algorithms

MDPI

Article

Community Structure Detection for Directed Networks through Modularity Optimisation

Lingjian Yang [1,†], Jonathan C. Silva [2,†], Lazaros G. Papageorgiou [1,*] and Sophia Tsoka [2,*]

1 Centre for Process Systems Engineering, Department of Chemical Engineering, University College London, London WC1E 7JE, UK; lingjian.yang.10@alumni.ucl.ac.uk

2 Department of Informatics, Faculty of Natural and Mathematical Sciences, King's College London, London WC2R 2LS, UK; jonathan.silva@kcl.ac.uk

* Correspondence: l.papageorgiou@ucl.ac.uk (L.G.P.); sophia.tsoka@kcl.ac.uk (S.T.)

† These authors contributed equally to this work.

Academic Editors: Tatsuya Akutsu and Takeyuki Tamura

Received: 14 August 2016; Accepted: 18 October 2016; Published: 1 November 2016

Abstract: Networks constitute powerful means of representing various types of complex systems, where nodes denote the system entities and edges express the interactions between the entities. An important topological property in complex networks is community structure, where the density of edges within subgraphs is much higher than across different subgraphs. Each of these subgraphs forms a community (or module). In literature, a metric called modularity is defined that measures the quality of a partition of nodes into different mutually exclusive communities. One means of deriving community structure is modularity maximisation. In this paper, a novel mathematical programming-based model, DiMod, is proposed that tackles the problem of maximising modularity for directed networks.

Keywords: community detection; directed networks; modularity optimisation; integer programming; complex networks

1. Introduction

It is increasingly clear that a wide range of systems across different disciplines can be described using network representations. The edges in a network can be binary, weighted, directed and with a positive or negative sign, thus rendering networks a suitable tool to model diverse types of interactions [1]. One common observation is that real world networks are not random graphs with homogeneous edge distribution, rather, a high density of edges exist within subsets of the network, while edge density across these subgraphs is much lower, giving rise to the emergence of a property known as community structure [2].

Community structure detection refers to the procedure of identifying the inherent higher order structure of a network by partitioning nodes into different modules. The modularity metric (Q) was introduced by Newman and Girvan [3] for undirected networks and measures the quality of a network partition into communities. A modularity value close to 1 suggests strong community structure, i.e., a larger proportion of edges falling within modules than at random.

Community detection is often formulated as an optimisation problem, where the partition that yields the maximum modularity for a target network is sought. This metric, however, has been proven to be NP-complete [4], and, therefore, exhaustive search algorithms are only applicable to small networks, as the number of possible partitions increases at least exponentially with the number of nodes. Modularity has also been shown to have a resolution limit [5] and a degenerate solution space [6]. There have been attempts in the literature to address resolution limit issues by adding a

resolution parameter, but, as this parameter is hard to determine and increases method complexity, the problem is still not fully resolved [7].

Despite its limitations, modularity is still the most popular community detection metric [7] and it has been used in a variety of applications. Various optimisation algorithms have been proposed in literature that are based on a number of methodologies, including simulated annealing [8,9], greedy algorithm [10], extremal optimisation [11], spectral algorithm [3] and integer programming-based optimisation model [12–17].

While modularity optimisation for undirected networks has been well studied over the past decade, little has been done for module detection in directed networks. Many real world networks are inherently directed, including the World Wide Web [18], brain neural networks [19] and metabolic networks, where the directed edge represents material flow from one substrate to a product that may be irreversible [20]. In literature, the conventional manner of tackling the problem of community detection in directed networks is simply to treat the networks as undirected and apply the methods described above [21]; however, such approaches lose valuable information carried in edge directionality. Some algorithms use edge directionality to transform directed graphs to weighted networks or bipartite networks [22]. Recent algorithms propose the use of spectral optimisation [23], local extraction of communities [24] and blockmodeling or statistical inference models [25].

In this study, we use the mathematical description of modularity for directed networks introduced by Leicht and Newman [26] that explicitly considers the in and out-degree distributions of nodes in the network. We propose a two-step algorithm named DiMod composed of two mathematical programming models to detect modules in directed networks by maximising modularity. The two models have equivalent objective functions, but the terms of the equations are rearranged differently to accentuate a desired property of the mathematical model, as described in the section below. We compare the results of our method to three algorithms implemented in the Radatools software (v.4.0, DEIM, Tarragona, Spain) [27] on synthetic and real networks used as test cases.

2. Iterative Mathematical Programming Model for Modularity Optimisation on Directed Networks

We propose an iterative procedure that contains two major mathematical models to optimise modularity for directed networks. The first is a Mixed Integer Non-linear Programming Model (MINLP) that can provide a good partition of the network quickly but is likely to converge to a local optimal region, while the second model, a Mixed Integer Linear Programming Model (MIP), is harder to solve but is capable of finding a solution of higher quality. The algorithm starts by solving the MINLP a number of times, and the best partition is then selected and given as an initial point to the reduced MIP model that will iteratively improve the solution. We reduce the complexity of the MIP by allowing a few nodes to change modules while keeping all other nodes in their initial allocation and we solve each of these reduced MIPs for each module in an iteration.

As mentioned previously, the reason for proposing two different models for solving the same problem is that although a MINLP can find an acceptable solution for a large problem, it often converges to locally optimal solutions, and thus the quality of the solution is hard to guarantee. On the other hand, an MIP model can be solved to global optimality for small problems but consumes large computational resources for larger networks. We combine the best of these two types of models by first solving the MINLP and then the reduced MIPs to improve the acquired solution. The entire computational strategy is named DiMod and the mathematical description follows.

The sets, parameters and variables used in the models are described below:

Sets

n, e	node
m	module
l_{ne}	directed edge pointing from node n to e

Parameters

β_{ne}	weight of edge point from node n to e
d_n^{in}	sum of weights over all edges points to node n; incoming edge weight
d_n^{out}	sum of weights over all edges points from node n; outgoing edge weight
L	total amount of weights over all edges in the given network

Binary Variables

Y_{nm}	1 if node n belongs to module m; 0 otherwise

Free Variables

D_m^{in}	sum of d_n^{in} for all nodes that belong to module m ($Y_{nm} = 1$)
D_m^{out}	sum of d_n^{out} for all nodes that belong to module m ($Y_{nm} = 1$)
L_m	sum of edge weights in module m
LS_{nem}	a positive intermediate variable. $LS_{nem} = \beta_{ne}$ if both nodes n to e belong to module m; 0 otherwise
$D_m^{in}Y_{nm}$	represent the product of D_m^{in} and Y_{nm}, used as an intermediate variable for the MIP model
$DD_m^{in\_out}$	represent the product of D_m^{in} and D_m^{out}, used as an intermediate variable for the MIP model

The next two sections describe the two models, and the following section describes our proposed algorithm and how both stages are employed to find the models of directed networks.

2.1. First Model—MINLP

Modularity is defined as the number of edges that fall within communities minus the expected number of edges that should fall into communities in a null configuration of the equivalent network with edges being placed at random [3]. The modularity for directed networks can be formulated as below [26]:

$$Q = \sum_m \left(\frac{L_m}{L} - \frac{D_m^{in} D_m^{out}}{L^2} \right), \tag{1}$$

where $\frac{L_m}{L}$ represents the fraction of (weighted) edges that fall into module m, while $\frac{D_m^{in} D_m^{out}}{L^2}$ is the expected value for module m.

The sum of weights of edges in module m, L_m, is computed as:

$$L_m = \sum_{n,e \in l_{ne}} \beta_{ne} Y_{nm} Y_{em} \quad \forall m, \tag{2}$$

indicating that an edge pointing from node n to node e is included in a module m if and only if both nodes belong to module m, i.e., $Y_{nm} = 1$ and $Y_{em} = 1$.

For a given directed network, the sum of weights of edges coming into node n is denoted as parameter d_n^{in}, while the sum of weights of edges pointing from node n is d_n^{out}. For an unweighted network, d_n^{in} and d_n^{out} are simply the in-degree and out-degree of node n. For a module m, the sum of d_n^{in} and d_n^{out} over all nodes belonging to this module ($Y_{nm} = 1$) are, respectively, calculated as below:

$$D_m^{in} = \sum_n d_n^{in} Y_{nm} \quad \forall n, \tag{3}$$

$$D_m^{out} = \sum_n d_n^{out} Y_{nm} \quad \forall n. \tag{4}$$

We are interested in non-overlapping partitions where each node can only be allocated to exactly one module, and this is modeled via the following constraints:

$$\sum_m Y_{nm} = 1 \quad \forall n. \tag{5}$$

The first step of the algorithm, the full MINLP model, is summarised below:

maximize	Q	
subject to constraints	(Equations (1)–(5))	
	$L_m, D_m^{in}, D_m^{out} >= 0$	$\forall m,$
	$Y_{nm} \in \{0,1\}$	$\forall n, m.$

2.2. Second Model—MIP

The presence of non-linearity, combined with the use of integer variables, present considerable computational difficulty for finding globally optimal solutions. Solving MINLP problems typically involves repeatedly specifying different initial starting points and solving the model to identify locally optimal solutions, which can generally be realised in affordable computational time. Thus, the MINLP model in the previous section is used to provide an initial network division before a more sophisticated method can be applied to refine the division. In this section, the MINLP model is reformulated as an MIP by redefining the two non-linear constraints (Equation (2) and the multiplication $D_m^{in} D_m^{out}$ in Equation (1)) as linear constraints.

Firstly, Equation (2) can be replaced by the following three sets of constraints:

$$LS_{nem} \leq \beta_{ne} Y_{nm} \quad \forall n, e \in l_{ne}, m, \tag{6}$$

$$LS_{nem} \leq \beta_{ne} Y_{em} \quad \forall n, e \in l_{ne}, m, \tag{7}$$

$$LS_m = \sum_{n, e \in l_{ne}} LS_{nem} \quad \forall m, \tag{8}$$

where LS_{nem} are newly introduced positive intermediate variables. For edges from node n to e, LS_{nem} is equal to its weight, β_{ne}, if it belongs to module m; 0, otherwise.

The non-linear term in the objective function, $D_m^{in} D_m^{out}$, can be re-written as below:

$$\begin{aligned} D_m^{in} D_m^{out} &= D_m^{in} (\sum_n d_n^{out} Y_{nm}) \\ &= \sum_n d_n^{out} (D_m^{in} Y_{nm}), \end{aligned}$$

where $D_m^{in} Y_{nm}$ is the product of a continuous variable D_m^{in} and a binary variable Y_{nm} and can be made linear using the following set of equations:

$$DY_{nm} \geq D_m^{in} - U(1 - Y_{nm}) \quad \forall n, m, \tag{9}$$

$$DD_m^{in\_out} \geq \sum_n d_n^{out} DY_{nm} \quad \forall m, \tag{10}$$

where DY_{nm} are introduced as new positive variables to replace the term $D_m^{in} Y_{nm}$, the variable $DD_m^{in\_out}$ is introduced to replace $D_m^{in} D_m^{out}$, and U is an arbitrarily large number.

The formulation of the objective function for this model becomes:

$$Q = \sum_m \left(\frac{L_m}{L} - \frac{DD_m^{in\_out}}{L^2} \right), \tag{11}$$

and the full model is given by the set of equations below:

$$\begin{array}{lll} \text{maximize} & Q & \\ \text{subject to constraints} & (\text{Equations (3), (5)–(11)}) & \\ & L_m, D_m >= 0 & \forall m, \\ & Y_{nm} \in \{0,1\} & \forall n, m. \end{array}$$

2.3. Full Algorithm

The proposed algorithm has three user-defined parameters: N_{MINLP}, the number of iterations for the MINLP model; N_{MIP}, the number of iterations for the MIP model and $N_{relaxed}$, the number of nodes to be released for each module at each MIP run. The optimal number of modules is determined by the algorithm itself, but we set a maximum number of modules to generate the set m. After the first model is solved N_{MINLP} times, the best value of modularity (Q_{best}) and its corresponding partition (Y_{nm}^{best}) are selected and used in the second model. At each iteration of the MIP, $N_{relaxed}$ nodes are allowed to change modules, while the remaining nodes are kept in their previous modules. Each node within the current module is relaxed once only and the procedure of random node selection and re-allocation continues until all the nodes within the current module have been relaxed. In our preliminary tests, we have found that $N_{relaxed} = 60$ provided the best compromise in calculation time and quality of solution for networks of different sizes. The full proposed model is summarised in Algorithm 1.

Algorithm 1: Our proposed algorithm DiMod for detecting modules in directed networks.

```
Data: Directed network
Result: The solution at the end of the final iteration is taken as the final network division
Q_best = 0;
for N_runs ← 1 to N_MINLP do
    Solve MINLP model maximising modularity (Q);
    if Q > Q_best then
        Q_best ← Q;
        Y_nm^best ← Y_nm;
    end
end
for N_runs ← 1 to N_MIP do
    for each non-empty module do
        repeat
            Select N_relaxed nodes ;
            Remove node-module allocations (Y_nm) for the selected nodes ;
            Solve MIP model maximising Q, determining the community memberships of the
            relaxed nodes;
        until all nodes in the current module have been selected;
    end
end
```

Compared to solving one large MIP model that directly optimises the community memberships of all nodes simultaneously, solving a series of reduced models has the advantage of reaching global optimality for each reduced problem at a small computational cost. In the next section, a number of directed networks are used to demonstrate the applicability and efficiency of the proposed community detection algorithm.

3. Results

3.1. Synthetic Networks

The performance of our proposed community detection method is tested against other established methods in literature using synthetic networks generated by the Lancichinetti-Fortunato-Radicchi (LFR) benchmark [28]. We have generated seven non-overlapping directed unweighted networks with the following parameters: 500 nodes; default minus exponent for degree sequence and community size distribution, $t1 = 2$ and $t2 = 1$, respectively; maximum in-degree $maxk = 50$; and minimum and maximum number of nodes per community, $minc = 20$ and $maxc = 100$, respectively. Each of the networks had a different mixing parameter ($\mu \in \{0.1, 0.2, \ldots, 0.7\}$), where larger μ indicates more connections across different communities. Overall, the collection of synthetic networks used allows validating the efficiency of our algorithm in networks with different properties of community structure.

We compare our results to three popular algorithms that optimise modularity for directed networks: extremal optimisation [11], fast algorithm [10] and Tabu search [29], as implemented in Radatools [27]. We have executed each algorithm individually 100 times or until they reach a time limit of 24 h, and the best network division is reported for comparison. In terms of the proposed community detection approach, we solve 10 MINLPs ($N_{MINLP} = 10$), we perform 100 iterations on the MIP step of the algorithm ($N_{MIP} = 100$) and 60 nodes are relaxed for each solve of the MIP model ($N_{relaxed} = 60$). We implemented the models in GAMS [30], we use SBB [31] to solve the MINLP models and CPLEX [32] to solve the MIP models, and the optimality gap is set to 0, i.e., global optimum is achieved if the solver is allowed to run for unlimited time. The CPU time limit for each model was set to 1500 s. The deterministic nature of the proposed approach means only one single execution is required.

Figure 1 shows how the solution compares to the true community structure using the Normalized Mutual Information metric (NMI) [33]. We note that DiMod uncovers the true network communities even in larger mixing parameters. The Extremal Optimisation algorithm performs relatively well, while Fast algorithm and Tabu search fail to identify the ground truth communities before $\mu = 0.5$.

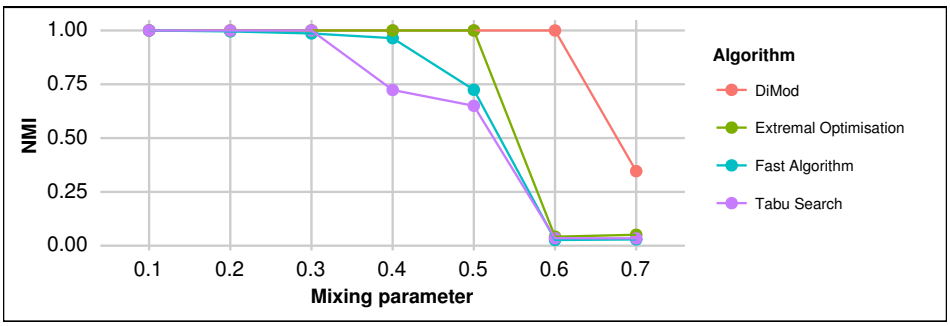

Figure 1. Performance of algorithms on synthetic directed networks.

3.2. Real Networks

We demonstrate the performance of our proposed method in tests of five real networks. Table 1 shows a summary of the network properties. We include the directed and weighted neural network of *Caenorhabditis elegans* [34]. Networks for *Mycobacterium tuberculosis* and *Plasmodium falciparum*, represent biological pathway interactions as extracted from the Reactome database (May 2014) [35]. The network for Roget's Thesaurus details cross-references between categories of English words, where one edge points from one category to another if a reference is provided to the latter among the words of the former [36]. In order to show that DiMod can solve larger problems, we include a snapshot of the Gnutella peer-to-peer network, gnutella08, obtained from the Stanford Large Network Dataset Collection (SNAP) [37].

Table 1. Summary of networks used as benchmarks in this study.

	Nodes	Edges	Type of Network
Mycobacterium tuberculosis	194	849	unweighted
Caenorhabditis elegans	297	2345	weighted
Roget's thesaurus	994	5058	unweighted
Plasmodium falciparum	1390	6497	unweighted
gnutella08	6301	20,777	unweighted

Our tests have shown that the second model (MIP) improves the quality of network division obtained by the MINLP model in the first stage. Table 2 below outlines the improvement in modularity between the more coarse initial network division provided by the MINLP and that of the final solution identified by the MIP step. Roget's Thesaurus is the network benefiting the most from the iterative procedure, where modularity is improved by about 16%. The iterative algorithm also boosts modularity by nearly 10% for *Mycobacterium tuberculosis* while improvements of around 4% can be observed in *C. elegans* and *Plasmodium falciparum* networks. The gnutella08 network also has an improvement of approximately 8% from the first to the second step and Figure 2 shows how modularity is improved at the end of each MIP iteration.

Table 2. Modularity improvement achieved by second step of the proposed method over the initial division network given by the MINLP.

	Myc. tub.	*C. elegans*	Roget	*P. falc.*	gnutella08
Initial modularity	0.4636	0.4877	0.5063	0.6978	0.4333
Final modularity	0.5073	0.5076	0.5860	0.7238	0.4678
Improvement	9.43%	4.08%	15.75%	3.72%	7.97%
Number of modules	9	5	13	20	24

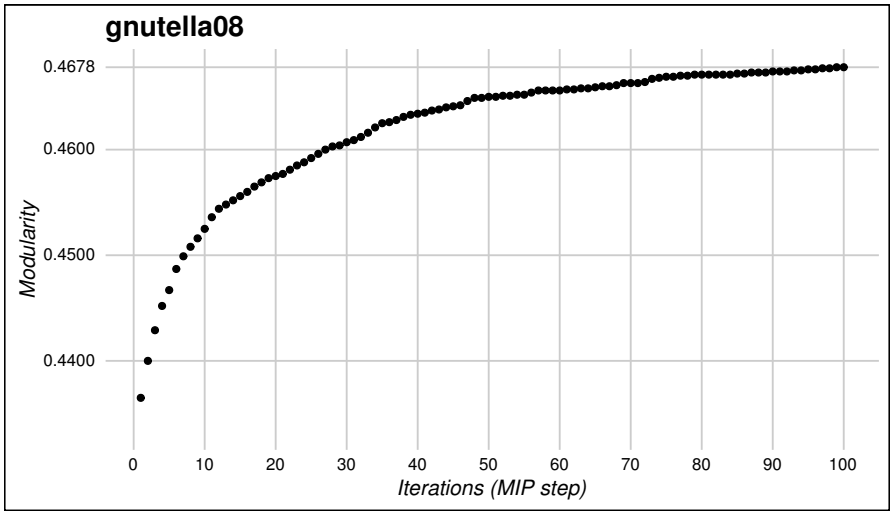

Figure 2. Improvement of modularity at the end of each MIP iteration for gnutella08 network.

As for synthetic networks, results obtained through DiMod for real networks are compared against modularity maximisation approaches in the literature (extremal optimisation [11], fast algorithm [10] and Tabu search [29]). Overall, the methodology proposed here consistently outperforms other methods, as shown in Table 3, which summarises our computational results.

Table 3. Comparison of different community detection methods on real networks.

	Myc. tub.	*C. elegans*	**Roget**	*P. falc.*	**gnutella08**
Extremal	0.4802	0.4731	0.5582	0.6685	0.2475
Fast algorithm	0.4567	0.5058	0.5002	0.6846	0.4624
Tabu search	0.4635	0.4438	0.5021	0.6496	0.2281
DiMod	**0.5073**	**0.5076**	**0.5860**	**0.7238**	**0.4678**

4. Conclusions

This works reports novel mathematical programming formulation to address the problem of community structure detection in directed networks. While modularity optimisation has been extensively studied for undirected networks, there is little research effort to detect modules in directed networks. A mathematical programming-based optimisation approach has been introduced to fill the gap in literature. Modularity optimisation for a directed network can be conveniently formulated as an MINLP model, which can converge to locally optimal solutions quickly even in large networks. The MINLP model is reformulated as an MIP model by re-writing non-linear terms into linear equivalents, which can then be solved to obtain globally optimal solutions for small networks. The novel community detection method proposed consists of two major steps, taking advantage of both models. Firstly, the MINLP model is solved to produce an initial coarse network division. Given the initial network division, the iterative algorithm works by repeatedly removing the community memberships of random sets of nodes, solving the reduced MILP model and re-allocating the relaxed nodes to communities.

Using synthetic and real-world directed networks covering a wide range of node and edge sizes, the proposed iterative algorithm considerably improves the quality of the initial coarse network partition. Compared with three community detection methods in literature, the proposed approach is demonstrated to identify the best network division consistently, yielding the largest modularity value. Another advantage of the proposed approach is its deterministic nature, which means multiple executions are likely to converge into the same network division.

Finally, we note that owing to the nature of the mathematical programming framework used here, modelling can be flexible enough to allow additional constraints and parameters to be easily implemented according to user requirements [38]. Prior knowledge on a particular system can be incorporated, for example in the form of nodes with similar functional annotations that may be constrained to be allocated in the same community [39]. In future work, we plan to study how the density of nodes and distributions of in-degree and out-degree affect the hardness and calculation time of the optimisation problem. We also plan to modify current integer programming models so as to detect and prevent the resolution limit.

Acknowledgments: Lingjian Yang acknowledges funding from the UK Engineering and Physical Sciences Research Council (EPSRC) Centre for Innovative Manufacturing in Emergent Macromolecular Therapies (EP/I033270/1) and Jonathan C. Silva acknowledges funding from Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brasília, Brazil (process number 13312138). Funding from the EU (to Sophia Tsoka, HEALTH-F2-2011-261366), the Leverhulme Trust (to Sophia Tsoka and Lazaros G. Papageorgiou, RPG- 2012-686) and the UK Engineering & Physical Sciences Research Council (to Lazaros G. Papageorgiou, EPSRC Centre for Innovative Manufacturing in Emergent Macromolecular Therapies) is gratefully acknowledged. We gratefully acknowledge Aristotelis Kittas for data preparation.

Author Contributions: Lingjian Yang, Jonathan C. Silva, Lazaros G. Papageorgiou and Sophia Tsoka conceived and designed the experiments. Lingjian Yang and Lazaros G. Papageorgiou designed the computational model. Lingjian Yang and Jonathan C. Silva performed the experiments. Lingjian Yang and Jonathan C. Silva gathered network data. Lingjian Yang, Jonathan C. Silva, Lazaros G. Papageorgiou and Sophia Tsoka analysed the data. Lingjian Yang, Jonathan C. Silva, Lazaros G. Papageorgiou and Sophia Tsoka wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MINLP	Mixed Integer Non-Linear Programming
MIP	Mixed Integer Linear Programming

References

1. Boccaletti, S.; Latora, V.; Moreno, Y.; Chavez, M.; Hwang, D. Complex networks: Structure and dynamics. *Phys. Rep.* **2006**, *424*, 175–308.
2. Fortunato, S. Community detection in graphs. *Phys. Rep.* **2010**, *486*, 75–174.
3. Newman, M.E.J. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA* **2006**, *103*, 8577–8582.
4. Brandes, U.; Delling, D.; Gaertler, M.; Gorke, R.; Hoefer, M.; Nikoloski, Z.; Wagner, D. On Modularity Clustering. *IEEE Trans. Knowl. Data Eng.* **2008**, *20*, 172–188.
5. Lancichinetti, A.; Fortunato, S. Limits of modularity maximization in community detection. *Phys. Rev. E* **2011**, *84*, 066122.
6. Good, B.H.; De Montjoye, Y.A.; Clauset, A. Performance of modularity maximization in practical contexts. *Phys. Rev. E* **2010**, *81*, 046106.
7. Fortunato, S.; Hric, D. Community Detection in Networks: A User Guide. *Phys. Rep.* **2016**, doi:10.1016/j.physrep.2016.09.002.
8. Danon, L.; Díaz-Guilera, A.; Duch, J.; Arenas, A. Comparing community structure identification. *J. Stat. Mech. Theory Exp.* **2005**, *2005*, P09008.
9. Medus, A.; Acuña, G.; Dorso, C. Detection of community structures in networks via global optimization. *Phys. A Stat. Mech. Appl.* **2005**, *358*, 593–604.
10. Newman, M.E.J. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* **2004**, *69*, 066133.
11. Duch, J.; Arenas, A. Community detection in complex networks using extremal optimization. *Phys. Rev. E* **2005**, *72*, 027104.
12. Aloise, D.; Cafieri, S.; Caporossi, G.; Hansen, P.; Perron, S.; Liberti, L. Column generation algorithms for exact modularity maximization in networks. *Phys. Rev. E* **2010**, *82*, 1–9.
13. Xu, G.; Bennett, L.; Papageorgiou, L.G.; Tsoka, S. Module detection in complex networks using integer optimisation. *Algorithms Mol. Biol.* **2010**, doi:10.1186/1748-7188-5-36.
14. Bennett, L.; Liu, S.; Papageorgiou, L.G.; Tsoka, S. Detection of Disjoint and Overlapping Modules in Weighted Complex Networks. *Adv. Complex Syst.* **2012**, *15*, doi:10.1142/S0219525911500238.
15. Cafieri, S.; Hansen, P.; Liberti, L. Improving heuristics for network modularity maximization using an exact algorithm. *Discret. Appl. Math.* **2014**, *163*, 65–72.
16. Bennett, L.; Kittas, A.; Muirhead, G.; Papageorgiou, L.G.; Tsoka, S. Detection of Composite Communities in Multiplex Biological Networks. *Sci. Rep.* **2015**, *5*, 1–12, doi:10.1038/srep10345.
17. Silva, J.C.; Bennett, L.; Papageorgiou, L.G.; Tsoka, S. A mathematical programming approach for sequential clustering of dynamic networks. *Eur. Phys. J. B* **2016**, doi:10.1140/epjb/e2015-60656-5.
18. Dourisboure, Y.; Geraci, F.; Pellegrini, M. Extraction and Classification of Dense Communities in the Web. In Proceedings of the 16th International Conference on World Wide Web, Banff, AB, Canada, 8–12 May 2007; pp. 461–470.
19. Liu, Y.; Moser, J.; Aviyente, S. Network community structure detection for directional neural networks inferred from multichannel multisubject EEG data. *IEEE Trans. Biomed. Eng.* **2014**, *61*, 1919–1930.
20. Guimerà, R.; Nunes Amaral, L.A. Functional cartography of complex metabolic networks. *Nature* **2005**, *433*, 895–900.
21. Lai, D.; Lu, H.; Nardini, C. Extracting weights from edge directions to find communities in directed networks. *J. Stat. Mech. Theory Exp.* **2010**, *2010*, P06003.
22. Satuluri, V.; Parthasarathy, S. Symmetrizations for Clustering Directed Graphs. In Proceedings of the 14th International Conference on Extending Database Technology, Uppsala, Sweden, 21–24 March 2011; pp. 343–354.

23. Zheng, Q.; Skillicorn, D.B. Spectral embedding of directed networks. *Soc. Netw. Anal. Min.* **2016**, *6*, 1–15.
24. Ning, X.; Liu, Z.; Zhang, S. Local community extraction in directed networks. *Phys. A Stat. Mech. Appl.* **2016**, *452*, 258–265.
25. Malliaros, F.D.; Vazirgiannis, M. Clustering and community detection in directed networks: A survey. *Phys. Rep.* **2013**, *533*, 95–142.
26. Leicht, E.A.; Newman, M.E.J. Community structure in directed networks. *Phys. Rev. Lett.* **2008**, *100*, 1–4.
27. Gomez, S. Radatools—Communities Detection in Complex Networks and Other tools. Available online: http://deim.urv.cat/~sergio.gomez/radatools.php (accessed on 20 July 2016).
28. Lancichinetti, A.; Fortunato, S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E* **2009**, *80*, 016118.
29. Arenas, A.; Fernández, A.; Gómez, S. Analysis of the structure of complex networks at different resolution levels. *New J. Phys.* **2008**, *10*, 53039.
30. Rosenthal, R.E. *GAMS—A User's Guide*; GAMS Development Corporation: Washington, DC, USA, 2016.
31. Bussieck, M.R.; Drud, A. SBB: A New Solver for Mixed Integer Nonlinear Programming. Available online: http://ww.atlatec.-port.gams.com/presentations/present_sbb.pdf (accessed on 26 October 2016).
32. IBM CPLEX Optimizer—United States. Available online: https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/ (accessed on 27 September 2016).
33. Esquivel, A.V.; Rosvall, M. Comparing network covers using mutual information. *arXiv* **2012**, arXiv:1202.0425.
34. Watts, D.J.; Strogatz, S.H. Collective dynamics of "small-world" networks. *Nature* **1998**, *393*, 440–442.
35. Kittas, A.; Bennett, L.; Hermjakob, H.; Tsoka, S. Organizational principles of the Reactome human BioPAX model using graph theory methods. *J. Complex Netw.* **2016**, doi:10.1093/comnet/cnw003.
36. Batagelj, V. Pajek Data: Roget's Thesaurus, 1879. Available online: http://vlado.fmf.uni-lj.si/pub/networks/data/dic/roget/Roget.htm (accessed on 20 July 2016).
37. Leskovec, J.; Krevl, A. SNAP Datasets: Stanford Large Network Dataset Collection. Available online: http://snap.stanford.edu/data (accessed on 26 October 2016).
38. Xu, G.; Tsoka, S.; Papageorgiou, L.G. Finding community structures in complex networks using mixed integer optimisation. *Eur. Phys. J. B* **2007**, *60*, 231–239, doi:10.1140/epjb/e2007-00331-0.
39. Bennett, L.; Kittas, A.; Liu, S.; Papageorgiou, L.G.; Tsoka, S. Community Structure Detection for Overlapping Modules through Mathematical Programming in Protein Interaction Networks. *PLoS ONE* **2014**, *9*, e112821, doi:10.1371/journal.pone.0112821.