Efficient Solution of the Correlation Clustering Problem: An Application to Structural Balance

Lúcia Drummond[2], Rosa Figueiredo[1], Yuri Frota[2], and Mário Levorato[3]

[1] CIDMA, Department of Mathematics, University of Aveiro
3810-193 Aveiro, Portugal
rosa.figueiredo@ua.pt
[2] Department of Computer Science, Fluminense Federal University
24210-240 Niterói-RJ, Brazil
{lucia,yuri}@ic.uff.br
[3] Petróleo Brasileiro S.A.
levorato@petrobras.com.br

Abstract. One challenge for social network researchers is to evaluate balance in a social network. The degree of balance in a social group can be used as a tool to study whether and how this group evolves to a possible balanced state. The solution of clustering problems defined on signed graphs can be used as a criterion to measure the degree of balance in social networks. By considering the original definition of the structural balance, the optimal solution of the Correlation Clustering (CC) Problem arises as one possible measure. In this work, we contribute to the efficient solution of the CC problem by developing sequential and parallel GRASP metaheuristics. Then, by using our GRASP algorithms, we solve the problem of measuring the structural balance of large social networks.

1 Introduction

Signed graphs were introduced by Heider [1] with the purpose of describing sentiment relations between people pertaining to a same social group and to provide a systematic statement of social balance theory. Cartwright *et al.* [2] formalized Heider's theory stating that a balanced social group could be partitioned into two mutually hostile subgroups each having internal solidarity. In the last decades, signed graphs have shown to be a very attractive discrete structure for social network researchers [3–8]. One challenge in this area is to evaluate balance in a social network. Different criteria and different solution approaches have been used in the literature as an attempt to quantify and evaluate balance in a signed social network [5, 8–10].

Clustering is the action of partitioning individual elements into groups based on their similarity. Clustering problems defined on signed graphs arise in many scientific areas [11–16]. The common element among these applications is the collaborative *vs.* conflicting environment in which they are defined. The solution of clustering problems defined on signed graphs can be used as a criteria to measure the degree of balance in social networks [4, 5, 17]. By considering the original definition [1] of structural balance, the optimal solution of the very known Correlation Clustering (CC) Problem arises as a measure for the degree

Y.T. Demey and H. Panetto (Eds.): OTM 2013 Workshops, LNCS 8186, pp. 674–683, 2013.

of balance in a social network. Alternative measures to the structural balance and the clustering problems associated with them were recently discussed [5, 17].

To the best of our knowledge, the CC problem was addressed for the first time in [4] (not under this name) where its heuristic solution was used as a criteria for analyzing structural balance in social networks. The heuristic approach proposed by the authors is a simple greedy neighborhood search procedure that assumes a prior knowledge of the number of clusters in the solution. This heuristic is implemented in software Pajek [18]. Lately, motivated by the solution of a document clustering problem the unweighted version of the CC problem was formalized in [13]. The weighted version of the problem was addressed in [19]. The CC problem has been largely investigated from the point of view of constant factor approximation algorithms and has been applied in the solution of many applications (see references in [20]). A comparison of several heuristic strategies (greedy and local search methods) for this problem is presented in [20] and applied to document clustering and natural language processing. In [7] the CC problem is called *community mining* and an agent-based heuristic is proposed to its solution. An approach based on genetic algorithm has been proposed in [21] for the CC problem and applied to documents clustering.

From a practical point of view, in solving the clustering problem treated in this paper, heuristic approaches are primarily of interest since large social networks may have to be analyzed [8, 9, 22]. The definition of a measure to represent the balance/imbalance of a social network involves itself a degree of approximation for the task of evaluating balance in a social network. Thus, it is imperative that the clustering problem associated with this measure be solved efficiently. To the best of our knowledge, [21] presents the only metaheuristic approach applied to the CC problem. However, the Greedy Randomized Adaptive Search Procedure (GRASP) [23] has been successfully applied to clustering problems [24–26] and to a related problem [27]. Moreover, numerical experiments described in [20] indicate that combining greedy heuristics with local search procedures is an efficient strategy to the solution of the CC problem.

Our contributions are two-fold. At first, we contribute to the efficient solution of the CC problem by developing sequential and parallel GRASP metaheuristics for this problem. Then, by using our GRASP algorithms we solve the problem of measuring the structural balance of a large social network.

2 Problem Definition and Mathematical Formulation

Let $G = (V, E)$ be an undirected graph where V is the set of n vertices and E is the set of edges. In this text, a graph is assumed to have no loops. For a vertex set $S \subseteq V$, let $E[S] = \{(i, j) \in E \mid i, j \in S\}$ denote the *subset of edges induced by* S. For two vertex sets $S, W \subseteq V$, let $E[S : W] = \{(i, j) \in E \mid i \in S, j \in W\}$. One observes that, by definition, $E[S : S] = E[S]$. Consider a function $s : E \to \{+, -\}$ that assigns a sign to each edge in E. An undirected graph G together with a function s is called a *signed graph*. An edge $e \in E$ is called *negative* if $s(e) = -$ and *positive* if $s(e) = +$. Let E^- and E^+ denote, respectively, the set of negative and positive edges in a signed graph.

A *partition* of V is a division of V into non-overlapping and non-empty subsets. Consider a partition $P = \{S_1, S_2, \ldots, S_l\}$ of V. The *cut edges* and the *uncut edges* related with this partition are defined, respectively, as the edges in sets $\cup_{1 \le i < j \le l} E[S_i : S_j]$ and $\cup_{1 \le i \le l} E[S_i]$. Let w_e be a nonnegative edge weight associated with edge $e \in E$. Also, for $1 \le i, j \le l$, let

$$\Omega^+(S_i, S_j) = \sum_{e \in E^+ \cap E[S_i:S_j]} w_e \text{ and } \Omega^-(S_i, S_j) = \sum_{e \in E^- \cap E[S_i:S_j]} w_e.$$

The *imbalance* $I(P)$ of a partition P is defined as the total weight of negative uncut edges and positive cut edges, i.e.,

$$I(P) = \sum_{1 \le i \le l} \Omega^-(S_i, S_i) + \sum_{1 \le i < j \le l} \Omega^+(S_i, S_j). \tag{1}$$

Likewise, the *balance* $B(P)$ of a partition P can be defined as the total weight of positive uncut edges and negative cut edges. Clearly, $B(P) + I(P) = \sum_{e \in E} w_e$.

Problem 21 (CC problem). *Let $G = (V, E, s)$ be a signed graph and w_e be a nonnegative edge weight associated with each edge $e \in E$. The correlation clustering problem is the problem of finding a partition P of V such that the imbalance $I(P)$ is minimized or, equivalently, the balance $B(P)$ is maximized.*

The classical formulation for the CC problem is an integer linear programming (ILP) model proposed to uncapacitated clustering problems (see references in [28]). In this formulation a binary decision variable x_{ij} is assigned to each pair of vertices $i, j \in V$, $i \ne j$, and defined as follows: $x_{ij} = 0$ if i and j are in a common set; $x_{ij} = 1$ otherwise. The model minimizes the total imbalance.

$$\text{minimize} \sum_{(i,j) \in E^-} w_{ij}(1 - x_{ij}) + \sum_{(i,j) \in E^+} w_{ij} x_{ij} \tag{2}$$

$$\text{subject to } x_{ip} + x_{pj} \ge x_{ij}, \qquad \forall\, i, p, j \in V, \tag{3}$$

$$x_{ij} = x_{ji}, \qquad \forall\, i, j \in V, \tag{4}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall\, i, j \in V. \tag{5}$$

The triangle inequalities (3) say that if i and p are in a same cluster as well as p and j, then vertices i and j are also in a same cluster. Constraint (4) written to $i, j \in V$ establishes that variables x_{ij} and x_{ji} assume always the same value in this formulation. Constraints (5) impose binary restrictions to the variables while the objective function (2) minimizes the total imbalance defined by equation (1). Notice that, according to constraints (4), half of the variables can be eliminated which reduces both the number of variables and the number of constraints.

A set partitioning formulation [28] is proposed in the literature to uncapacitated clustering problems and could also be used in the solution of the CC problem. As we can expect, these two formulations are not appropriate solution approaches when time limit is a constraint in the solution process. The authors in [17] reports that the classical formulation starts to fail (time limit set to 1h) with random instances of 40 vertices and negative density equal to 0.5.

3 Sequential and Parallel GRASP to CC Problem

Metaheuristics have been used successfully for solving hard combinatorial optimization problems as they can provide sub-optimal solutions in a reasonable time. GRASP is a multi-start metaheuristic in which each iteration consists basically of two phases: construction and local search [29].

In this work, we develop a GRASP method to the solution of the CC problem. The proposed heuristic (*GraspCC*) is based on a GRASP method for the maximum modularity problem [26]. It is composed of a construction phase *coCC* and a local search phase *lsCC*. In phase *coCC*, a randomized greed function is used to build up an initial feasible solution. Let $P = \{S_1, S_2, ..., S_l\}$ denote a *partial partition* (i.e., a partition of a proper subset of V) and let $M_{n\times n} = (m_{ij})$ be the *modularity signed matrix* [26]. We define below a *function* $f : (V \setminus \bigcup_{1\leq k\leq l} S_k) \to \mathbb{R}$, which will measure the impact of inserting a vertex i in the partial partition P.

$$f(i) = \max\left(m_{ii}, m_{ii} + \max_{1\leq k\leq l} \sum_{j\in S_k} 2m_{ij}\right). \tag{6}$$

Our *GraspCC* is described in Algorithm 1. The parameter *iter* denotes the maximum number of iterations without improvement in the best solution found. The first task in every iteration of *GraspCC* is to construct a solution in a greedy randomized fashion. This task is performed in phase *coCC* described in Algorithm 2. In this phase, the ordered set L_f is defined (line 3) as the set of vertices $V \setminus \bigcup_{1\leq k\leq l} S_k$ ordered in decreasing order of function f. At each iteration, in lines 4-8, we choose a vertex i randomly among the first $\lfloor \alpha . \mid L_f \mid \rfloor$ vertices in this set and add it to the partial partition. Note that parameter α defines the degree of randomness the construction phase will have. This process is repeated until a partition of V is obtained.

Algorithm 1. *GraspCC*

```
1  Input: G = (V, E) and α
2  Output: partition P*
3  P* = ∅;  I(P*) = ∞;  i = 0;
4  while i ⩽ iter
5     P = coCC(G, α);
6     P = lsCC(P, G);
7     if (I(P) < I(P*))
8        P* = P;  i = 0;
9     end if
10    i = i + 1
11 end while
12 return P*;
```

There is no guarantee that the construction method returns a locally optimal solution with respect to some neighborhood. Therefore, the solution P obtained

in phase *coCC* may be improved by the local search procedure *lsCC* described in Algorithm 3. The neighborhood $N_r(P)$ is defined as the family of all partitions obtained by moving r vertices in P from one cluster into another. Note that the neighborhood analyzes partitions with different number of clusters (i.e. a vertex can be moved into a new cluster or can be removed from a single vertex cluster) The *lsCC* method starts with the partition provided by the construction phase. It iteratively replaces the current partition by that with minimum imbalance cost within its neighborhood (lines 4-8). The local search halts when no better partition is found in the neighborhood of the current solution. In this work we employed a neighborhood with $r \leq 2$.

Although metaheuristics aim to reduce historical difficulties of conventional construction and local search methods (such as premature stops in local optima solutions distant from an optimal solution) in optimization problems, they may require a large amount of time to find good primal bounds as penalty in many cases. This fact has motivated the development of a parallel metaheuristic, taking advantage of the inherent parallelism present in the GRASP method [30].

In this work, the well known independent approach was employed in the parallel program. This scheme limits communication between processors only for problem input, detection of process termination, and determination of best overall solution. When p processors are used, a single process reads the problem data and passes it to the remaining $p-1$ processes. Each process executes a copy of the GRASP program. Processes send a message to all others when they either stop upon finding a solution at least as good as the target value or complete the maximum number of allotted iterations.

Algorithm 2. *coCC*

```
1 Input: G = (V, E) and α
2 Output: partition P
3 P = ∅; L_f = Order(V);
4 while (L_f ≠ ∅)
5    Choose vertex i randomly among the first ⌊α. | L_f |⌋ elements of L_f
6    Update S_k = S_k ∪ {i} where k is the component in P that maximizes f
7    L_f = L_f − {i};  Re-order(L_f);
8 end while
9 return P
```

4 CC Solution Applied to Structural Balance and Conclusions

The algorithms described in the previous section were implemented in ANSI C and MPI(MPICH2) for message passing. All experiments were performed (with exclusive access) on a cluster with 42 nodes, each one with two processors Intel Xeon QuadCore 2.66Hz and 16Gb of RAM under Linux (Red Hat) 5.3 operating system. The formulation is coded in Xpress Mosel 3.2.0 with solver Xpress Optimizer 21.01.00. The CPU time limit is set to 1 hour for the ILP formulation.

Algorithm 3. *lsCC*

```
Input: G = (V, E) and a partition P
Output: partition P
while (P improving)
  for all P̄ ∈ (N1(P) ∪ N2(P))
    if (I(P̄) < I(P))
      P = P̄
    end if
  end for
end while
```

All heuristic outcomes are average results of 5 independent executions. Computational experiments were carried out on (i) a set of 28 social networks from the literature, (ii) a set of 6 network instances that represent the United Nations General Assembly (UNGA), and (iii) a set of 12 random instances. Next, we describe briefly these instances[1].

(i) This set of instances is composed by 22 small size instances normally used in blockmodeling approaches to structural balance (see [5, 17, 31]) and 6 signed networks extracted from the large scale social network representing the technology-related news website Slashdot (see [8, 9]). The small instances were used to parametrize the GRASP heuristics.

(ii) We generated 6 medium-sized social networks based on UNGA voting records of the separate annual sessions between 2003 and 2008[2]. These instances are weighted versions of UNGA signed graphs described in [27].

(iii) We generated random social networks with 50 vertices ($n = 50$) varying network density $d = 2 \times |E|/(n^2 - n)$ and negative graph density defined here as $d^- = |E^-|/|E|$. We considered a set of 12 random instances having d and d^- ranging, respectively, in sets $\{0.1, 0.2, 0.5, 0.8\}$ and $\{0.2, 0.5, 0.8\}$. These instances were also used in [17].

As reported in the literature [28], the linear relaxation of the ILP formulation described in Section 2 provides a very good representation of the problem which allows to find the optimal solution for many instances by solving this linear relaxation. Experiments reported in [17] confirm this assertion: the 22 small instances in set (i) were solved to optimality in some seconds; the formulation failed to solve the random instances in set (iii) having $d \geq 0.2$ and $d^- = 0.5$. In our experiments, the 6 instances in set (ii) were solved to optimality by the ILP formulation in the root of the branch and bound tree. The results obtained on these instances with the ILP formulation and with the sequential GRASP ($iter = 400$, $r \leq 2$,$\alpha = 0.8$, time limit set to 30 minutes) are reported in Table 1.

[1] All instances are available in `http://www.ic.uff.br/~yuri/files/CCinst.zip`.

[2] United Nations General Assembly Voting Data, by Anton Strezhnev and Erik Voeten, `http://hdl.handle.net/1902.1/12379`. Accessed in June 2013.

Table 1. Results obtained on UNGA instances with Xpress and the sequential GRASP algorithm (400 iterations without improvement). Number of vertices (n); number of negative ($|E^-|$) and positive ($|E^+|$) edges; sum of negative ($w(E^-)$) and positive ($w(E^+)$) weights; number of clusters (k) in the optimal solution $\bar{P}$; percentage imbalance ($\%I = 100 \times IP(\bar{P})/w(E)$); negative percentage imbalance ($\%I^- = 100 \times IP^-(\bar{P})/w(E^-)$) where $IP^-(\bar{P})$ is the sum of weights of negative edges in the optimal solution; positive percentage imbalance ($\%I^+ = 100 \times IP^+(\bar{P})/w(E^+)$) where $IP^+(\bar{P})$ is the sum of weights of positive edges in the optimal solution; and time in seconds spent by the Xpress software to solve the ILP formulation ($T(Xpress)$) and by the sequential GRASP procedure GraspSeq ($T(seq)$).

	Instance					Optimal Solution				Times (s)	
Session	n	$\|E^-\|$	$\|E^+\|$	$w(E^-)$	$w(E^+)$	k	$\%I$	$\%I^-$	$\%I^+$	$T(Xpress)$	$T(seq)$
2003	191	1119	16636	247.33	8247.60	2	0.09	0.47	0.08	281	1327
2004	191	945	17121	315.15	8646.75	2	0.23	2.74	0.14	301	1410
2005	192	1120	16566	251.28	8271.56	3	0.47	3.03	0.40	338	1783
2006	192	886	17378	248.60	8873.92	2	0.32	1.59	0.28	223	1739
2007	192	1109	17148	260.26	8646.04	2	0.51	4.18	0.40	262	1468
2008	192	1055	17091	256.66	8854.55	2	0.40	5.05	0.27	178	1342

Table 2. Results obtained on Slashdot signed graphs. BestSol is the value of the best solution found in the time limit.

	Instance				ILP Formulation		GraspSeq		GraspPar 8	
n	$\|E^-\|$	$\|E^+\|$	$w(E^-)$	$w(E^+)$	k	BestSol	k	BestSol	k	BestSol
200	62	825	62	825	1	65	12	48.4	12	49.0
300	82	981	82	981	1	85	23	58.8	25	61.8
400	87	1192	87	1192	1	87	23	64.6	23	64.6
600	156	1761	156	1761	–	–	29	121.4	31	121.8
800	371	2965	371	2965	–	–	50	257.4	54	254.5
1000	859	5132	859	5132	–	–	62	647.0	63	650.25

We can conclude that these signed networks are almost perfect balanced with most part of the imbalance given by negative relations.

The drawback of the ILP approach appears when we try to solve the instances based on Slashdot: the ILP formulation turn to be very big and Xpress is not able to solve any of it. To solve these instances with our heuristics, we propose two approaches. First we tackle the problem with the sequential GRASP with $iter = 400$, $r = 1$, $\alpha = 0.4$ and time limit of 1 hour. Then we solve these intances with our parallel GRASP by using 8 processors (GraspPar 8) with the same parameters but $iter = 50$. We discard the costly neighborhood with $r = 2$ and increase the time limit in order to manage the large number of vertices and edges. On the other hand, we drop the randomness of the method to avoid low quality solutions with (probably) a reduced number of iterations. These results are reported in Table 2. The notations in this table are the same as in Table 1 ("–" means no feasible integer solution was reported in the time limit).

Table 3. Results obtained on random instances ($n = 50$) with sequential and parallel GRASP algorithm

File	ILP BestSol	GraspSeq BestSol	GraspSeq $T(seq)$	GraspPar 8 BestSol	GraspPar 8 $T(8)$	$Su(8)$	$E(8)$
$d = 0.1$ $d^- = 0.2$	48	48.00	51.64	48.00	9.24	5.59	0.70
$d = 0.1$ $d^- = 0.5$	55	56.43	395.57	56.80	85.84	4.61	0.58
$d = 0.1$ $d^- = 0.8$	18	18.43	648.25	18.20	129.46	5.01	0.63
$d = 0.2$ $d^- = 0.2$	98	98.00	8.69	98.00	1.69	5.13	0.64
$d = 0.2$ $d^- = 0.5$	**159**	149.57	164.24	150.00	35.45	4.63	0.58
$d = 0.2$ $d^- = 0.8$	58	58.20	999.26	58.20	209.45	4.77	0.60
$d = 0.5$ $d^- = 0.2$	245	245.00	7.52	245.00	1.25	6.03	0.75
$d = 0.5$ $d^- = 0.5$	**523**	461.00	205.82	461.80	61.03	3.37	0.42
$d = 0.5$ $d^- = 0.8$	196	197.67	1800.61	197.20	519.89	3.46	0.43
$d = 0.8$ $d^- = 0.8$	392	392.00	7.52	392.00	1.18	6.36	0.80
$d = 0.8$ $d^- = 0.5$	**879**	775.29	228.47	775.60	51.15	4.47	0.56
$d = 0.8$ $d^- = 0.8$	334	335.00	1803.40	334.80	809.19	2.23	0.28
Avg	250.42	236.22	526.75	236.30	159.57	4.64	0.58

In [17], computational experiments are reported with Pajek [18] on the 22 small instances in (i) and on the random instances in (iii). We run our GRASP procedures on these instances and they performed equally or better than the greedy heuristic implemented in Pajek: strictly better in (i) 1 instance; and in (iii) 7 instances.

Table 3 compares the results obtained on the set of random instances (iii) by GraspSeq ($iter = 400$, $r \leq 2$ and $\alpha = 0.8$) and GraspPar 8 ($iter = 50$, $r \leq 2$ and $\alpha = 0.8$) with time limit set to 30 minutes. The instances that were not solved to optimality by the ILP formulation are marked with bold values in column (ILP-BestBound); in that case this column exhibits the value of the best integer solution found in the time limit. Clearly the GRASP algorithms achieved strong bounds, reaching an average imbalance of 236.22 (236.30 for the parallel version) while the average imbalance found by the exact method was 250.42. The parallel algorithm presents average speed-up and efficiency of 4.64 and 0.58 for 8 processors (one core per processor): speed-up measures the acceleration observed for the parallel algorithm when compared with its sequential version; efficiency measures the fraction of time along which each process is effectively used. Thus, $Su(8) = T(seq)/T(8)$, such that $T(8)$ is the time required for the parallel algorithm run on 8 processors, and $E(8) = Su(8)/8$. We did not obtain linear speedups due to the different stopping criteria employed in those algorithms. In the sequential version, the program finishes after 400 consecutive iterations without improvement while in the parallel algorithm every task must execute 50 iterations without improvement to finish. The sequential algorithm reached that condition proportionately faster than its parallel counterpart.

The obtained computational results indicate that our GRASP metaheuristic is an efficient approach for the heuristic solution of the CC problem. Additional numerical experiments need to be done with instances from other applications. The numerical experience over larger social networks indicates that,

in order to handle the enlarged instances like Epinions (131.828 vertices and 841.372 edges) or Slashdot (82.144 vertices and 549.202 edges) networks, we need to implement better parallelization strategies. As we have mentioned in the introduction, alternative measures have been proposed in the literature to the structural balance [5, 17]. The next steps of this research includes the adaptation of the GRASP heuristic to deal with these different measures.

Acknowledgements. Rosa Figueiredo is supported by FEDER founds through COMPETE-Operational Pro- gramme Factors of Competitiveness and by Portuguese founds through the CIDMA (University of Aveiro) and FCT, within project PEst-C/MAT/UI4106/2011 with COM- PETE number FCOMP-01-0124-FEDER-022690.

References

1. Heider, F.: Attitudes and cognitive organization. Journal of Psychology 21, 107–112 (1946)
2. Cartwright, D., Harary, F.: Structural balance: A generalization of heiders theory. Psychological Review 63, 277–293 (1956)
3. Abell, P., Ludwig, M.: Structural balance: a dynamic perspective. Journal of Mathematical Sociology 33, 129–155 (2009)
4. Doreian, P., Mrvar, A.: A partitioning approach to structural balance. Social Networks 18, 149–168 (1996)
5. Doreian, P., Mrvar, A.: Partitioning signed social networks. Social Networks 31, 1–11 (2009)
6. Inohara, T.: On conditions for a meeting not to reach a deadlock. Applied Mathematics and Computation 90, 1–9 (1998)
7. Yang, B., Cheung, W., Liu, J.: Community mining from signed social networks. IEEE Transactions on Knowledge and Data Engineering 19, 1333–1348 (2007)
8. Facchetti, G., Iacono, G., Altafini, C.: Computing global structural balance in large-scale signed social networks. Proceedings of the National Academy of Sciences of the United States of America 108, 20953–20958 (2011)
9. Leskovec, J., Huttenlocher, D., Kleinberg, J.: Signed networks in social media. In: CHI 2010 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1361–1370 (2010)
10. Srinivasan, A.: Local balancing influences global structure in social networks. Proceedings of the National Academy of Sciences of the United States of America 108, 1751–1752 (2011)
11. Huffner, F., Betzler, N., Niedermeier, R.: Separator-based data reduction for signed graph balancing. Journal of Combinatorial Optimization 20, 335–360 (2010)
12. DasGupta, B., Encisob, G.A., Sontag, E., Zhanga, Y.: Algorithmic and complexity results for decompositions of biological networks into monotone subsystems. BioSystems 90, 161–178 (2007)
13. Bansal, N., Blum, A., Chawla, S.: Correlation clustering. In: Proceedings of the 43rd Annual IEEE Symposium of Foundations of Computer Science, Vancouver, Canada, pp. 238–250 (2002)
14. Gülpinar, N., Gutin, G., Mitra, G., Zverovitch, A.: Extracting pure network submatrices in linear programs using signed graphs. Discrete Applied Mathematics 137, 359–372 (2004)

15. Macon, K., Mucha, P., Porter, M.: Community structure in the united nations general assembly. Physica A: Statistical Mechanics and its Applications 391, 343–361 (2012)
16. Traag, V., Bruggeman, J.: Community detection in networks with positive and negative links. Physical Review E 80, 36115 (2009)
17. Figueiredo, R., Moura, G.: Mixed integer programming formulations for clustering problems related to structural balance (2012) (Paper submitted)
18. Pajek, `http://pajek.imfm.si/` (accessed June 2013)
19. Demaine, E.D., Emanuel, D., Fiat, A., Immorlica, N.: Correlation clustering in general weighted graphs. Theoretical Computer Science 361, 172–187 (2006)
20. Elsner, M., Schudy, W.: Bounding and comparing methods for correlation clustering beyond ilp. In: ILP 2009 Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing, pp. 19–27 (2009)
21. Zhang, Z., Cheng, H., Chen, W., Zhang, S., Fang, Q.: Correlation clustering based on genetic algorithm for documents clustering. In: IEEE Congress on Evolutionary Computation, pp. 3193–3198 (2008)
22. Kunegis, J., Lommatzsch, A., Bauckhage, C.: The slashdot zoo: mining a social network with negative edges. In: WWW 2009 Proceedings of the 18th International Conference on World Wide Web, pp. 741–750 (2009)
23. Resende, M., Ribeiro, C.: Search Methodologies, 2nd edn. Springer (2011)
24. Nascimento, M., Toledo, F., de Carvalho, A.: Investigation of a new grasp-based clustering algorithm applied to biological data. Computers Operations Research 37, 1381–1388 (2010)
25. Frinhani, R., Silva, R., Mateus, G., Festa, P., Resende, M.: Grasp with path-relinking for data clustering: A case study for biological data. In: Pardalos, P.M., Rebennack, S. (eds.) SEA 2011. LNCS, vol. 6630, pp. 410–420. Springer, Heidelberg (2011)
26. Nascimento, M.C., Pitsoulis, L.: Community detection by modularity maximization using grasp with path relinking. Computers Operations Research (2013) (available online on March 2013)
27. Figueiredo, R., Frota, Y.: The maximum balanced subgraph of a signed graph: applications and solution approaches (2012) (paper submitted)
28. Mehrotraa, A., Trick, M.: Cliques and clustering: A combinatorial approach. Operations Research Letters 22, 1–12 (1998)
29. Resende, M., Ribeiro, C.: Grasp with path-relinking: Recent advances and applications. In: Ibaraki, T., Nonobe, K., Yagiura, M. (eds.) Metaheuristics: Progress as real problem solvers, pp. 29–63. Springer (2005)
30. Aiex, R.M., Binato, S., Resende, M.G.C.: Parallel grasp with path-relinking for job shop scheduling. Parallel Computing 29, 393–430 (2004)
31. Brusco, M.: An enhanced branch-and-bound algorithm for a partitioning problem. British Journal of Mathematical and Statistical Psychology 56, 83–92 (2003)