Exact Signed Modularity Density Maximization Solutions and Their Real Meaning*

Rafael de Santiago
Department of Computer Science and Statistics
Federal University of Santa Catarina
Florianópolis, Brazil
r.santiago@ufsc.br

Luís C. Lamb
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
lamb@inf.ufrgs.br

***Abstract*—In the context of social media analyses, signed social networks are important representations of relationships between entities. These networks can represent positive or negative relationships between every two entities of a social network. One of the usual analyses over social networks is the communities/clustering search. In this paper, we report analyses on the Signed Modularity Density Maximization problem, which searches for meaningful clusters inside a signed social network. In this analysis, we present a new branch-and-price algorithm to find optimal solutions to this problem. The optimal solutions of two real signed networks and some artificial networks are generated and analyzed. The results suggest that the problem finds meaningful clusterings in signed social networks, and some parameters are suggested to obtain the best results.**

***Index Terms*—signed social networks, clustering, modularity density maximization**

I. Introduction

Social network analyses are useful to extract information for several reasons. They can, e.g., be used to identify relationship patterns in the networks. Social media have contributed to this area due to a large amount of data available, allowing one to test new models and methods in relationship analyses. Although the rise of social networks helped in this kind of analyses, their nodes and links are simple representations that do not detail the richness of complex relationships [1].

Signed social networks try to capture more of the real meaning of relationships of social networks. In these signed networks, each link has a positive or negative value. A positive value represents a friendship-like relationship [1]–[4]. A negative value represents a dislike relationship. In social networks like Facebook or Twitter, these negative links can be collected by the analysis of posts and replies.

Community detection problems in the context of social networks are computational problems that try to find the clustering of such networks such that the clusters group nodes which share common features [5], [6]. These problems are applied in several fields, e.g.: (*i*) in astronomy, for automatic stellar cluster recognition [7]; (*ii*) in biology, for finding protein complexes [8] and mapping metabolic reactions [9]; (*iii*) in the health sciences, to identify functional memory involved in olfactory recognition [10]; (*iv*) in social sciences, to identify criminal organizations [11], [12].

This work is partly supported by the Brazilian Research Council CNPq (MCTIC/CNPq 2018 - 408771/2018-6) and Federal University of Santa Catarina.

One of the most popular community detection problems is Newman's Modularity Maximization (MM) [13]. MM defines an optimization problem that has an objective function that measures the difference between the number of internal links and the expected number of links inside of each cluster. The original problem is applied to undirected, unsigned, and unweighted networks. Versions of the Modularity Maximization objective function for weighted and directed graphs are reported in [14] and [15], respectively. [16] reported a model for signed networks.

Although Modularity Maximization is a popular problem, some degeneracies are reported in the literature. The most important of them is named "resolution limit" proved by [17] in which cliques with a different number of nodes are merged into the optimal solution, even if they are connected by a single edge. This behavior is a degeneracy for Modularity Maximization because the modular property is violated for optimal partitions. Two other degeneracies are reported by [18]: there is an exponential number of suboptimal solutions, and the number of nodes has a positive correlation with the optimal modularity value. Efforts to avoid these degeneracies are made by reformulating the objective function of Modularity Maximization in, e.g. [19]–[27].

One of the main alternative problems to Newman's Modularity Maximization is the Modularity Density Maximization. Defined by [21], this problem uses the difference between the internal and external density of links for each cluster to evaluate the problem solutions. This value is normalized by the number of nodes inside of each cluster. This problem has attracted recent interest in science [28]–[38].

The problem version of Modularity Density Maximization for signed networks is presented in [16], but an evaluation around exact solutions are missed. In their paper, the solutions are obtained by non-exact solvers. In this context, this paper presents an exact solver for the Signed Modularity Density Problem and discusses the results over the obtained exact solutions. Our solver is computational expensive in time. In fact, no exact polynomial solver for the problem is known. Our proposed solver is a branch-and-price procedure.

The following sections of this paper present the Signed

Modularity Density optimization problem, the column generation model, the branch-and-price model, analyses on the exact solutions, and conclusions.

II. Signed Modularity Density Maximization

The objective function of the Signed Modularity Density Maximization problem is shown in Equation (1), given a signed graph G and a clustering C [16]. Consider $G = (V, E^+, E^-)$, where V is the set of nodes, E^+ is the set of positive edges, and E^- is the set of negative edges. In this context, we consider that $w^+_{uv} = 1$ if $\{u, v\} \in E^+$, otherwise $w^+_{uv} = 0$, and $w^-_{uv} = 1$ if $\{u, v\} \in E^-$, otherwise $w^-_{uv} = 0$. The degrees of each node are also divided into positive and negative ones. So, the positive degree of v is given by $d^+_v = \sum_{u\in V} w^+_{uv}$, and the negative degree is given by $d^-_v = \sum_{u\in V} w^-_{uv}$. The sets E^+_c and E^-_c are composed the edges positive and negative respectively of a cluster $c \in C$.

$$D_\lambda(C) = \sum_{c\in C} \left(\frac{2\lambda|E^+_c| - 2(1-\lambda)(\sum_{v\in c} d^+_v - |E^+_c|)}{|c|} \right. \\ \left. \frac{-2(1-\lambda)|E^-_c| + 2\lambda(\sum_{v\in c} d^-_v - |E^-_c|)}{|c|} \right). \tag{1}$$

The parameter λ for quantified Signed Modularity Density Maximization is used to obtain the "ratio association" to find small clusters when $\lambda > 0.5$, and the "ratio cut" to find large clusters when $\lambda < 0.5$. [21] suggested that this function can be used to find the appropriate level of the topological structure of graphs to find proper partitions.

In this problem, each node is assigned to one cluster. This kind of clustering is called non-overlapping clustering. So, the number of possible solutions is given by the Bell number (Equation (2)) over the number of nodes of G. Thus, there is an exponential number of possible solutions.

$$B_n = \begin{cases} 1, & \text{if } n \in \{0, 1\} \\ \sum_{k=0}^{n-1} \left[\binom{n-1}{k} B_k \right], & \text{if } n > 1 \end{cases} \tag{2}$$

It is not known an exact algorithm for the non-Signed and Signed Modularity Density Maximization problem. For the non-signed version, the exact solving is a binary nonlinear problem (0–1 NLP). Li et al. [21] have presented exact binary nonlinear (0-1 NLP) mathematical programming which makes it difficult to use with common solvers. Karimi-Majd et al. [30] have shown improvement to the [21] model, where the parameter of the number of clusters is not required. Costa [32] has created four different mixed integer linear models converted from the 0-1 NLP. Recent column generation algorithms have been showing exact methods that solve instances with more than 100 nodes [35], [37]. No exact algorithm for the Signed Modularity Density Maximization is known, so this paper presents the first one.

For the non-signed version of the problem, heuristics for Modularity Density Maximization have been reported in [28], [29], and [30]. They have presented some evolutionary heuristics. Costa et al. [33] have presented eight divisive heuristics for Modularity Density Maximization that have provided solutions close to the optimal one. These divisive heuristics are based on mathematical programming formulations. Recently, constructive and multilevel heuristics have been presented at [36] and [38]. For the signed version, Li et al. [16] have proposed evolutionary and memetic heuristics to find solutions to the problem.

III. Exact Method

We have developed a branch-and-price method to find exact solutions for the Signed Modularity Density Maximization problem. Due to the exponential number of variables of the exact problem, the branch-and-price method uses a column generation procedure to discover important variables. Then the integer values are found for them. This section describes details around the algorithm used.

A. Column Generation Model

Our column generation method is derived from the integer linear model related in Equation (3), that is inspired on previous models relating similar non-signed problems [37], [39], [40]. In this model, each cluster is a subset of nodes, so there are $|T| = 2^{|V|}$ possible clusters, where $T = \left\{1, \ldots, 2^{|V|}\right\}$. The variables z_t are binary. If $z_t = 0$, the cluster t does not belong to the solution; when $z_t = 1$, the cluster t belongs to the solution. The value of c_t is the contribution of cluster t to the objective function. This value is defined in Equation (4). The value a_{vt} is a binary number. If $a_{vt} = 1$, the node $v \in V$ belongs to the cluster t, and when $a_{vt} = 0$, the node v does not belong to this cluster. To obtain the dual problem, we have relaxed this model by redefining the Constraint (3c) to $z_t \geq 0, \forall t \in T$. In column generation contexts, we named this model as the master model.

$$\textbf{maximize} \left\{ \sum_{t\in T} c_t z_t \right\} \tag{3a}$$

$$\textbf{subject to:} \sum_{t\in T} a_{vt} z_t = 1, \forall v \in V \tag{3b}$$

$$z_t \in \{0, 1\}, \forall t \in T. \tag{3c}$$

$$\begin{aligned} c_t = & \frac{4\lambda \sum_{u,v\in V: u<v} a_{ut} a_{vt} w^+_{uv}}{\sum_{v\in V} a_{vt}} \\ & + \frac{-2(1-\lambda)(\sum_{v\in V} d^+_v a_{vt} - 2\sum_{u,v\in V:u<v} a_{ut} a_{vt} w^+_{uv})}{\sum_{v\in V} a_{vt}} \\ & + \frac{-4(1-\lambda)\sum_{u,v\in V:u<v} a_{ut} a_{vt} w^-_{uv}}{\sum_{v\in V} a_{vt}} \\ & + \frac{+2\lambda(\sum_{v\in V} d^-_v a_{vt} - 2\sum_{u,v\in V:u<v} a_{ut} a_{vt} w^-_{uv})}{\sum_{v\in V} a_{vt}}. \end{aligned}$$

The auxiliary model is described in Equations (4). The variable y_v is the dual variable related to the v constraint of the master model.

$$\textbf{minimize}\Big\{2(1-\lambda)\sum_{v\in V} d_v^+ a_v - 2\sum_{u\in V}\sum_{v\in V} w_{uv}^+ a_u a_v - 2\lambda\sum_{v\in V} d_v^- a_v + 2\sum_{u\in V}\sum_{v\in V} w_{uv}^- a_u a_v + \sum_{u\in V}\sum_{v\in V} a_u a_v y_v\Big\} \tag{4a}$$

$$\textbf{subject to: } a_v \in \{0,1\}, \forall v \in V. \tag{4b}$$

B. The Branch-and-Price Method

The column generation finds the optimal solution in continuous solution space. For this reason, we have used a branch-and-price method to find the optimal solution in integer solution space, related to the z_t variables. The branch-and-price method is reported in Algorithm 1.

The branch-and-price algorithm starts by defining the reduced master problem. The initial variables are defined by a two-step local search method of [38] adapted for signed networks. Then the heap that stores the highest D value in brach-and-price method is defined. The *lowerbound* variable stores the objective value from the best solution found. At each iteration, the branch-and-price node with the highest D value is obtained by the heap. If the solution of this node has a worse D value than the *lowerbound* value, its sibling nodes will not find a better solution, and the branch-and-price stops. If the search does not stop, the selected node is submitted to the column generation procedure.

The column generation procedure starts adding the branch-and-price node constraints to the *RM*. At each iteration of the column generation, a new column is added in the model by using the *Auxiliary* model. During the selection of a new column, the *RM* model is solved, and the values for the primal variables *z*, the dual variables *y*, and the objective value *Dens* are stored. Then the *Auxiliary* model is solved, and if a new variable is found, it is inserted into the *RM* model. After the loop, if the solution found is integer, the *lowerbound* variable is updated or not. If not, it is created two branch-and-price nodes. The variable z that has the largest fraction value has its value fixed in 0 for a node and 1 for the other node.

Algorithm 1 Branch-and-price

```
1: create the reduced master model RM with initial variables
   given by HLSMD$_\lambda\pm$
2: heap = new FibonacciHeap()
3: // constructing the starting node of branch-and-price
4: heap.push(($D = -\infty, cons = \{\}$))
5: lowerBound = $-\infty$
6: stop = false
7: while (not stop) do
8:     data = heap.pop()
9:     if (data.$D <$ lowerBound ) then
10:        stop = true
11:    end if
12:    if (not stop) then
13:        columnGeneration (RM, data, lowerbound)
14:    end if
15:    if heap.empty() then
16:        stop = true
17:    end if
18: end while
```

IV. Experiments on Real Signed Networks

To test our exact method and the results of Signed Modularity Density in real signed networks, we have used two benchmark signed networks which have been tested in [16], [41]. The first signed network is known as "Slovene Parliamentary Party Network" and is related to the relation among ten political parties of the Slovene Parliamentary in 1994 [42]. The second signed network is known as "Gahuku-Gama Subtribes Network" and represents the relationship of tribes from New Guinea [43]. The positive and negative relationships of these networks have been represented without considering weights in this paper.

Table I shows the runtime (in seconds) and the D value obtained for each exact solution found by our branch-and-price method. Each quantitative parameter λ was tested. We have used the solver IBM ILOG CPLEX and C++ language to implement our branch-and-price. The environment has been Linux, processor Intel i7-7500U, 2.7Ghz and 16GB RAM.

TABLE I
Runtime in seconds and D values related to the execution of the branch-and-price method on the two real instances.

	Slovene Parliamentary Party		Gahuku-Gama Subtribes	
λ	Time(s)	D*	Time(s)	D*
0.1	0.051	2.000	0.032	3.037
0.2	0.056	5.800	0.061	7.445
0.3	0.035	11.200	0.159	11.854
0.4	0.016	16.600	0.286	17.076
0.5	0.019	23.000	0.400	24.238
0.6	0.057	36.000	0.427	36.520
0.7	0.050	53.999	0.305	55.749
0.8	0.052	72.000	0.185	75.500
0.9	0.056	89.999	0.209	95.466

Figure 1 shows the optimal clustering obtained by our algorithm with different values of λ factor for each tested real clustering. The clusters are identified by color, except for clusters with a single node. In this latter case, the nodes are filled with white. The edges are identified by two colors. Blue colored edges are positive, and the red ones are negative.

For the "Slovene Parliamentary Party" network, with $\lambda \in \{0.1, 0.2, 0.3, 0.4\}$ the network was divided into two clusters connected only by negative edges. The only internal negative edges are {"ZS-ESS", "SNS"} and {"DS", "SNS"}. With $\lambda = 0.5$, 'ZS-ESS", "SNS", and "DS" are separated from their community when $\lambda \leq 0.4$. With a larger λ factor, smaller clusters are identified, so a node needs being more connected with a cluster to belong to it. When $\lambda \geq 0.6$, each node belongs to its own cluster (alone).

For the "Gahuku-Gama Subtribes" network, the clustering is composed of three clusters with no negative edge connecting two nodes that belong to the same cluster when $\lambda \in \{0.1, 0.2\}$. The difference between this clustering and the clustering obtained with $\lambda = 0.3$ is that the node "GEHAN" passed to belong the "red" cluster in the last partition. Comparing the clustering $\lambda = 0.3$, the nodes "NOTOH", "UHETO", and "KOHIK" form a cluster composed of themselves in the clustering $\lambda = 0.4$. Passing from $\lambda = 0.4$ to $\lambda = 0.5$, the nodes "NAGAD" and "GAMA" compose two singleton clusters, the nodes "NAGAN" and "SEUVE" form their proper cluster, "NAGAN" and "SEUVE" form their own cluster. From the clustering $\lambda = 0.5$ to 0.6, the nodes "ASARO", "GEHAN", and "NOTOH" form two singleton clusters. For the clusterings obtained with $\lambda \in \{0.7, 0.8\}$, the unique non-singleton cluster is composed of nodes "ALIKA", "MASIL", "OVE", and "UKUDZ" which are in the same cluster for all $\lambda \leq 0.8$. When $\lambda = 0.9$, there are singleton nodes only.

Comparing the obtained clusterings of Figures 1, the $\lambda \leq 0.5$ appears to be more usable because of the almost complete absence of singleton clusters. Observing the clustering formation and the positive and negative edges, the results suggest that the Signed Modularity Density Maximization obtains coherent clusterings, avoiding to keep together nodes that are connected by negative edges.

V. Experiments on Artificial Networks

We have performed experiments in artificial benchmarks to test scalability and influence of negative weights in the search algorithm. We would like to understand how much runtime is required to solve instances and share these results for further comparisons in the field. Moreover, we believe that negative relationships inside modules turn them harder to be detected. So, we would like to understand if this belief makes sense.

We have created an artificial instance creator inspired by [44]. This software uses a uniform distribution to create instances with the following parameters:

- The number of expected clusters (k);
- The number of nodes (n);
- The set of proportions of nodes each cluster have ($\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$ in which $\sum_{i \in k} \omega_i = 1$);
- The proportion of negative (Π^-) and positive (Π^+) weighted edges inside each cluster;
- The proportion of negative (Θ^-) and positive (Θ^+) weighted inter-edges.

For the following experiments, we have created instances with parameters $n \in \{30, 40, 50\}$, $k = \lceil \log_2 n \rceil$, $\omega_i = \frac{n}{k}$ for all $i \in \{1, 2, \ldots, k\}$, $\Pi^+ = 0.75$, $\Theta^+ = 0.1$, $\Pi^- \in \{0.01, 0.05, 0.1, 0.2\}$, and $\Theta^- = 0.075$. We have tested these instances with $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ factors for objective function.

The experiments were run on a PC with an Intel i7-7500U, 2.7Ghz and 16GB RAM over Linux Ubuntu 16.04.1 LTS operating system. Each experiment was done by using a single thread. The language used to program the column generations was C++, with the "GCC" compiler. The solver IBM ILOG CPLEX 12.6 [45] was used to solve the mathematical programming model.

Table II shows the ϕ correlation value obtained to find the optimal solution during the experiments. The value "tl" means that all results for that method and instance reached the time limit of ten hours. For the ground truth analyses, we used the *Matthews Correlation Coefficient* that takes account of true positives (N_{11}), true negatives (N_{00}), false negatives (N_{10}), and false positives (N_{01}) [46]. The ϕ function is described in Equation (5). The resulting values of the ϕ function are bound between $[-1, 1]$. The larger the coefficient ϕ is, the stronger the correlation between the partition obtained by the heuristic and the correct partition is.

$$\phi = \frac{N_{00}N_{11} - N_{10}N_{01}}{\sqrt{(N_{11} + N_{01})(N_{11} + N_{10})(N_{00} + N_{01})(N_{00} + N_{10})}}. \tag{5}$$

Each N has the number of pairs of nodes, comparing the correct partitions and the one obtained by the tested methods. They are described as follow:

- N_{00}: the number of pairs of nodes which are not in the same cluster in the expected partition, and which are not in the same cluster in the obtained partition by our tested method;
- N_{01}: the number of pairs of nodes which are not in the same cluster in the expected partition, but which are in the same cluster in the obtained partition by our tested method;
- N_{10}: the number of pairs of nodes which are in the same cluster in the expected partition, but which are not in the same cluster in the obtained partition by our tested method;
- N_{11}: the number of pairs of nodes which are in the same cluster in the expected partition, and which are in the same cluster in the obtained partition by our tested method.

It is possible to observe three different aspects of the results in Table II. The first one is about the number of nodes. For the instances with $\Pi^- \in \{0.05, 0.1, 0.2\}$, one can see the positive correlation between the number of nodes and the ϕ values. It suggests that the model has presented difficulty in capturing the expected clustering in tiny networks. The second aspect

TABLE II
ϕ CORRELATION VALUES OBTAINED TO FIND THE OPTIMAL SOLUTION FOR EACH EXPERIMENTED INSTANCE AND PARAMETER. THE VALUE "TL" MEANS THAT THE RESULT FOR THAT INSTANCE REACHED THE TIME LIMIT OF TEN HOURS. THE BEST RESULTS ARE MARKED IN BOLD.

		Π^-			
Nodes	λ	**0.01**	**0.05**	**0.10**	**0.20**
30	**0.1**	0.0000	0.0000	0.0000	0.0000
	0.3	0.0000	0.0000	0.0000	0.0000
	0.5	0.4260	**0.4260**	0.5170	0.5170
	0.7	**0.4754**	*tl*	**0.5928**	**0.5319**
	0.9	0.1683	0.1683	0.2748	0.2934
40	**0.1**	0.0000	0.0000	0.0000	0.0000
	0.3	0.0000	0.0000	0.0000	0.0000
	0.5	**0.8788**	**0.8788**	0.5527	**0.6919**
	0.7	0.3695	0.3695	**0.6376**	0.3489
	0.9	0.0286	0.0286	0.0514	0.0260
50	**0.1**	0.0000	0.0000	0.0000	0.0000
	0.3	0.0000	0.0000	0.0000	0.0000
	0.5	*tl*	**0.9484**	*tl*	**0.7163**
	0.7	**0.6027**	0.6027	**0.6085**	0.5286
	0.9	0.0738	0.0738	0.1594	0.0686

TABLE III
RUNTIME THAT HAS BEEN REQUIRED TO FIND THE OPTIMAL SOLUTION FOR EACH EXPERIMENTED INSTANCE AND PARAMETER. THE VALUE "TL" MEANS THAT THE RESULT FOR THAT INSTANCE REACHED THE TIME LIMIT OF TEN HOURS. THE VALUES ARE IN SECONDS.

		Π^-			
Nodes	λ	**0.01**	**0.05**	**0.10**	**0.20**
30	**0.1**	0.0095	0.0094	0.0095	0.0206
	0.3	0.0636	0.0720	0.0579	0.0571
	0.5	0.9005	0.8586	0.5619	0.2374
	0.7	36.0004	*tl*	0.4001	0.3620
	0.9	0.0654	0.0657	0.0250	0.0322
40	**0.1**	0.0088	0.0093	0.0052	0.0099
	0.3	0.0239	0.0259	0.0661	0.0446
	0.5	4.4276	4.4503	11.0482	3.6149
	0.7	2.8841	2.8816	2.7542	2.7541
	0.9	0.2966	0.2938	0.1890	0.2260
50	**0.1**	0.1287	0.1238	0.1313	0.1143
	0.3	0.6812	0.6800	1.1254	1.7397
	0.5	*tl*	36.0051	*tl*	36.1318
	0.7	14.3484	14.3953	9.3605	10.4980
	0.9	0.4374	0.3797	0.4325	0.3319

is that the best values are related to the λ values between 0.5 and 0.7. The third one is about the internal proportion of negative weighted edges. Our experiments suggests that the model have captured a good correlation for $\Pi^- = 0.05$ considering the instances with $n \in \{40, 50\}$. Considering our experiments, the best correlations have been obtained while using $\lambda \in \{0.5, 0.7\}$ for networks with 40 and 50 nodes, and an internal proportion of 5% of negative weighted edges.

Table III shows the runtime in seconds obtained to find the optimal solution during the experiments. The value "tl" means that all results for that method and instance reached the time limit of ten hours. It is possible to see that the higher is the number of nodes, the larger is the runtime required to find the optimal solution. We cannot find in these results a behavior connecting the Π^- and the runtime required. Considering the best correlation values reported in Table II, the best values were obtained in 37 seconds.

VI. Conclusions

In this paper, we have aimed at solving exactly the Signed Modularity Density Maximization problem and to analyze the optimal solutions in real signed networks. In order to do so, we have proposed a branch-and-price algorithm, detailing its column generation procedure to solve the problem. We have also analyzed the optimal solutions with their real representations over different quantitative λ parameters to understand how far they are from the real meaning of the clusterings found.

Our experiments suggest that for $\lambda \leq 0.5$, the clustering obtained by the exact solving preserve important classification properties and can be used to identify meaningful clusters of the signed graphs. For $\lambda > 0.5$, several nodes were classified as singleton nodes, rendering difficult the search for important clusters.

In the experiments with artificial networks, the best correlations are obtained while using $\lambda \in \{0.5, 0.7\}$ for networks with 40 and 50 nodes, and internal proportion of 5% of negative weighted edges. We also have identified that the higher is the number of nodes, the larger is the runtime required to find the optimal solution as expected.

As further research, we suggest amplifying these analyses for several clustering problems in social media contexts. We also suggest that new exact approaches can be compared with our branch-and-price results.

Acknowledgments

This work is partly supported by the Brazilian Research Council CNPq (MCTIC/CNPq 2018 - 408771/2018-6) and Federal University of Santa Catarina. The authors would like to thank IBM for the Academic Initiative program that offered the IBM ILOG CPLEX for our experiments.

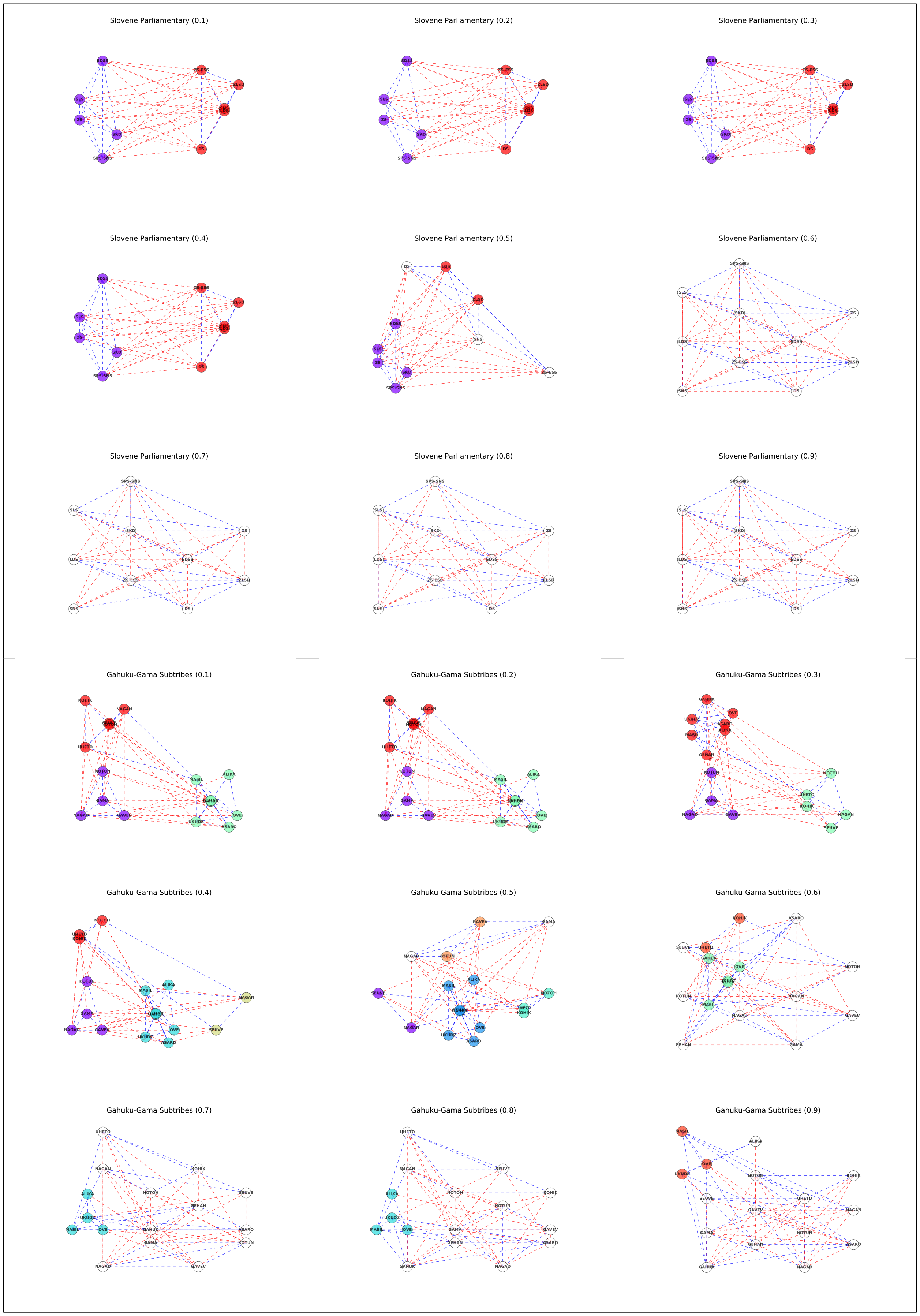
Slovene Parliamentary (0.1)
Slovene Parliamentary (0.2)
Slovene Parliamentary (0.3)
Slovene Parliamentary (0.4)
Slovene Parliamentary (0.5)
Slovene Parliamentary (0.6)
Slovene Parliamentary (0.7)
Slovene Parliamentary (0.8)
Slovene Parliamentary (0.9)
Gahuku-Gama Subtribes (0.1)
Gahuku-Gama Subtribes (0.2)
Gahuku-Gama Subtribes (0.3)
Gahuku-Gama Subtribes (0.4)
Gahuku-Gama Subtribes (0.5)
Gahuku-Gama Subtribes (0.6)
Gahuku-Gama Subtribes (0.7)
Gahuku-Gama Subtribes (0.8)
Gahuku-Gama Subtribes (0.9)

Fig. 1. Our branch-and-price method has obtained the clustering structures presented in this figure for ‘Slovene Parliamentary Party” and “Gahuku-Gama Subtribes” networks.

References

[1] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*. New York, New York, USA: ACM Press, 2010, p. 1361.

[2] ——, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th international conference on World wide web - WWW '10*. New York, New York, USA: ACM Press, 2010, p. 641.

[3] D. Centola, "The Spread of Behavior in an Online Social Network Experiment," *Science*, vol. 329, no. 5996, pp. 1194–1197, sep 2010. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/20813952 http://www.sciencemag.org/cgi/doi/10.1126/science.1185231

[4] P. Wang, B. Xu, Y. Wu, and X. Zhou, "Link prediction in social networks: the state-of-the-art," *Science China Information Sciences*, vol. 58, no. 1, pp. 1–38, jan 2015.

[5] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 9, pp. 2658–2663, 2004.

[6] S. Fortunato and C. Castellano, "Community Structure in Graphs," in *Computational Complexity*, R. A. Meyers, Ed. New York, NY: Springer New York, 2012, pp. 490–512.

[7] S. Schmeja, "Identifying star clusters in a field: a comparison of different algorithms," *Astronomische Nachrichten*, vol. 332, no. 2, pp. 172–184, Feb. 2011.

[8] T. Nepusz, H. Yu, and A. Paccanaro, "Detecting overlapping protein complexes in protein-protein interaction networks," *Nature Methods*, vol. 9, no. 5, pp. 471–472, may 2012.

[9] R. Guimera and L. Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, no. February, pp. 895–900, 2005.

[10] D. Meunier, P. Fonlupt, A.-L. Saive, J. Plailly, N. Ravel, and J.-P. Royet, "Modular structure of functional networks in olfactory memory." *NeuroImage*, vol. 95, pp. 264–75, Jul. 2014.

[11] E. Ferrara, P. De Meo, S. Catanese, and G. Fiumara, "Detecting criminal organizations in mobile phone networks," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5733–5750, Oct. 2014.

[12] F. Calderoni, D. Brunetto, and C. Piccardi, "Communities in criminal networks: a case study," *Social Networks*, vol. 48, pp. 116–125, jan. 2017.

[13] M. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, p. 026113, Feb. 2004.

[14] M. E. J. Newman, "Analysis of weighted networks," *Physical Review E*, vol. 70, no. 5, p. 056131, nov 2004.

[15] E. Leicht and M. Newman, "Community structure in directed networks," *Physical Review Letters*, vol. 100, no. 11, p. 118703, Mar. 2008.

[16] Y. Li, J. Liu, and C. Liu, "A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks," *Soft Computing*, vol. 18, no. 2, pp. 329–348, Feb. 2014.

[17] S. Fortunato and M. Barthélemy, "Resolution limit in community detection." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 1, pp. 36–41, Jan. 2007.

[18] B. H. Good, Y.-A. de Montjoye, and A. Clauset, "Performance of modularity maximization in practical contexts," *Physical Review E*, vol. 81, no. 4, p. 046106, Apr. 2010.

[19] S. Muff, F. Rao, and A. Caflisch, "Local modularity measure for network clusterizations," *Physical Review E*, vol. 72, no. 5, p. 056107, Nov. 2005.

[20] A. Arenas, A. Fernandez, and S. Gomez, "Analysis of the structure of complex networks at different resolution levels," *New Journal of Physics*, vol. 10, p. 053039, 2008.

[21] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, "Quantitative function for community detection," *Physical Review E*, vol. 77, no. 3, p. 036109, Mar. 2008.

[22] S. Cafieri, P. Hansen, and L. Liberti, "Loops and multiple edges in modularity maximization of networks." *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 81, no. 4 Pt 2, p. 046102, 2010.

[23] V. A. Traag, P. Van Dooren, and Y. Nesterov, "Narrow scope for resolution-limit-free community detection," *Physical Review E*, vol. 84, no. 1, p. 016114, Jul. 2011.

[24] C. Granell, S. Gómez, and A. Arenas, "Hierarchical multiresolution method to overcome the resolution limit in complex networks," *International Journal of Bifurcation and Chaos*, vol. 22, no. 07, pp. 1 250 171-1 – 1 250 171-7, Jul. 2012.

[25] T. Chakraborty, S. Srinivasan, N. Ganguly, S. Bhowmick, and A. Mukherjee, "Constant communities in complex networks." *Scientific reports*, vol. 3, p. 1825, 2013.

[26] M. Chen, T. Nguyen, and B. K. Szymanski, "On Measuring the Quality of a Network Community Structure," in *International Conference on Social Computing (SocialCom), 2013*. Alexandria: IEEE, 2013, pp. 122–127.

[27] M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE Transactions on Computational Social Systems*, vol. 1, no. 1, pp. 46–65, Mar. 2014.

[28] J. Liu and J. Zeng, "Community detection based on modularity density and genetic algorithm," in *Proceedings of International Conference on Computational Aspects of Social Networks*, Taiyuan, 2010, pp. 29–32.

[29] M. Gong, Q. Cai, Y. Li, and J. Ma, "An improved memetic algorithm for community detection in complex networks," in *2012 IEEE Congress on Evolutionary Computation*. Brisbane, QLD: IEEE, 2012, pp. 1–8.

[30] A.-M. Karimi-Majd, M. Fathian, and B. Amiri, "A hybrid artificial immune network for detecting communities in complex networks," *Computing*, vol. 97, no. 5, pp. 483–507, 2014.

[31] Z. Li, R.-S. Wang, S. Zhang, and X. Zhang, "Quantitative function and algorithm for community detection in bipartite networks," *American Journal of Operations Research*, vol. 5, pp. 421–434, jan 2015.

[32] A. Costa, "MILP formulations for the modularity density maximization problem," *European Journal of Operational Research*, vol. 245, no. 1, pp. 14–21, 2015.

[33] A. Costa, S. Kushnarev, L. Liberti, and Z. Sun, "Divisive heuristic for modularity density maximization," *Computers & Operations Research*, vol. 71, pp. 100–109, 2016.

[34] Y. Izunaga, T. Matsui, and Y. Yamamoto, "A doubly nonnegative relaxation for modularity density maximization," University of Tsukuba, Tsukuba, Tech. Rep. 1339, 2016. [Online]. Available: http://www.optimization-online.org/DB_HTML/2016/03/5368.html

[35] K. Sato and Y. Izunaga, "An enhanced MILP-based branch-and-price approach to modularity density maximization on graphs," pp. 1–18, may 2018.

[36] R. Santiago and L. C. Lamb, "Efficient modularity density heuristics for large graphs," *European Journal of Operational Research*, vol. 258, no. 3, pp. 844–865, may 2017.

[37] R. de Santiago and L. C. Lamb, "Exact computational solution of Modularity Density Maximization by effective column generation," *Computers & Operations Research*, vol. 86, pp. 18–29, oct 2017. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0305054817301041

[38] R. Santiago and L. C. Lamb, "Efficient Quantitative Heuristics for Graph Clustering," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. Berlin: ACM New York, 2017, pp. 117–118.

[39] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti, "Column generation algorithms for exact modularity maximization in networks," *Physical Review E*, vol. 82, no. 4, p. 046112, Oct. 2010.

[40] R. de Santiago and L. C. Lamb, "A ground truth contest between modularity maximization and modularity density maximization," *Artificial Intelligence Review*, Jan 2020. [Online]. Available: https://doi.org/10.1007/s10462-019-09802-8

[41] B. Yang, W. K. Cheung, and J. Liu, "Community mining from signed social networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 10, pp. 1333–1348, 2007.

[42] S. Kropivnik and A. Mrvar, "An Analysis of the Slovene Parliamentary Parties Network," *Developments in Statistics and Methodology, Metodološki zvezki*, vol. 12, pp. 209–216, 1996.

[43] K. E. Read, "Cultures of the Central Highlands, New Guinea," *Southwestern Journal of Anthropology*, vol. 10, no. 1, pp. 1–43, apr 1954. [Online]. Available: http://www.journals.uchicago.edu/doi/10.1086/soutjanth.10.1.3629074

[44] B. Yang, X. Liu, Y. Li, and X. Zhao, "Stochastic Blockmodeling and Variational Bayes Learning for Signed Network Analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 9, pp. 2026–2039, sep 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7917265/

[45] IBM, *IBM ILOG CPLEX Optimization Studio V12.6.3 documentation*. IBM, 2015.

[46] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *BBA - Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.