and/or asymmetric ones representing directed networks.

2. Trying to understand the influence exercised by, and in particular the apparent "phase transition" behavior of, the control parameter **s** as illustrated by Figs 2, 3, 4, 6, and 7.

3. Analyzing the "landscape" defined on the set of target graphs by a given (**s**-parameterized) penalty function using, in particular, the entropy concept from statistical physics.

4. Developing approximative algorithms for *large-scale* applications.

5. Creating a data base containing the results obtained by applying the algorithm(s) to *real-world* data gathered from the existing literature.

References

[1] R. Albert, H. Jeong, and A.-L. Barabási, Diameter of the World-Wide Web. *Nature* 401, 130-131 (1999).

[2] L. A. N. Amaral, A. Scala, M. Barthélémy and H. E. Stanley, Classes of samll-world networks. *Proc. Natl. Acad. Sci. USA* 97, 11149-11152 (2000).

[3] J. P. Bagrow and E. M. Bollt, Local method for detecting communities. *Phys. Rev. E* 72, 046108 (2005).

[4] D. Baird and R. E. Ulanowicz, The seasonal dynamics of the Chesapeake Bay ecosystem. *Ecological Monographs* 59, 329-364 (1989).

[5] A.-L. Barabási and R. Albert, Emergence of scaling in random networks. *Science* 286, 509-512 (1999).

[6] C. Bondavalli, R. Ulanowicz, email communication (2006).

[7] K.-H. Borgwardt, Untersuchungen zur asymptotik der mittleren schriftzahl von simplexverfahren in der linearen optimierung. Dissertation, Universität, Kaiserlautern (1977).

[8] K.-H. Borgwardt, Some distribution-independent results about the asymptotic order of the average number of pivot steps of the simplex method. *Math. Oper. Res.* 7, 441-462 (1982).

[9] K.-H. Borgwardt, The average number of steps required by the simplex method is polynomial. *Zeitschrift für Operations Research* 26, 157-177 (1982).

[10] K.-H. Borgwardt, Average behavior of the simplex algorithm: Some improvements in the analysis of the rotation-symmetry-model. *The 12th Symposium on Mathematical Programming*, Cambridge, Mass.

[11] I. Charon, L. Denoeud, A. Guénoche and O. Hudry, Maximum transfer distace between partitions. *J. Classif.* 23 (1), 103-121 (2006).

[12] W. Y. C. Chen, A. W. M. Dress and W. Q. Yu, Checking the reliability of a linear-programming based approach towards detecting community structures in networks, *IET Sys. Biol.*, to appear.

[13] A. Clauset, Finding local community structure in networks. *Phys. Rev. E* 72, 026132 (2005).

[14] A. Clauset, M. E. J. Newman, and C. Moore, Finding community structure in very large networks, *Phys. Rev. E* 69, 026113 (2004).

[15] G. W. Flake, S. R. Lawrence, C. L. Giles, and F. M. Coetzee, Self-organization and identification of Web communities. *IEEE Computer* 35, 66-71 (2002).

[16] M. Girvan and M. E. J. Newman, Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* 99, 7821–7826 (2002).

[17] M. Grötschel, Y. Wakabayashi, A cutting plane algorithm for a clustering problem. *Mathematical Programming* 45, 59-96 (1989).

[18] M. Grötschel, Y. Wakabayashi, Facets of the Clique Partitioning Polytope. *Mathematical Programming* 47, 367-387 (1990).

[19] `http://www.cbl.umces.edu/~atlss/ATLSS.html`, accessed November 2006.

[20] `http://glaros.dtc.umn.edu/gkhome/metis/metis/download`, accessed March 2007.

[21] G. Karypis and V. Kumar, Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing* 48(1), 96C129 (1998).

[22] G. Karypis and V. Kumar, A fast and highly quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.* 20(1), 359-392 (1998).

[23] B. W. Kernighan and S. Lin, An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal* 49, 291-307 (1970).

[24] M. E. Newman, Fast algorithm for detecting community structure in networks. *Phys. Rev. E* 69, 066133 (2004).

[25] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, *Phys. Rev. E* 69, 026113 (2004).

[26] A. Pocklington, M. Cumiskey, J. Armstrong and S. Grant, The proteomes of neurotransmitter receptor complexes form modular networks with distributed functionality underlying plasticity and behaviour. *Molecular System Biology* doi: 10.1038/msb4100041, (2006).

[27] A. Pothen, H. Simon, and K.-P. Liou, Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.* 11, 430-452 (1990).

[28] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto and D. Parisi, Defining and idetifying communities in networks. *Proc. Natl. Acad. Sci. USA* 101, 2658-2663 (2004).

[29] J. Reichardt and S. Bornholdt, Detecting Fuzzy Community Structures in Complex Networks with a Potts Model, *Phys. Rev. Lett.* 93, 218701 (2004).

[30] S. Smale, The problem of the average speed of the simplex method. In A. Bachem, M. Grötschel and B. Korte (eds.), *Mathematical programming: The state of the art*, 530-539. Springer-verlag, Berlin.

[31] J. R. Tyler, D. M. Wilkinson, and B. A. Huberman, Email as spectroscopy: Automated discovery of community structure within organizations. In M. Huysman, E. Wenger, and V. Wulf (eds.), *Proceedings of the First International Conference on Communities and Technologies*, Kluwer, Dordrecht (2003).

[32] R. Ulanowicz, Quantitative methods for ecological network analysis. *Comput. Biol. Chem.* 28, 321-339 (2004).

[33] S. Wasserman and K. Faust, *Social Network Analysis.* Cambridge University Press, Cambridge (1994).

[34] D. J. Watts and S. H. Strogatz, Collective dynamics of 'small world' networks. *Nature* 393, 440-442 (1998).

[35] F. Wu and B. A. Huberman, Finding communities in linear time: A physics approach. *Eur. Phys. J. B* 38, 331-338 (2004).

[36] W. W. Zachary, An information flow model for conflict and fisson in small groups. *Journal of Anthropological Research* 33, 452-473 (1977).

Acknowledgment

The authors are grateful to R. Ulanowicz and C. Bondavalli for their help on the food-web data and the comments on our results, and to Roger Q.L. Yu and Tyll Krüger for many helpful conversations. This work was supported by the 973 Project on Mathematical Mechanization, the PCSIRT Project of the Ministry of Education, the Ministry of Education, the Ministry of Science and Technology, and the National Science Foundation of China, and the Max Planck Society, Germany.

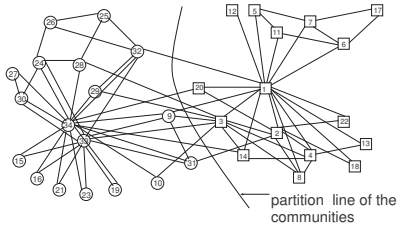

Figure 1: The original karate-club graph and the result of our algorithm.

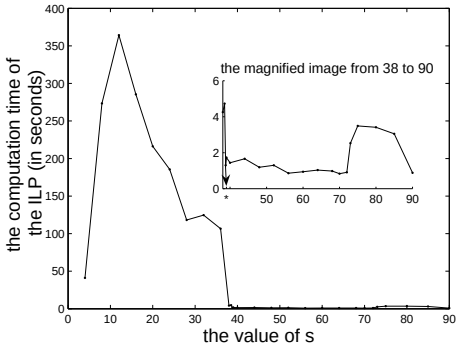

Figure 2: The ILP computation time as a function of the parameter **s** .

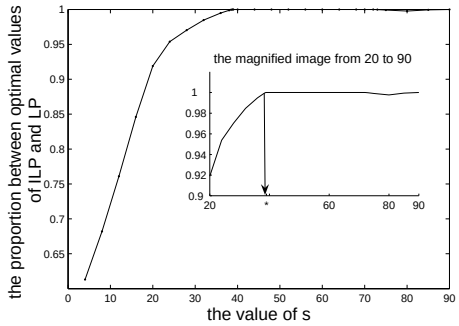

Figure 3: The proportion between the optimal ILP and LP values as a function of the parameter **s**.

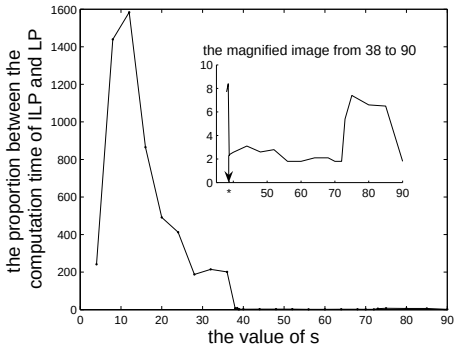

Figure 4: The proportion between the ILP and LP computation time as a function of the parameter **s**.

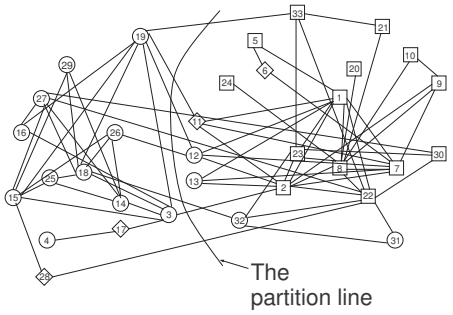

Figure 5: The Chesapeake-Bay food web.

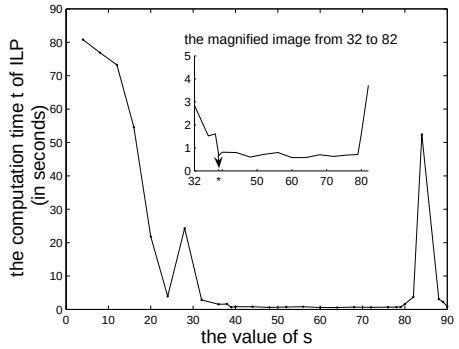

Figure 6: The ILP computation time for the Chesapeake-Bay food web as a function of the parameter **s**.

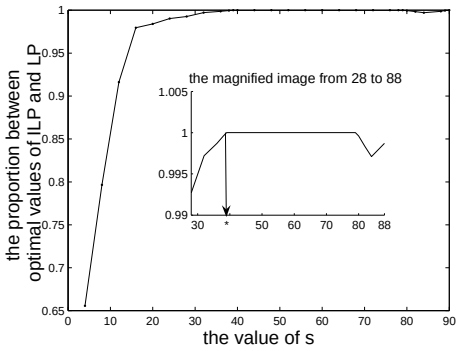

Figure 7: The proportion between the optimal ILP and LP values for the Chesapeake-Bay food web as a function of the parameter **s** .

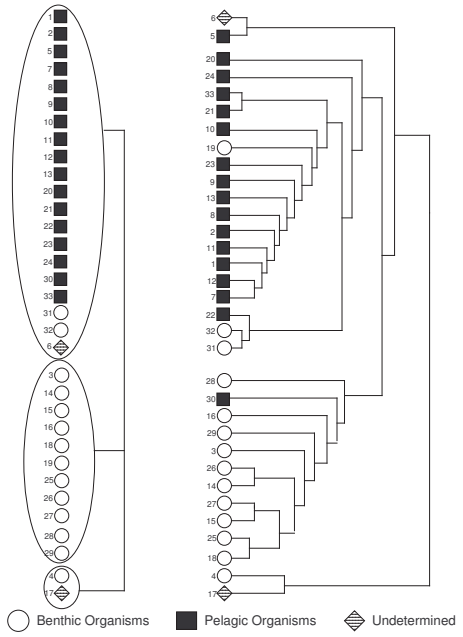

Figure 8: The results of the GN-algorithm (right) and our algorithm for the Chesapeake-Bay food web.