Evaluating balancing on social networks through the efficient solution of Correlation Clustering problems

Mario Levorato, Rosa Figueiredo, Yuri Y. Frota, Lúcia Drummond

Noname manuscript No.
(will be inserted by the editor)

Evaluating balancing on social networks through the efficient solution of Correlation Clustering problems

Mario Levorato · Rosa Figueiredo · Yuri Frota · Lúcia Drummond

Received: date / Accepted: date

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 and this measure can be obtained with the optimal solution of the Correlation Clustering (CC) problem, as well as a variation of it, the Relaxed Correlation Clustering (RCC) problem. However, solving these problems is no easy task, especially when large network instances need to be analyzed. In this work, we contribute to the efficient solution of both problems by developing sequential and parallel ILS metaheuristics. Then, by using our algorithms, we solve the problem of measuring the structural balance on large real-world social networks.

Keywords Correlation Clustering · Social Network · Structural Balance · ILS · VND

1 Introduction

Structural balance is recognized as a major social process which can be employed as a framework for studying the stability of social groups, their relationships and

M. Levorato
Department of Computer Science, Fluminense Federal University, 24210-240 Niterói-RJ, Brazil.
E-mail: mlevorato@ic.uff.br

R. Figueiredo
Laboratoire d'Informatique d'Avignon, University of Avignon, 84911 Avignon, France.
E-mail: rosa.figueiredo@univ-avignon.fr

Y. Frota
Department of Computer Science, Fluminense Federal University, 24210-240 Niterói-RJ, Brazil.
E-mail: yuri@ic.uff.br

L. Drummond
Department of Computer Science, Fluminense Federal University, 24210-240 Niterói-RJ, Brazil.
E-mail: lucia@ic.uff.br

their conflicts. According to structural balance theory, the balance or equilibrium of a social system seeks the human propensity to preserve a cognitive consistency between hostility and friendship. The supporting principle, as proposed by Heider (Heider 1946), is fairly simple: "my friend's friend is my friend, the enemy of a friend is my enemy, my enemy's enemy is my friend (or may become one), and so on". When balance is not present, tension is propagated in the minds of group members in a way that can ultimately lead to the altering of opinion.

Structural balance theory was first formulated by Heider (Heider 1946) with the purpose of describing antagonistic relationships between those people pertaining to the same social group (like/dislike, love/hate, respect/disrespect, or trust/distrust). Signed graphs were then introduced by Cartwright et al. (Cartwright and Harary 1956), which formalized Heiders theory and stated that a balanced social group could be partitioned into two groups (or clusters), being that all relationships (or edges) within clusters are positive (internal solidarity) and all those between clusters are negative (mutually hostile groups). Davis (Davis 1967) later introduced a notion called "weak balance", that further generalizes social balance with an assertion that a balanced social group can be divided into two or more mutually antagonistic subgroups (or clusters), each having internal solidarity.

Structural balance theory has a multitude of applications. Investigating the temporal dynamics of communities has attracted an increasing amount of attention, with models attempting to forecast how networks evolve over time. Known works involve the dynamics of signed structures and group formation (Doreian and Krackhardt 2001), including balance adjustment processes (Abell and Ludwig 2009). Structural balance is also potentially useful to study similarity and correlation networks, such as those determined by common voting patterns, or alliances and disputes (Traag and Bruggeman 2009, Macon et al. 2012).

In social network research, signed graphs have been widely used to represent antagonistic systems (Doreian and Mrvar 1996a, Inohara 1998, Yang et al. 2007, Abell and Ludwig 2009, Doreian and Mrvar 2009, Facchetti et al. 2011, Esmailian et al. 2014). Based on these structures, researchers are challenged to measure and evaluate balance of social networks. Numerous criteria and resolution approaches have been incorporated into the literature as an attempt to accomplish this task. However, most methodological research in this field is based on unsigned networks such as Facebook and Twitter (Duch and Arenas 2005, Newman 2006, Brandes et al. 2008), which contain only positive relationships (e.g. friend or trust) between users. For signed social networks, quantifying and evaluating balance is still difficult and problematic (Doreian and Mrvar 2009, Leskovec et al. 2010, Facchetti et al. 2011, Srinivasan 2011, Esmailian et al. 2014). Our contribution is to the evaluation of structural balance within these networks.

Clustering is the action of partitioning individual elements into groups (clusters) based on their similarity. Clustering problems on signed graphs appear in several scientific areas: efficient document classification (Bansal et al. 2002), detection of embedded matrix structures (Gülpinar et al. 2004), biological systems (Das-Gupta et al. 2007), community structure (Traag and Bruggeman 2009, Macon et al. 2012), and image segmentation (Kim et al. 2014). In particular, the clustering task in a signed network attempts to identify k antagonistic groups in the network, being that most entities within the same cluster are friends while most entities belonging to different clusters are enemies. Notice that, since this (weak)

balance definition solely applies to signed networks, most traditional clustering algorithms for unsigned networks cannot be directly applied.

The solution of clustering problems is an important criterion to measure the level of balance in signed social networks (Doreian and Mrvar 1996a, 2009, Figueiredo and Moura 2013). One such instance is the Correlation Clustering (CC) problem introduced by Bansal et al. (2002). In fact, minimizing the imbalance defined by structural balance is exactly equivalent to solving the CC problem. As stated in Bansal et al. (2002), it is the problem of partitioning n objects with the goal of minimizing the sum of negative elements that occur within clusters, plus the sum of positive elements that occur between clusters. The main idea is that, in a signed network, there are some links that create imbalance. The number of such links can be expressed as an amount of frustration. The CC objective function consists of minimizing that frustration. Alternative measures to structural balance and the associated clustering problems have been previously discussed as well (Doreian and Mrvar 2009, Figueiredo and Moura 2013). Our contribution focuses on structural balance evaluation using the CC and a relaxed version of this problem.

In a pragmatic approach, since large social networks[1] require analysis (Kunegis et al. 2009, Leskovec et al. 2010, Facchetti et al. 2011), clustering problems are primarily solved by the use of heuristic methods. For example, large-scale online networks with opposite types of relationships have gained in popularity. Slashdot (Slashdot 1997), a website that features news stories on science and technology, incorporates a feature allowing users to tag each other as friends or foes. On-line review websites such as Epinions (Epinions 1999) allow users to either like or dislike other people's reviews. The definition of a measure to represent the balance/imbalance of a social network adds to itself a degree of approximation to the task of evaluating balance in a social network. Thus, it is imperative that the clustering problem associated with this measure be solved efficiently. This is particularly challenging when, as in social networks retrieved from on-line media, the size of the community is very big, of the order of 10^5 individuals or higher.

As far as we know, there are only two metaheuristic approaches to solve the CC problem: Zhang et al. (2008) and Drummond et al. (2013). The first consisting of a genetic algorithm for the CC problem, applied to document clustering, while the latter presents a Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende 1995) implementation capable of efficiently solving the problem in networks of up to 8000 vertices. Since previous works on the CC problem have used greedy local search algorithms with success (VOTE/BOEM (Elsner and Schudy 2009) and Doreian Mrvar algorithm (Doreian and Mrvar 1996b)), we have chosen to focus our research in the development of local search algorithms with a greedy component. Moreover, population-based algorithms (like genetic algorithms) usually have a high computational cost, while local search algorithms with a greedy component (like GRASP and ILS approaches) are faster when solving combinatorial problems on huge instances. A neighborhood search heuristic for a related problem (clique partitioning), developed by Brusco and Köhn (2009), has confirmed our choice.

[1] We consider small networks those comprised of dozens of nodes, while medium-sized networks contain hundreds of elements and large-scale ones can have more than a hundred thousand nodes.

Based on the GRASP proposed by Drummond et al. (2013), we have contributed to the efficient solution of the CC problem by developing sequential and parallel Iterated Local Search (ILS) (Lourenço et al. 2003) metaheuristics. This work is an extention of the work presented in Levorato et al. (2015), which highlighted preliminary results obtained through initial experiments. These findings motivated the execution of several other tests in order to achieve a detailed analysis of the clustering problems discussed in this work, focused on the application perspective.

In this paper, we first present a thorough analysis of the ILS introduced in Levorato et al. (2015) in comparison with other solution approaches proposed in the literature. We give a detailed view of all aspects of the sequential and parallel ILS algorithms. We then extend the procedure to solve the Symmetric Relaxed Correlation Clustering (SRCC) problem, proposed by Doreian and Mrvar (2009) as an alternative measure of the structural balance of a social network. Roughly speaking, the SRCC problem is another version of the Correlation Clustering problem, with a costly objective function. Extending the ILS metaheuristic to solve the SRCC problem demanded not only a new objective function to evaluate the partition. The algorithm used additional data structures in order to reduce the computational effort needed to analyze the solutions visited at each iteration. Our work is the first one to solve the SRCC problem on large-scale real-world instances (Leskovec and Krevl 2014) and to conduct an analysis of the relative imbalance provided by the CC and SRCC problems.

As previously mentioned, structural balance theory affirms that human societies tend to avoid tension and conflicted relationships. In a signed graph that represents a real-world social network, this translates into a level of balance greater than expected, when compared to a random signed graph of equivalent size (Facchetti et al. 2011). Partial hints that real-world, currently available signed social networks are more balanced than expected are provided by Kunegis et al. (2009), Leskovec et al. (2010), Kunegis et al. (2010), Facchetti et al. (2011). Our analysis confirms that three well-known signed social networks (WikiElections, Epinions and Slashdot) are indeed extremely balanced, hence supporting previous related works.

We also present a historical and geopolitical analysis of the results obtained from the voting on resolutions in the United Nations General Assembly (UNGA), this based on the solutions for the CC and SRCC problems. Other works have also applied different signed network clustering methods to networks of international alliances and disputes (Traag and Bruggeman 2009, Macon et al. 2012).

2 The Correlation Clustering problem

Correlation Clustering (Bansal et al. 2002) is a clustering technique stemming from the problem of document clustering in which, when given a large corpus of documents such as web pages, one wants to group them into the optimal number of clusters, without specifying that number in advance. The basic problem consists of minimizing the number of unrelated pairs that are clustered together, in addition to the number of related pairs that are separate. In this section, we formally describe the CC problems and present a mathematical formulation, to be accompanied by a literature review.

2.1 The Correlation Clustering problem - Mathematical Formulation

Let $G = (V, E)$ be an undirected graph[2] where V is the set of n vertices and E is the set of edges. In this text, a signed graph is allowed to have parallel edges (i.e. two edges associated with a given pair of vertices) but no loops. Also, we assume that parallel edges always have opposite signs. 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 $E[S : W] = \{(i, j) \in E \mid (i \in S, j \in W) \vee (i \in W, j \in S)\}$. 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*, denoted by $G = (V, E, s)$. 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. The negative graph density is defined as $d^- = |E^-|/|E|$, while the positive graph density is $d^+ = |E^+|/|E|$.

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.,

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

That being said, we are ready to provide a formal definition to the CC problem.

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.*

The classical mathematical formulation for the CC problem is an integer linear programming (ILP) model proposed to uncapacitated clustering problems (which are also known as clique partitioning problems whenever the underlying graph is complete; see references in (Mehrotra and Trick 1998)). In this formulation, a binary decision variable x_{ij} is assigned to each pair of vertices $i, j \in V$, $i \neq j$, and defined as follows: $x_{ij} = 0$ if i and j are in a common set; $x_{ij} = 1$ otherwise. The

[2] It is possible to solve the CC problem on directed graphs. One must first convert the directed graph to an undirected one: opposite arcs with the same sign are converted to a single edge whose weight is equal to the sum of the arcs' weights; opposite arcs with different signs become two parallel edges, each one with the original sign and weight of each arc.

model minimizes the total imbalance.

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

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

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

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

Even though this formulation is polynomial-sized, having $n(n-1)$ variables and n^3+n^2 constraints, notice that, according to constraints (4), half of the variables can be eliminated, which reduces both the number of variables and constraints of the formulation.

A set partitioning formulation (Mehrotra and Trick 1998) proposed in the literature to uncapacitated clustering problems, 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 (Figueiredo and Moura 2013) report that the classical formulation begins to fail with random instances of 40 vertices and negative density (d^-) equal to 0.5.

2.1.1 Literature Review of the CC problem

To our knowledge, the CC problem (as defined above) was first addressed by Doreian et al. (Doreian and Mrvar 1996a) (although not under this name), who provided a heuristic solution method for analyzing structural balance in signed social networks. Their method, implemented in software Pajek (De Nooy et al. 2011), consists of a simple greedy neighborhood search procedure which requires the number of clusters in the desired solution. Having a document clustering problem in mind, Bansal et al.(Bansal et al. 2002) formalized the unweighted version of the CC problem and also discussed its NP-completeness proof, whereas the weighted version of the problem was addressed in (Demaine et al. 2006). Integer linear programming (ILP) can be used to solve the CC problem optimally, but only when the number of data points is small. Since it consists of a NP-hard minimization problem, the only available solutions for large instances are either heuristic or approximate. Predominately investigated from the viewpoint of constant factor approximation algorithms (Bansal et al. 2002, Swamy 2004, Charikara et al. 2005, Demaine et al. 2006, Giotis and Guruswami 2006, Ailon et al. 2008), the problem has been applied in several areas, such as portfolio analysis in risk management (Harary et al. 2003, Huffner et al. 2010), biological systems (DasGupta et al. 2007, Huffner et al. 2010), grouping of genes (Bhattacharya and De 2008), efficient document classification (Bansal et al. 2002), detection of embedded matrix structures (Gülpinar et al. 2004), image segmentation (Kim et al. 2014) and community structure (Traag and Bruggeman 2009, Macon et al. 2012).

Additionally, some authors have focused on the detection of overlapping communities in signed graphs (Zhang et al. 2007, Bonchi et al. 2011). In particular, (Bonchi et al. 2011) define an optimization problem that extends the framework of Correlation Clustering to allow overlaps.

An evaluation of different heuristic strategies (greedy and local search methods) for the CC problem is performed in (Elsner and Schudy 2009), using two practical

tasks: document clustering[3] and natural language processing (instances of $n = 1000$), to which ILP does not scale. In this situation, the authors' recommendation for CC Problem solution is the $VOTE/BOEM$ greedy algorithm, which rapidly obtains good objective values (with tight bounds). A greedy neighborhood-based heuristic for the problem has also been proposed by (Wang and Li 2013). Instead of stepping through the vertices according to a vertex permutation, their algorithm, known as Restoring, partitions vertices according to an edge permutation.

In (Yang et al. 2007), the CC problem is known as *community mining* and an agent-based heuristic called FEC is proposed to its solution. In order to assess the performance of FEC, the authors present a method for generating random signed networks with controlled community structures, based on a set of parameters. However, they only test their algorithm results against the Doreian Mrvar (DM) algorithm (Doreian and Mrvar 1996b), based on the clusters listed in each solution, and without presenting a numerical measure of imbalance. Also, according to the authors, the success of the FEC algorithm is based on its capacity of extracting the communities initially defined by the aforementioned generation routine.

Genetic algorithms have been applied to document clustering, using the CC problem as objective function (Zhang et al. 2008). Unfortunately, no explanation about the genetic operators is available, and we were unable to obtain the program code or generate the instances used in the experiments, making it difficult to understand and recreate the proposed procedure. Lately, we presented a Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende 1995) implementation that provides an efficient solution to the CC problem in networks of up to 8000 vertices (Drummond et al. 2013). Then, based on this method, we introduced sequential and parallel ILS (Iterated Local Search (Lourenço et al. 2003)) procedures for the CC problem (Levorato et al. 2015). In this work, besides providing a thorough analysis of ILS in comparison with other solution approaches, we focus on the application point of view, performing structural balance analysis on real-world signed social networks.

2.2 Relaxed Correlation Clustering: an alternative measure for structural balance

As we noted in the introduction, other measures of structural balance are proposed in the literature. In (Doreian and Mrvar 2009), the definition of a k-balanced signed graph was informally extended in order to include relevant processes (polarization, mediation, differential popularity and subgroup internal hostility) that originally were viewed as violations of structural balance. For example, the existence of a group of individuals who share only positive relationships with everyone in the network counts as imbalance in the CC Problem. Nonetheless, the individuals in this group could be construed as mediators (i.e. their relations probably won't change over time) and, as pointed in (Esmailian et al. 2014), their relations should not be considered as a contribution to the imbalance of the network.

Using this new definition, structural balance was generalized to a version labeled as *relaxed structural balance* (Doreian and Mrvar 2009). Similarly to the CC

[3] The original data set from UCI Machine Learning Repository (Bache and Lichman 2013) consists of 20000 messages taken from 20 newsgroups. However, since the authors' bounding technique did not scale to the full dataset, they restricted their testbed to a subsample of 100 messages from each newsgroup, for a total of 2000 messages.

problem, measuring the relaxed structural balance can be accomplished through the solution to the Relaxed Correlation Clustering (RCC) problem. A redefinition of relaxed imbalance of a partition P that takes into account symmetric relationships among clusters is available, with the name of Symmetric Relaxed Imbalance (SRI) (Figueiredo and Moura 2013).

$$SRI(P) = \sum_{1 \leq i \leq l} \min\{{\Omega^+(S_i,S_i), \atop \Omega^-(S_i,S_i)}\} + \sum_{1 \leq i < j \leq l} \min\{{\Omega^+(S_i,S_j), \atop \Omega^-(S_i,S_j)}\}. \quad (6)$$

This gives rise to a new graph clustering problem, the Symmetric Relaxed Correlation Clustering Problem, which will be used in this work. The problem is stated as follows.

Problem 22 (Symmetric RCC problem) *Let $G = (V, E)$ be a signed graph, w_e be a non-negative edge weight associated with edge $e \in E$ and k be an integer value satisfying $1 \leq k \leq n$. The symmetric relaxed correlation clustering problem is the problem of finding a l-partition P of V, with $l \leq k$, such that the symmetric relaxed imbalance $SRI(P)$ is minimized. Let us denote this minimal value by SRCC(G,k).*

It is worth noting that the Symmetric RCC (SRCC) problem is closely related with the CC problem but it is not a particular case nor is it a generalization. Actually, each feasible solution (a graph partition) of the SRCC problem is also feasible in the CC problem but the problems have different cost functions, i.e., there are different ways of evaluating the imbalance of a partition. The SRCC problem is intuitively as difficult as the CC problem and is indeed a NP-hard problem (Figueiredo and Moura 2013).

2.2.1 Literature review of the RCC problems

To the best of our knowledge, the RCC problems have been proposed for only the evaluation of structural balance in social networks. However, these problems could be also used as an approach to efficient community detection. Two solution methods were initially proposed in the literature for RCC problems: a greedy heuristic approach (Doreian and Mrvar 2009) and a branch-and-bound procedure (Brusco et al. 2011). Computational experiments with both procedures were reported over literature instances with up to 29 vertices (small instances in Section 4.2, item (i)) and for random instances with $|V| \in \{20, 40\}$ (Doreian and Mrvar 2009, Brusco et al. 2011). For the branch-and-bound procedure, the values considered for k were $k \leq 7$ for literature instances and $k \in \{3, 5\}$ for the set of random instances.

Further on, (Figueiredo and Moura 2013) presented the first mathematical formulation for the RCC and SRCC problems: a representatives ILP formulation, which is harder to solve than the CC ILP formulation. For the set of benchmark instances, the ILP formulation was able to solve the problem when $k = 2$, $k = 3$ (for some instances) and for high values of k. The results presented for the branch-and-bound procedure in (Brusco et al. 2011) demonstrated that this approach was efficient in the solution of RCC instances with $k \leq 8$. At this point, no metaheuristic approaches have been applied to solve the RCC problems.

3 Solving the Correlation Clustering problem by metaheuristics

We have implemented ILS (Lourenço et al. 2003) for the solution of the CC problem. Our ILS algorithm was first implemented sequentially. Then, in order to obtain performance improvements that would allow the program to scale to large network instances, we extended the algorithm to run tasks in parallel by applying two well-known strategies.

3.1 Multistart Iterated Local Search (ILS) heuristic for the CC Problem

The Iterated Local Search (Lourenço et al. 2003) (ILS) belongs to a class of local search algorithms (as VNS (Mladenović and Hansen 1997), IGS (Ruiz and Stützle 2007), GRASP (Feo and Resende 1995) and ad hoc local-search schemes (Brusco and Köhn 2009)) proposed in the literature to efficiently investigate the solution space of combinatorial problems. The difference among all these methods is the philosophy used to walk in the solution space. ILS is a metaheuristic based on stochastic multi-restart search, which iteratively applies local search to perturbations of the current best solution, generating a random walk in the space of local optima. Each new solution obtained is then refined using an embedded heuristic.

As mentioned in Section 2.1, the CC problem is related to the clique partitioning problem. The authors in (Brusco and Köhn 2009) developed a neighborhood search heuristic (known as NS-R) for the clique partitioning problem, which is similar to the one proposed in this section. NS-R includes the basic principles of random perturbation and pursuit of a local optimum through a local search procedure also present in ILS.

Our ILS algorithm to solve the CC problem reuses the constructive and local search modules of GRASP, as well as its *multistart* nature. We have also added a procedure that creates perturbations in the best current solution, creating a multistart Iterated Local Search (ILS) algorithm called $ILSMultiStartCC$. Multistart provides an additional approach to diversify the obtained solutions, restarting the ILS algorithm with a new randomized initial solution any time a search region has been largely explored.

Algorithm 1 outlines how ILS works, by the use of 4 components: (i) *ConstructivePhase*, which creates an initial solution, (ii) the local search module *VariableNeighborhoodDescent*, (iii) a procedure called *Perturbation* to modify a solution at random, and (iv) an acceptance criterion, which decides from which solution the search will restart. The *iter* parameter contains the number of multistart iterations – how many times ILS is to be invoked, returning the best solution from all ILS executions. The $iterMaxILS$ value represents the number of ILS iterations (inner loop), including perturbation and local search operations. Lastly, the $perturbationMax$ parameter restricts the degree of perturbations applied to a specific solution. Below we provide a full description of each step of the algorithm.

Each ILS iteration starts with the construction of an initial greedy-randomized solution. This is performed inside the *ConstructivePhase* procedure, in Algorithm 2. Formalizing it requires additional notation: let $P = \{S_1, S_2, ..., S_l\}$ denote a *partial partition* (i.e., a partition of a proper subset of V). We specify below a *function* $f : (V \setminus \bigcup_{1 \leq k \leq l} S_k) \to \mathbb{R}$, which will measure the effect on imbalance I of

inserting a vertex i in the partial partition P.

$$f(i) = \min\left(I(P \cup \{i\}), \min_{\substack{1 \leq k \leq l \\ S_k \cup \{i\}}} I(P)\right). \tag{7}$$

This minimization function compares the cost of inserting vertex i in a new cluster ($f(i) = I(P \cup \{i\})$) with the cost of inserting it into one of the l clusters in P. Remark that a vertex having a low cost function value will possibly generate less imbalance when added to the partial partition P.

In this phase, the ordered set L_f is defined (line 3) as the set of vertices $\{v : v \in V \setminus \bigcup_{1 \leq k \leq l} S_k\}$ sorted in decreasing order of the function value $f(v)$. At each iteration (lines 4-8), we randomly pick a vertex i between the first $\lfloor \alpha . \mid L_f \mid \rfloor$ elements in this set and add it to the partial partition. Observe that the parameter α denotes the randomness level of the construction phase. This operation is repeated until a partition of V is generated.

The construction method does not necessarily return a locally optimal solution with regard to a certain neighborhood. Thus the solution P returned by *ConstructivePhase* can be improved by the local search procedure *VariableNeighborhoodDescent* (Algorithm 3).

For local search, we apply the Variable Neighborhood Descent (VND) heuristic, a variant of the Variable Neighborhood Search (VNS) method, proposed by Mladenović and Hansen (1997). Unlike some methods which employ only a single neighborhood, VND systematically alternates between different neighborhoods within a local search. It is based on the idea that a local optimum found by one neighborhood structure is not necessarily the local optimum of another neighborhood structure. VND begins with the partition returned by the construction phase, trying to improve the solution value $I(P)$ with the new value $I(\overline{P})$ from the best solution found in the $r-neighborhood$, denoted by $N_r(P)$, and defined as the family of all partitions obtained by moving r vertices in P from one cluster into another. If an enhancement is found, r returns to its initial value and the best solution is updated (lines 7 and 8). Otherwise, the next neighborhood is traversed (line 11). If no improving partition is found in the most distant neighborhood of the current solution, local search stops. Note that the neighborhood analyzes partitions of various sizes (i.e. a vertex can be moved into a new cluster or can be removed from a single vertex cluster). Also observe that we have applied a neighborhood with $r \leq 2$ and as its complete traversal was very time-consuming, we adopted a first improvement (first descent) heuristic, which means that local search will halt whenever it finds an improvement of the current solution.

The subsequent stage of the algorithm is producing a perturbation, consisting of t random vertex movements between clusters (Algorithm 4). In the end of the main loop (line 21 in Algorithm 1), a simple acceptance criterion was applied. Only improving moves were allowed, that is, a move that forces the cost to decrease, which in turn corresponds to a first-improvement descent in the set of solutions.

As observed by Den Besten et al. (2001), the efficacy of the local search module is of extreme importance to ILS, since it influences not only the quality of the final solution of the metaheuristic, but also the total time spent. In turn, the perturbations should enable ILS to escape from local optima, meanwhile avoiding excessively strong movements that bring the same disadvantages of random

Algorithm 1: *ILSMultiStartCC*

```
Input: G = (V, E), α, iter, iterMaxILS and perturbationMax
Output: partition P*
P* = ∅;  I(P*) = ∞;  i = 1;
while (i ⩽ iter)
    P = ConstructivePhase(G, α);
    P = VariableNeighborhoodDescent(P, G);
    j = 1;  t = 1;
    repeat
        P̄ = Perturbation(P, t);
        P̄ = VariableNeighborhoodDescent(P̄, G);
        if (I(P̄) < I(P))
            P = P̄;  j = 1;  t = 1;
        else
            j = j + 1;
            if (j ⩾ iterMaxILS)
                t = t + 1;  j = 1;
            end if
        end if
    while (t ⩽ perturbationMax)
    i = i + 1;
    if (I(P) < I(P*))
        P* = P;
end while
return P*;
```

Algorithm 2: *ConstructivePhase*

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

Algorithm 3: *VariableNeighborhoodDescent*

```
Input: G = (V, E) and a partition P
Output: partition P
r = 1;
while (r ≤ 2)
  for all P̄ ∈ (N_r(P))
    if (I(P̄) < I(P))
      r = 1;
      P = P̄;
  end for
  if (P has not improved)
    r = r + 1;
end while
return P;
```

restart. The perturbations, as well as the acceptance criteria, have huge influence on the walk through the search space and can be applied to algorithm tuning, oscillating between intensification and diversification of the search. Designing an ILS algorithm thus requires an appropriate configuration of its modules, with global efficiency in mind.

Algorithm 4: *Perturbation*

```
1  Input: G = (V, E), a partition P and perturbation level t
2  Output: partition P
3  i = 1;
4  while (i ⩽ t)
5     Choose a random cluster c_x ∈ P and a random vertex i ∈ c_x;
6     Choose a random cluster c_y ∈ P such that x ≠ y;
7     Move vertex i from cluster c_x to cluster c_y;
8     i = i + 1;
9  end while
10 return P;
```

3.2 Parallel ILS and Parallel Variable Neighborhood Descent

In practice, interesting real-world optimization problems are often NP-hard, complex, and time consuming. Although the utilization of a sequential metaheuristic provides significant time reduction in the search process, execution time remains high for real-world problems arising in both academic and industrial domains. For example, when it comes to signed social networks, the instances generated from Wikipedia, Slashdot and Epinions websites, available in Leskovec and Krevl (2014), have thousands of nodes and in some cases almost a million relationships[4]. Therefore, parallelism presents as a natural way not to only reduce the search time, but also to improve the quality of the provided solutions.

The independent approach was our first method to create a parallel program (Gendreau and Potvin 2010), applying message passing for inter-process communication. Information exchange between processors occurs only on problem input, detection of process termination and collection of best overall solution. When p processors are executed, a single (master) process reads the problem data and forwards it to the other $p-1$ processes. Each process is then in charge of executing a portion of the multistart iterations. The program terminates once each individual process finishes executing. Observe that distinct random seeds are used by each process in the construction phase (line 5 in Algorithm 2) to benefit the exploration of various search spaces.

Processing the search space in parallel can also increase the exploration in the search space, reducing the computation time as well (Alba 2005, Crainic and Toulouse 2010). With this idea in mind, an interesting approach is extending parallel metaheuristics to use the Parallel Variable Neighborhood Descent (PVND) (Ekşioglu et al. 2002) algorithm inside the local search phase. In this additional parallelization approach, besides having a set of (master) processes responsible for executing a copy of ILS, there is a new kind of process known as search slave, which is specialized in executing the local search (VND) algorithm over a subset of the search space. The only difference from the previously explained VND procedure is the neighborhood traversal (line 5 in Algorithm 3), which is now executed in parallel, attempting to obtain a balanced load among the processor cores. This way the search space of the $r-neighborhood$ of the current best solution ($N_r(P)$) is partitioned among the available search slave processes in order to search for the combination that minimizes the imbalance value $I(P)$. Search slave processes send a message to an ILS master process whenever they find an improved solution or

[4] Wikipedia adminship election data has 7,000 vertices and 100,000 edges, Epinions signed social network has 131,828 vertices and 841,372 edges and Slashdot Zoo signed social network (from February 21 2009) is comprised of 82,144 vertices and 549,202 edges.

finish their search space traversal. In the end, the ILS master process will gather each solution produced in parallel, selecting the best one.

4 Experiments

The following methods will be used to access the perfomance of the ILS introduced:

- **Integer Linear Programming (ILP) model**: can provide an exact solution to the CC problem, but is unable to process larger instances. Besides using the exact solution provided by the ILP model, we identify for which instance size the ILP formulation becomes very big and the solver is not able to return a good solution within the time limit. In these cases, the use of heuristics is necessary.
- **GRASP metaheuristic**[5] (Drummond et al. 2013): provides efficient solutions to the CC problem for signed graphs of up to 8000 nodes. Additionally, we have made two modifications to the original GRASP algorithm from Drummond et al. (2013). First, we have adapted the construction phase, originally based on Nascimento and Pitsoulis (2013), replacing the modularity gain function (used on unsigned graphs) by an imbalance (minimization) gain function, which makes more sense for signed graphs. In order to improve the local search procedure, we replaced the simple neighborhood traversal with a Variable Neighborhood Descent heuristic;
- **VNS metaheuristic** (Mladenović and Hansen 1997): Variable Neighborhood Search consists of a well-established standard metaheuristic for combinatorial optimization. Based on our ILS algorithm, we have implemented VNS by doing the following changes to Algorithm 1:
  - line 3: replacing "$P^* \leftarrow \emptyset$" and "$I(P^*) = \infty$" by "$P^* \leftarrow ConstructivePhase(G, \alpha)$";
  - line 5: replacing "$P \leftarrow ConstructivePhase(G, \alpha)$" by "$P \leftarrow P^*$".
- **Doreian Mrvar Method, implemented in Pajek software**[6] (De Nooy et al. 2011): consists of a relocation algorithm for partitioning signed graphs. In simple terms, a given partition (randomly generated at each iteration) is optimized to get as much as possible positive edges inside clusters and negative edges between clusters. In the local optimization procedure, vertices can be moved from one cluster to another or pairs of vertices can be interchanged between clusters. For each such change, the criterion function is evaluated. If a relocating change leads to a decrease of the criterion function, the new partition is retained and the process continues until the criterion function cannot be lowered. If the procedure finds several optimal solutions, all of them are reported. More details are available in Doreian and Mrvar (1996b);
- **VOTE/BOEM heuristic** (Elsner and Schudy 2009): provides good correlation clustering solutions on datasets far too large for ILP to scale. The algorithm

[5] In simple terms, GRASP (Feo and Resende 1995) is a multistart metaheuristic in which each iteration consists basically of two phases: construction (Algorithm 2) and local search (Algorithm 3).

[6] The Pajek program is for analysis and visualization of large networks, which provides features such as cluster identification, decomposition of large networks, visualization tools, as well as an implementation of several efficient algorithms for analysis of large networks, having thousands or even millions of vertices.

is composed of two phases: (1) the VOTE algorithm adds each vertex to the cluster that minimizes the correlation clustering objective; and (2) the best one element move (BOEM) algorithm repeatedly makes the most profitable and the best element move until a local optimum is reached.

As previously metioned, although an approach based on genetic algorithms has been proposed in Zhang et al. (2008) for the CC problem and applied to document clustering, unfortunately, we were unable to implement the procedure and compare it to our algorithms. There was no explanation as to how the genetic operators were applied and we were also unable to obtain or recreate the instances used in the experiments.

Moreover, we were also unable to directly compare our results with the solutions of the FEC algorithm in Yang et al. (2007), even though we had generated and used, in our experiments, the random signed networks with controlled community structures, as presented in their work. We verified that, since the FEC heuristic is focused on community detection, using a different criterion than minimizing imbalance, FEC does not present optimal (or close to optimal) solutions for the CC problem. In fact, when using the aforementioned random instances as reference, the imbalance measures of the best clustering configurations pointed by FEC are higher than the solutions generated by our algorithms.

We next present extensive computational results obtained with four benchmarks.

4.1 Computational environment

The algorithms described in the previous section were implemented in ANSI C++ and MPI (OpenMPI) for message passing. All experiments were performed (with exclusive access) on a cluster with 42 nodes, each one with two Intel Xeon Quad-Core 2.66GHz processors and 16Gb of RAM under Linux (Red Hat 5.3). The ILP formulation presented in Section 2 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. Additionally, all heuristic outcomes represent the average of 25 independent runs.

4.2 Test problems

Computational experiments were undertaken on (i) a set of 32 social networks from the literature, (ii) a set of 63 network instances that represent the United Nations General Assembly (UNGA) voting data, (iii) a set of 60 completely random instances, and (iv) a set of 45 random signed networks with a predefined community structure. We will briefly describe these instances[7].

(i) This set of instances is composed by 22 small size instances normally used in blockmodeling approaches to structural balance (Brusco 2003, Doreian and Mrvar 2009, Figueiredo and Moura 2013) and 6 signed networks (where $n \in \{600, 1000, 2000, 4000, 8000, 10000\}$) extracted from the large scale social

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

network representing the technology-related news website Slashdot[8](Leskovec et al. 2010, Facchetti et al. 2011).

(ii) We generated 63 medium-sized social networks based on the United Nations General Assembly (UNGA) voting records of each annual session between the years of 1946 and 2008[9]. Such networks are weighted versions of UNGA signed graphs used by Figueiredo and Frota (2014). The set of vertices in each signed graph represents the list of countries in the associate annual voting session. The set of weighted positive/negative edges is defined as follows. For each pair of vertices (countries) i, j and for each resolution voted in the session, we totaled the weights associated with all pairs of votes from i and j: edge weights can be equal to 1.0 or 0.5; a positive edge means an agreement, while a negative edge represents a disagreement. Following an observation from Macon et al. (2012), we treat differently the disagreement (agreement) in a yes-no (yes-yes or no-no) pair of votes on a same resolution from a yes-abstain or no-abstain (abstain-abstain) pair. We normalize by the total number of votes in a session.

(iii) We generated random social networks with $n \in \{200, 400, 600\}$, varying network density $d = 2 \times |E|/(n^2 - n)$ and negative graph density defined here as $d^- = |E^-|/|E|$. For each value of n, 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 $\{0.2, 0.5, 0.8\}$. The random selection of graph edges follows a uniform distribution.

(iv) We generated 45 random signed networks with a predefined community structure, according to Yang et al. (2007). The random signed network is defined as $SG(c, n, k, p_{in}, p_-, p_+)$, where c is the number of communities in the network, n is the number of nodes in each community, k is the degree of each node, p_{in} is the probability of each node connecting other nodes in the same community, p_- denotes the probability of negative links appearing within communities, and p_+ denotes that of positive links appearing between communities. The signs of the generated links within communities are positive, whereas those between communities are negative. Based on the network generation process, there will be $c \times n \times k \times p_{in} \times p_-$ negative links within communities and $c \times n \times k \times (1 - p_{in}) \times p_+$ positive links outside communities.

4.3 Methodology

We used the instances in (i) and (ii) to parametrize the heuristics. After testing various combinations of parameters, including the maximum level of perturbation, the configurations in Table 1 presented the best results for ILS, depending on the instance size. In order to apply the GRASP algorithm, the configuration in Table 2 presented the best results.

When testing the independent parallel version of ILS, if p is the number of simultaneous processes, the number of multistart iterations (*iter* parameter) is reduced to $iter/p$. This is due to the fact that the parallel procedure executes p

[8] We have extracted subsets of Slashdot Zoo signed social network from February 21 2009 (Leskovec and Krevl 2014), containing the first n vertices, where $n \in \{600, 1000, 2000, 4000, 8000, 10000\}$. Since the original Slashdot network is a signed digraph, before extracting the subsets, we have converted it into an undirected graph.

[9] United Nations General Assembly Voting Data, by Anton Strezhnev and Erik Voeten, http://hdl.handle.net/1902.1/12379. Accessed in Apr 2016.

SetPar	**Graph size**	Time limit	Alpha	Neighborhood	Iterations	ILS iterations	Perturbation level
A	$n < 300$	1 hour	$\alpha = 0.4$	$r = 1$	$iter = 10$	$iterMaxILS = 5$	$perturbMax = 3$
B	$n \geq 300$	2 hours	$\alpha = 1.0$	$r = 1$	$iter = 10$	$iterMaxILS = 5$	$perturbMax = 30$

Table 1 Parameters used in the ILS algorithm.

SetPar	**Graph size**	Time limit	Alpha	Neighborhood	Number of iterations without improvement
A	$n < 200$	1 hour	$\alpha = 0.8$	$r \leq 2$	$iter = 400$
B	$n \geq 200$	2 hours	$\alpha = 1.0$	$r = 1$	$iter = 400$

Table 2 Parameters used in the GRASP algorithm.

independent metaheuristic procedures at the same time, which provides enough variability so that the aggregated results of parallel ILS are quality-equivalent to those of the sequential ILS.

4.4 Exact solution by ILP formulation

As noted in the literature (Mehrotra and Trick 1998), the linear relaxation of the ILP formulation described in Section 2.1 provides a very good representation of the problem which allows discovering the optimal solution for many instances by solving this linear relaxation. Experiments reported in Figueiredo and Moura (2013) confirm this assertion: the 22 small instances in set (i) were solved to optimality in some seconds. Also, in our experiments, the instances in set (ii) were solved to optimality by the ILP formulation in the root of the branch and bound tree.

The drawback of the ILP approach appears when we attempt to solve larger instances, which also happens to other related problems (Mehrotra and Trick 1998, Brusco and Köhn 2009). For most completely random instances in (iii), the solver is unable to find an optimal solution to the CC problem within the time limit. On Slashdot-based instances, the ILP formulation becomes too big and the solver is unable to return any feasible solution within the prescribed time limit.

4.5 Sequential ILS *vs.* sequential GRASP

We compared both SeqGRASP and SeqILS, in their best configurations. When applying the set of completely random instances (iii) as input, we observe the superiority of SeqILS over SeqGRASP. For random instances with $200 \leq n \leq 600$, as Figure 1 shows[10], the average time to target of ILS was smaller than GRASP's: on average, ILS is 5 times faster than GRASP. SeqILS running time was strictly better in 30 of these instances, while SeqGRASP found the target solution value[11] earlier for 6 instances.[12]

[10] Instances solved in less than 100 seconds were omitted from the graph.

[11] The target solution value is calculated as the best solution value that can be obtained by all heuristic methods when solving a specific instance.

[12] The full tables containing the comparison between SeqGRASP and SeqILS are available in http://www.ic.uff.br/~yuri/files/CC-ADDcomp.zip

Additionally, when applying both metaheuristics to solve the random instances with predefined community structure in (iv), SeqILS was, on average, superior to SeqGRASP in execution time, as shown in Figure 2.[13] An analysis of the execution time between SeqILS and SeqGRASP indicates the superiority of SeqILS in 28 of 44 instances.[14] On average, SeqILS is more than 2 times faster than SeqGRASP to find the specified target solution values.

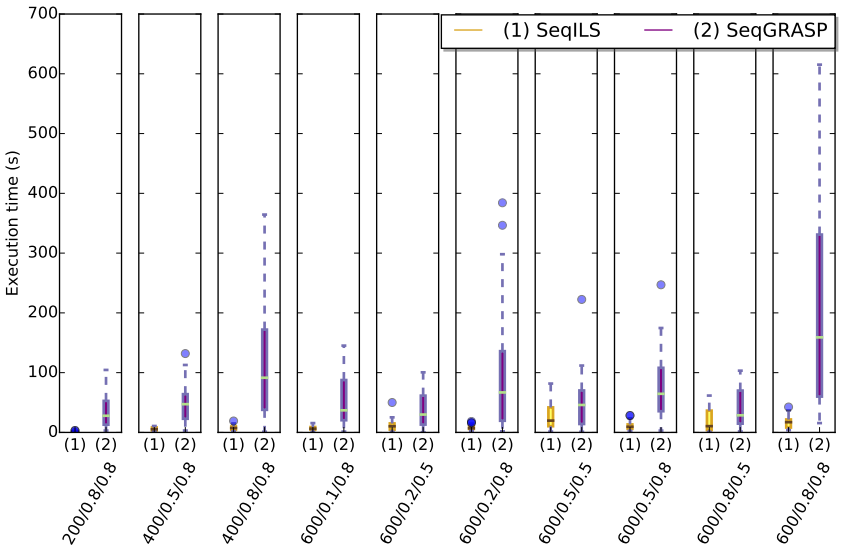

Fig. 1 Sequential GRASP (SeqGRASP) and sequential ILS (SeqILS) results for random instances in (iii). On the x-axis, instances are labeled according to their parameters $(n/d/d^-)$ and those solved in less than 100 seconds were omitted. Execution time spent (in seconds) by each algorithm to reach a specific target solution value, after 25 independent executions.

Experiments with larger social networks were also performed with SeqGRASP and SeqILS. In this context, when solving Slashdot-based instances in (i), after running both algorithms for 2 hours, ILS has improved the solution quality on larger instances, with more than 2000 vertices (Table 3). Additionally, both SeqGRASP and SeqILS solve the CC problem with low variances in solution value, as seen in the confidence intervals (columns SeqGRASP-CI and SeqILS-CI in Table 3). This confirms the robustness of these heuristics, for the stability of the solution values returned by both algorithms.

Figure 3 presents an additional comparison between both algorithms, based on Time-to-Target Plots (TTT-plots) (Aiex et al. 2007), used to analyze the behavior of stochastic algorithms. One can notice the improved behavior of SeqILS when compared to SeqGRASP. For example, when solving Slashdot $n = 8000$ instance, the probability of SeqILS finding the target solution of 16068 in 2000 seconds is almost equal to 100% while the same probability lies below 65% for SeqGRASP.

[13] Instances solved in less than 5 seconds were omitted from the graph.

[14] The full tables containing the comparison between SeqGRASP and SeqILS are available in http://www.ic.uff.br/~yuri/files/CC-ADDcomp.zip

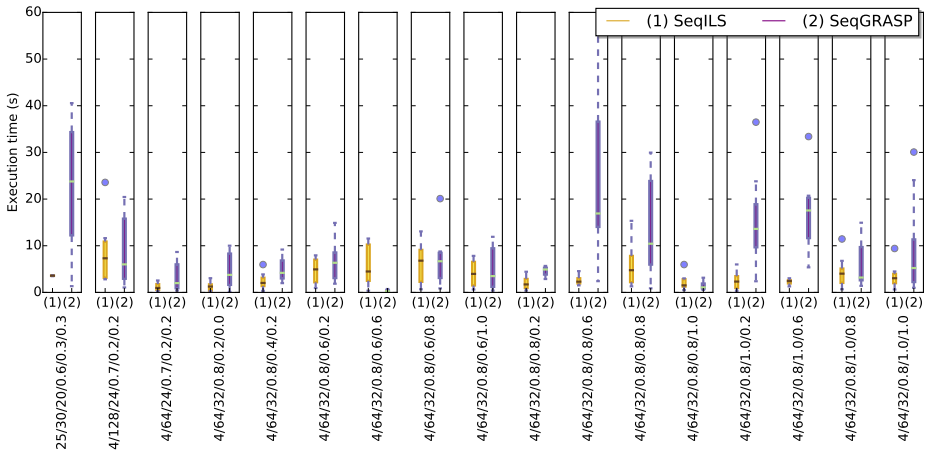

Fig. 2 Sequential GRASP (SeqGRASP) and sequential ILS (SeqILS) CC results for random instances with predefined community structure in (iv). The boxplot shows the execution time (in seconds) spent by each algorithm to reach the specified target solution value, after 25 independent executions. On the x-axis, instances are labeled according to their parameters ($c/n/k/p_{in}/p_-/p_+$) and those solved in less than 5 seconds were omitted.

	SeqILS		**SeqGRASP**			**SeqVNS**		
n	**Avg I(P)**	**CI**	**Avg I(P)**	**CI**	**Gap I(P)**	**Avg I(P)**	**CI**	**Gap I(P)**
1000	600	0	600	0	0	600.95	0.35	0.95
2000	2,184.14	0.15	2,184.76	0.17	0.62	2,237.34	9.55	53.20
4000	6,193.35	0.57	6,198.79	0.49	5.44	6,232.56	3.99	39.21
8000	16,060.55	1.66	16,076.52	0.88	15.97	16,063.82	4.74	3.27
10000	20,571.35	5.40	20,579.29	1.48	7.94	20,580.94	3.20	9.59

Table 3 Results obtained on Slashdot instances, with sequential ILS (SeqILS), sequential GRASP (SeqGRASP) and sequential VNS (SeqVNS). Number of vertices: n; Avg I(P): value of the best solution found after 2 hours of execution; CI is the 95% confidence interval of the solution value I(P), for each algorithm; Gap I(P)=[SeqGRASP or SeqVNS Avg I(P)]−[SeqILS Avg I(P)].

The above results lead us to conclude that the potential power of ILS lies in its random and biased sampling (perturbations) of the set of local optima found in the local search step. In most cases, i local searches within ILS perturbations can be invoked much faster than if the same i local searches were performed within GRASP's random restart framework.

4.6 Sequential ILS *vs.* sequential VNS

In order to compare the performance obtained with metaheuristics that focus on diversification *vs.* intensification, we compared our sequential ILS procedure (SeqILS) with Variable Neighborhood Search (SeqVNS) to solve the most difficult instances (Slashdot instances) in set (i). After running both algorithms for 2 hours, ILS has improved the solution quality on larger instances (Table 3). On the other hand, VNS is not able to reach the same average solution values.

In theory, ILS has the advantage of increased diversification, using perturbations to better explore the search space and escape from local optima. On its turn, VNS should be more lightweight to run, since it avoids executing local search over

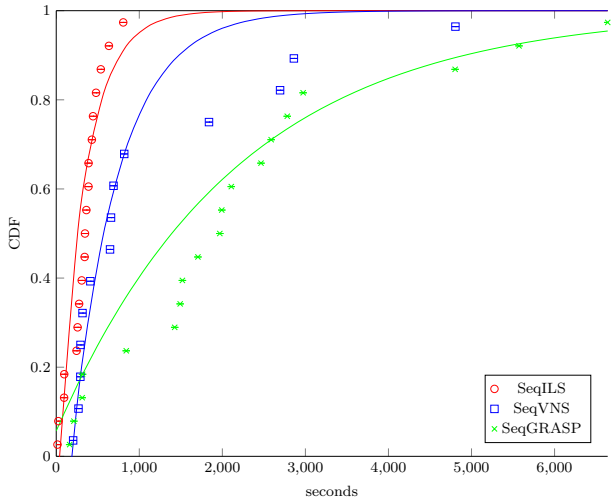

Fig. 3 Time-to-Target Plot (TTT-plot)(Aiex et al. 2007) for SeqILS, SeqVNS and SeqGRASP algorithms when solving Slashdot $n = 8000$ instance to obtain the target solution value $I(P) = 16068$. TTT-plots show, on the y-axis, the probability that an algorithm will find a solution at least as good as a given target solution value within a specific running time, displayed on the x-axis.

new initial solutions from random restart, a time consuming step. As a consequence, VNS tends to concentrate on the region of the best known solution.

That being said, we provide a second comparison between both algorithms, based on Time-to-Target Plots (TTT-plots), evidentiating the superiority of SeqILS when compared to SeqVNS. When solving Slashdot $n = 8000$ instance, the probability of SeqILS finding the target solution of 16068 in 1000 seconds is above 95% while the same probability lies below 80% for SeqVNS. In summary, when solving the CC problem, the ILS algorithm is more efficient than VNS.

4.7 Comparison with existing heuristics

Our goal in the next subsections is to verify the limit of Doreian Mrvar and VOTE/BOEM heuristics. Notice that these heuristics are simplified versions of a local search heuristic, so they tend to have inferior performance when compared to metaheuristics. In order to run the tests, we chose to slice a huge real-world network into smaller sizes. For that purpose, we used the Slashdot Zoo network from February 21 2009 instance (Leskovec and Krevl 2014), which presented one of the largest values of imbalance, for both the CC and SRCC measures (see Table 6).

4.7.1 Sequential ILS vs. Doreian Mrvar Method

The Doreian Mrvar Method (implemented in Pajek software) presents good-quality results (with respect to solution value and average time spent) for the small-sized literature instances in (i) and also for random social networks of up to 50 vertices (Figueiredo and Moura 2013). However, for larger instances, it presents poor performance.

Instance					*SeqILS*			Doreian Mrvar Method			*VOTE/BOEM*		
n	$\|E^-\|$	$\|E^+\|$	$w(E^-)$	$w(E^+)$	k	BestSol	Time	k	BestSol	Time	k	BestSol	Time
600	156	1761	156	1761	9	109.2	13.44	9	133.5	1010.83	12	111.0	1.14
1000	859	5132	859	5132	18	600.1	30.20	18	630.4	5654.79	44	608.9	4.27
2000	3217	17598	3217	17598	31	2191.9	99.76	-	-	-	105	2264.2	43.86
4000	8664	40868	8664	40868	44	6207.2	528.44	-	-	-	147	6216.5	395.10
8000	22789	86916	22789	86916	80	16071.6	2208.41	-	-	-	375	16115.8	3475.91
10000	29805	109266	29805	109266	95	20591.4	4277.18	-	-	-	169	23458.4	7041.97

Table 4 Sequential ILS for the CC Problem, here named $SeqILS$, *vs.* Doreian Mrvar Method *vs.* $VOTE/BOEM$ results obtained on Slashdot signed graphs. BestSol is the average value of the best imbalance found within the time limit, after 25 executions of each algorithm.

For this benchmark, we compared our ILS procedure with Doreian Mrvar Method to solve the Slashdot instances in set (i). We have obtained average results of 25 executions (Table 4). The software Pajek (version 4.09)[15] was invoked with the following configuration: Doreian Mrvar Method, starting from a random partition (1-mode), 100 repetitions, $\alpha = 0.5$, minimum number of vertices in clusters $= 1$.

Pajek's Doreian Mrvar Method accepts only one stopping criterion: number of iterations. Since an iteration of this method is fundamentally different from an iteration performed by ILS, the number of iterations is not a sound basis for comparison of both algorithms. Therefore, as an alternative, we have set the time-limit parameter of ILS to 2 hours and the number of repetitions of Doreian Mrvar Method to 100, which was enough for this method to either reach the same solution value of ILS, or exceed its time limit with an inferior solution value. In this way, we were able to compare both solution values and execution times in a fair manner. When it comes to these larger instances, our ILS procedure can get to the same solution or to a better one at least 13 times faster than Pajek's Doreian Mrvar Method. We do not include Doreian Mrvar Method execution data for instances with $n > 1000$, as it was unable to complete 100 iterations within 2 hours.

4.7.2 Sequential ILS vs. VOTE/BOEM

We have implemented the VOTE/BOEM heuristic for the CC Problem (here named $VOTE/BOEM$), comparing the obtained solutions with the previous results from our sequential ILS algorithm. Again, our goal was to solve the Slashdot instances in set (i). The time limit was set to 2 hours for both procedures.

As shown in Table 4, the results of the sequential ILS are always better than $VOTE/BOEM$. For smaller instances, the faster execution times of $VOTE/BOEM$ are not surprising, since the heuristic is equivalent to only one multistart iteration of the GRASP metaheuristic proposed in Drummond et al. (2013). However, as the number of vertices increases, the ILS procedure outperforms VOTE/BOEM. Our sequential ILS algorithm becomes the fastest choice to solve the CC problem when the number of vertices is big ($n \geq 8000$).

[15] In order to solve the CC Problem with Doreian Mrvar Method inside Pajek, we followed these steps: after loading the network file, we created a random initial partition using $1-mode$. At this point we defined the number of clusters k of the solution. We have used the same number of clusters of the solution returned by ILS. Finally, we called the resolution method from the menu (Network → Signed Network → Create Partition → Doreian Mrvar Method), specifying the number of repetitions (we set it to 100), the alpha parameter (set to 0.5 to match the CC Problem objective function) and the minimum number of vertices in clusters (set to 1).

Instance		SeqILS		ParILS/SeqVND(10)				ParILS/ParVND(8)			
	Target I(P)	Avg Time	CI	Avg Time	CI	Speedup	Efficiency	Avg Time	CI	Speedup	Efficiency
600	109	13.44	3.81	3.33	0.90	4.04	40.36%	4.74	0.37	2.83	35.41%
1000	600	30.20	1.35	5.75	0.41	5.25	52.55%	9.49	0.46	3.18	39.78%
2000	2185	99.76	6.14	18.39	1.82	5.43	54.25%	57.24	2.30	1.74	21.79%
4000	6195	528.44	35.20	91.14	7.75	5.80	57.98%	330.80	15.25	1.60	19.97%
8000	16065	2,208.41	165.49	425.57	36.25	5.19	51.89%	590.40	12.92	3.74	46.76%
10000	20580	4,277.18	291.29	709.04	47.25	6.03	60.32%	1,012.47	32.03	4.22	52.81%
Average						**5.29**	**52.89%**			**2.89**	**36.08%**

Table 5 CC results obtained on Slashdot signed graphs by the use of different metaheuristic approaches: Sequential ILS ($SeqILS$), Parallel ILS with sequential VND ($ParILS/SeqVND(np)$) and Parallel ILS with parallel VND ($ParILS/ParVND(np)$), where np is the number of processes in parallel. Target I(P) is the target imbalance value used as stopping criterion for each algorithm. Avg Time is the average execution time (in seconds) spent by each algorithm to reach the specified target solution value, after 25 independent executions. CI is the 95% confidence interval of the execution time, for each algorithm.

4.8 Sequential *vs.* parallel ILS

Since the Slashdot instances in (i) have shown the limits of the sequential version of ILS, we have used them to assess the performance of the parallel ILS algorithm. Three different solution methods were applied:

- sequential ILS metaheuristic ($SeqILS$, using only one core of a processor);
- independent parallel ILS with sequential local search ($ParILS/SeqVND$);
- parallel ILS with parallel local search ($ParILS/ParVND$).

The parameters of the procedures used in the parallel approaches of ILS are the same used for testing the sequential algorithm, except for the number of ILS multistart iterations. For independent parallel ILS ($ParILS/SeqVND$), in order to perfectly divide the number of multistart iterations of the original ILS procedure, 10 master processes execute one multistart iteration of ILS each. In $ParILS/ParVND$, the runtime configuration has 8 processes running in parallel, out of which 2 processes are ILS masters (each one running 5 ILS iterations) and the other 6 processes consist of VND search slaves, 3 for each master[16].

Finally, in order to compare the sequential and parallel procedures, the following additional stopping criterion was applied: all procedures stop whenever the specified target solution value [Target I(P)] is found.

On average, the independent parallel ILS ($ParILS/SeqVND$) with 10 processes is the best solution method[17]. As shown in Table 5, it can solve the Slashdot instances in set (i) up to 6 times faster than sequential ILS, with an average efficiency of 53%.

On the other hand, the results obtained with parallel ILS with parallel local search ($ParILS/ParVND$) have no such performance improvement as one would

[16] We conducted several experiments to find the optimal number of processes to be used in the parallel algorithms. This setting is closely related to the hardware configuration of the computer cluster used in the experiments. Since each machine in the computer cluster has 2 quad-core CPUs (8 processor cores), it can host 8 processes running in parallel. In $ParILS/ParVND$, we chose to group each ILS master process together with its corresponding VND search slaves, in order to maximize the performance of message exchange between related processes.

[17] Whenever it is required to assess the performance of parallel algorithms, two metrics are applied: speed-up $Su(p)$ measures the acceleration observed for the parallel algorithm when compared with its sequential version and efficiency $E(p)$ measures the average fraction of time along which each process is effectively used. Thus, $Su(p) = T(seq)/T(p)$, such that $T(seq)$ is the time required for the sequential algorithm and $T(p)$ the time required for the parallel algorithm run on p processors, and $E(p) = Su(p)/p$.

expect. Using 8 processes to execute the algorithm (2 ILS masters and 3 VND search processes per master) resulted in an average speedup of 2.9 (average efficiency of 36%) for Slashdot instances. Therefore, the additional computational resources required to run parallel local search are not worth the available acceleration and efficiency brought by this procedure.

It is worth noticing that, although we have also applied the same parallelization approaches to GRASP, the obtained experimental results were not as good (in terms of performance) as those of ILS[18].

4.9 SRCC experimental results

The RCC ILP formulation becomes numerically more difficult to solve with the symmetric definition and, as a consequence, the authors in Figueiredo and Moura (2013) were able to calculate the SRCC solution only for smaller networks, namely the literature instances in (i). We have applied these results to validate the quality of the SRCC solutions returned by our metaheuristics.

When adapting the ILS metaheuristic to solve the SRCC problem, instead of the CC problem, there is the need to change the objective function that evaluates the partition. When it comes to the SRCC problem, this evaluation becomes more costly, as it demands the solution of a series of minimization problems defined in equation 6 (Section 2.2). In order to reduce this computational cost, the algorithm manipulates matrices with precalculated imbalance values between clusters, and these matrices are incrementally updated whenever an improved solution is found. The matrices are then used to analyze the solutions visited at each local search iteration. In this way, our ILS heuristic is able to provide an efficient solution to the SRCC Problem, applying the same methods (and corresponding configurations) used to solve the CC Problem, with one major advantage over exact methods: it can process larger graph files.

Therefore we have extended the metaheuristics explained in the previous section to deal with this new measure, modifying the applied objective function. Also, in order to provide SRCC measures for larger graph files, we ran the ILS procedure restricting the number of clusters in the SRCC solution in order to match the exact number of clusters (k) found by the CC solution previously obtained by the program. This way, we were able to compare the imbalance of the solutions returned by both ILS algorithms (CC and SRCC).

After solving the SRCC problem, one can observe that, when compared to the CC problem, the imbalance measures obtained with SRCC are sometimes smaller, as can be seen on Figure 4 (UNGA instances). A reason why the CC problem tends to over-evaluate the imbalance is related to penalizing relationships associated with mediation processes that occur in the network.

5 Analysis of structural balance on large real-world social networks

In this section, we apply the previously presented ILS algorithms to efficiently solve the CC and SRCC problems, and analyze the obtained solutions to the United

[18] The table containing the performance comparison between Parallel GRASP and Parallel ILS is available in http://www.ic.uff.br/~yuri/files/CC-ADDcomp.zip

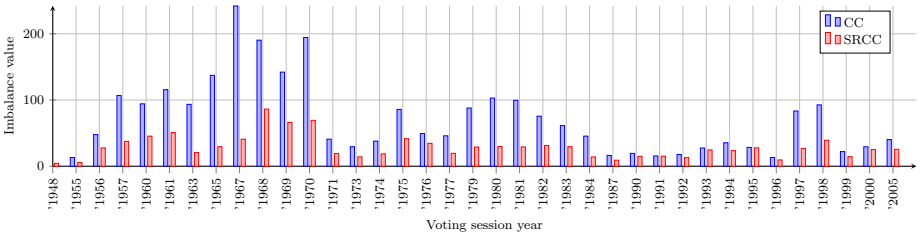

Fig. 4 CC and SRCC imbalance measures of UNGA instances listed in Section 4.2-(ii). In this graph, we only list the years when the CC and SRCC imbalance values differ. For a full report of these results, see the complementary material in http://www.ic.uff.br/~yuri/files/CCcomp.zip.

Nations General Assembly Voting Data and three large signed social networks (Epinions, Slashdot and Wikipedia) available in Leskovec and Krevl (2014).

5.1 United Nations General Assembly (UNGA) Voting Data

As noted in the introduction, some authors have applied different signed graph clustering methods to networks of international alliances and disputes. Traag and Bruggeman (2009) analyzed international relations taken from the Correlates of War (Stinnett et al. 2002) data set, while Macon et al. (2012) showed how three different network representations could be used to identify voting groups in UNGA annual sessions.

Likewise, our work is based on the voting on resolutions in the United Nations General Assembly (UNGA), separated by year. Having applied the sequential ILS algorithm using the UNGA instances listed in Section 4.2-(ii) as input, we give a taste of social network analysis that can be done with the obtained CC and SRCC results. [19] We interpret the results obtained on graphs associated with UNGA votes, i.e. groups of countries that minimize network imbalance, in addition to some historical and geopolitical facts.

5.1.1 UNGA CC results

According to our results, in 1946 (first UNGA voting session), the obtained CC solution has Poland, Czechoslovakia, Yugoslavia, Russia, Ukraine and Belarus together in the same group, reflecting the Soviet Union power in Eastern Europe. In the same period, the United States and Cuba appeared together in another group, and such behavior persisted until 1953, the year of the Cuban Revolution (Pérez-Stable 1993). Afterwards, in 1962, year of the Cuban missile crisis (Allison 1969), bipolarity was obvious during the Cold War. Our CC results indicate two clusters: (i) USA, with most Latin American countries, Western Europe, Japan, Taiwan, India, Australia and other Pacific Countries; and (ii) Russia, with Cuba, Poland, Hungary, Czechoslovakia, Albania, Yugoslavia, Bulgaria, Ukraine and many African countries. By 1963, Cuba had moved towards a full-fledged Communist system modeled on the Soviet Union.

[19] For a full report of these results, including the groups of countries in each solution, separated by year, see the complementary material in http://www.ic.uff.br/~yuri/files/CCcomp.zip.

Additional results are linked to Apartheid in South Africa: the country appears isolated in the CC clustering solution for the 1974 voting session. In that year a motion was passed to expel South Africa from the UN, but this was vetoed by France, the United Kingdom and the United States, all key trade associates of South Africa (Nesbitt 2004, McGreal 2006, Dowdall 2009). The results obtained for the graph associated to 1974 also show an approximation of the USSR (Russia) and Arab states (Libya, Sudan, Iran, Iraq, Egypt, Syria, Lebanon, Jordan, Saudi Arabia, Kuwait, Bahrain, Qatar, United Arab Emirates, Oman and Afghanistan), which appear in the same group. This behavior makes sense, since after the mid 1950's and throughout the remainder of the Cold War the Soviets unequivocally supported various Arab regimes in lieu of Israel (Golani 1995, Golan 2010).

Later on, from 1987 to 1991, during the period known as the First Intifada, characterized by Palestinian uprising against the Israeli occupation of the Palestinian Territories (Smith 2010), the CC results show that Israel is always in a small group while most other countries of the world form another group. The USA, however, always appears together with Israel and sometimes other countries like the UK, France and German Federal Republic also appear in the same group.

More recently, regarding the Gaza-Israel conflict, in 2006, Israel and USA appear together, isolated in a group, with the rest of the world in another group. This is probably due to the large-scale conventional warfare beyond the peripheries of the Gaza Strip (Mearsheimer and Walt 2006), in addition to the 2006 Lebanon War (Kreps 2007).

5.1.2 UNGA SRCC results

As explained in the previous section, an interesting analysis of the SRCC problem provides a different view over certain relations that were previously seen as violations to the balance on the network according the original CC problem definition. Since the SRCC problem does not count certain types of ties as violations (for considering some ties belonging to processes occurring in the network, such as positive and negative mediation), the values of imbalance tend to be even lower than those identified by the CC problem solver. Thus it is possible to extract and detect which actors possibly belong to a group of mediators. Taking the UNGA CC and SRCC solution values into account, as can be seen on Figure 4 in Section 4.9, it is possible to identify in which yearly voting sessions the SRCC relaxed imbalance is smaller than the CC imbalance measure. In these cases, there are relationships / ties that were not classified by SRCC as violations of the balance of the network. For example, in the voting session of 1987, we were able to identify a group of countries that acted as positive mediators: Canada, Ireland, Netherlands, Belgium, Luxembourg, France, Spain, Portugal, German Federal Republic, Italy, Norway, Denmark, Iceland, Japan, Australia and New Zealand. This group of sixteen countries has 100% of internal positive relationships and 92% of external positive relationships. Besides this, there are two other clusters: the first one with the USA, Dominica, the UK and Israel, and another one with 138 countries. The year of 1987 is exactly when the First Intifada commenced, so probably the small group with Israel and the large group with most countries of the World are in opposite camps, subject to the mediation of a third group of countries.

The same holds true in the analysis of the SRCC results of 1990. In this case, the group of mediators was composed of the following twelve countries, out of

which eleven were also present in the mediation group of 1987: Canada, Netherlands, Belgium, Luxembourg, Portugal, German Federal Republic, Poland, Italy, Norway, Denmark, Iceland and Japan. This group of twelve countries has 100% of internal positive relationships and 94% of external positive relationships. There are also two other groups. The second one comprising of nine countries (United States of America, Panama, United Kingdom, France, Sao Tome and Principe, Equatorial Guinea, Liberia, Israel and Cambodia) and the third one with 137 nations, including all Arab and Soviet states. The First Intifada lasted until the Madrid Conference in 1991, though some date its conclusion to 1993, with the signing of the Oslo Accords (Munem 2008). With this in mind, an evidence that supports the existence of the aforementioned mediation group is the second phase of the Madrid Peace Process, launched by the United States and Russia in Moscow, in January 1992. Foreign ministers and delegates from thirty six countries - including representatives from the Middle East, Europe, Japan, China and Canada - were involved.

5.2 Epinions, Slashdot and Wikipedia social media networks

In this section, we evaluate three large online signed social networks available in Leskovec and Krevl (2014): (i) the social network of the technology-news website Slashdot[20], where a signed link indicates that one user likes or dislikes the comments of another; (ii) the voting network of Wikipedia (WikiElections[21]), where a signed link indicates a positive or negative vote by one user on the promotion to admin status of another; and (iii) the trust network of Epinions product review website[22], where users can indicate their trust or distrust of the reviews of others.

Table 6 presents different imbalance measures found in the literature for the aforementioned social networks. Facchetti et al. (2011) defines a fraction called δ/m, which can be interpreted as a measure of the relative imbalance of the network, since it consists of the number of frustrated bipartite relationships divided by the total number of relationships of the network. In turn, Chiang et al. (2013) shows the proportion of balanced 3-cycles ($P(C_{li})$), and Leskovec et al. (2010) presents a measure of balanced undirected triads.

Method	Measure	Slashdot	Wikipedia	Epinions
Facchetti et al. (2011)	δ/m	14.76%	14.20%	7.17%
Chiang et al. (2013)	$1 - P(C_{li})$	13.35%	21.65%	9.50%
Leskovec et al. (2010)	% unbalanced triads	8.80%	9.10%	5.90%
ILS-CC	$\%I(P)$	14.04%	14.32%	7.89%
ILS-SRCC	$\%RI(P)$	13.90%	5.37%	7.81%

Table 6 Comparison of imbalance measures from different authors and our results (ILS-CC and ILS-SRCC).

By applying the best parallel version of the ILS algorithm, we provide an analysis of the relative imbalance of these networks, which corresponds with the

[20] Slashdot friends or foes network from Feb 21 2009: 82,144 vertices and 549,202 edges.
[21] Wikipedia adminship election data: 7,000 vertices and 100,000 edges.
[22] Epinions who-vote-whom network: 131,828 vertices and 841,372 edges.

results from three different works that have also shown how these networks are extremely balanced when compared to random networks of equivalent size.

According to Facchetti et al. (2011), the fraction of imbalance for the undirected Slashdot network is 14.76%, which is compatible with the results obtained by $ILS-CC$. Our metaheuristic has reached a solution with a relative imbalance ($\%I(P)$) of 14.04% (Table 6). Remark that the divergence in these values is expected and is probably due to the different measures applied. $ILS-CC$ managed to fully process Wikipedia network, resulting in a relative imbalance ($\%I(P)$) of 14.32%, while the paper's measure points to the value of 14.20%. Moreover, the relative imbalance of Epinions network obtained by $ILS-CC$ was 7.89%, being very close to the paper's result (7.17%).

Additionally, according to Chiang et al. (2013), the three former social networks exhibit high levels of balance - we show the corresponding values of imbalance ($1-P(C_{li})$) in Table 6. Even though this measure is not the same objective function of the CC problem, it still indicates that these networks are indeed extremely balanced, which is in accordance with the result of our work. Also remark that the same work presents another measure called $P_0(C_{li})$, which consists of the expected probability of finding balanced 3-cycles in an equivalent random network. The values of P_0 are a lot smaller than the values of P (P_0 equals an average of 60%). This leads to the conclusion that, in random networks of equivalent size, imbalance measures tend to be much higher than the actual imbalance of the three aforementioned social media networks.

Finally, a third work by Leskovec et al. (2010) confirms that the analysed social media networks have high relative balance (i.e. low relative imbalance). In the corresponding line of Table 6, we display the percentage of unbalanced triads, based on the measures provided by the authors.

In summary, the above comparisons indicate that the measure applied in Facchetti et al. (2011) matches our relative imbalance measure derived from the CC problem objective function, while the measures available in Chiang et al. (2013) and Leskovec et al. (2010) underevaluate the imbalance of the networks, as they are based only on unbalanced triangles.

Great part of the imbalance measure of the networks is due to unaccounted mediation relationships. Although $ILS-SRCC$ was not able to detect mediation processes occurring in Slashdot and Epinions networks, the Wikipedia network does indeed present a great quantity of relationships associated with mediation and this translates into a lower imbalance measure ($\%RI(P)$), as shown in Table 6. This points to relative values of (relaxed) imbalance even lower than those found by the CC problem solver. Thus, if we factor the SRCC results into consideration, the Wikipedia network is still more balanced than the measures from Facchetti et al. (2011) and also the CC problem.

6 Discussion and perspectives

The CC and SRCC consist of NP-hard combinatorial optimization problems. We have contributed to their efficient solution by incorporating the ILS metaheuristic, enabling the analysis of structural balance on real-world social networks. We have also demonstrated the potential of social network analysis over UNGA voting session data, Wikipedia election history, Slashdot friends or foes network and

Epinions who-trust-whom online social network. To the best of our knowledge, CC and SRCC measures of large social networks have yet to be calculated.

The multistart ILS metaheuristic, proposed here to solve these problems, is an improvement over the existing GRASP approach to solve the CC Problem, either by outperforming, in processing time, the GRASP metaheuristic proposed earlier, or by improving the solution quality. In fact, our tests have shown that the sequential version of ILS is more efficient than the existing heuristics (Doreian Mrvar Method - implemented in Pajek, and VOTE/BOEM). ILS also provides better solutions when compared to the VNS metaheuristic, showing that, to solve the CC problem, diversification is more efficient than intensification.

Additionally, the parallel ILS with sequential VND, which offers substantial speedups over the sequential algorithm, is the most efficient metaheuristic to solve larger instances, such as those based on real-world signed social networks (Wikipedia, Slashdot and Epinions), as well as completely random and random instances with a predefined community structure.

Regarding the Symmetric RCC Problem, we were able to calculate the Symmetric Relaxed Imbalance (SRI) of all network instances cited in the experiments by applying an extension of the parallel ILS metaheuristic used to solve the CC Problem.

The multistart ILS heuristic can also be incorporated in the efficient solution of instances from other problems. The CC problem can be applied in several areas, in which large instances probably require solving. With this idea in mind, practitioners interested in these applications can apply our heuristic. In future work, we can explore many paths by generalizing various aspects of our research. We intend to use our methods to analyse other real networks that represent a social group in different periods of time. On top of this, it is possible to generalize our heuristic to manage other ways of evaluating the imbalance of the network, which implies solving other relaxed versions of the CC problem. It is also possible to look for overlapping communities.

Acknowledgements The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

Abell, P., M. Ludwig. 2009. Structural balance: a dynamic perspective. *Journal of Mathematical Sociology* **33** 129–155. 2

Aiex, Renata M, Mauricio GC Resende, Celso C Ribeiro. 2007. Ttt plots: a perl program to create time-to-target plots. *Optimization Letters* **1**(4) 355–366. 17, 19

Ailon, Nir, Moses Charikar, Alantha Newman. 2008. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)* **55**(5) 23. 6

Alba, Enrique. 2005. *Parallel metaheuristics: a new class of algorithms*, vol. 47. John Wiley & Sons. 12

Allison, Graham T. 1969. Conceptual models and the cuban missile crisis. *American political science review* **63**(03) 689–718. 23

Bache, K., M. Lichman. 2013. UCI machine learning repository. URL http://archive.ics.uci.edu/ml. 7

Bansal, N., A. Blum, S. Chawla. 2002. Correlation clustering. *Proceedings of the 43rd annual IEEE symposium of foundations of computer science*. Vancouver, Canada, 238–250. 2, 3, 4, 6

Bhattacharya, Anindya, Rajat K De. 2008. Divisive correlation clustering algorithm (dcca) for grouping of genes: detecting varying patterns in expression profiles. *bioinformatics* **24**(11) 1359–1366. 6

Bonchi, Francesco, Aristides Gionis, Antti Ukkonen. 2011. Overlapping correlation clustering. *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 51–60. 6

Brandes, Ulrik, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, Dorothea Wagner. 2008. On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on* **20**(2) 172–188. 2

Brusco, M. 2003. An enhanced branch-and-bound algorithm for a partitioning problem. *British Journal of Mathematical and Statistical Psychology* **56** 83–92. 14

Brusco, M., P. Doreian, A. Mrvar, D. Steinly. 2011. Two algorithms for relaxed structural balance partitioning: linking theory, models and data to understand social network phenomena. *Sociological Methods & Research* **40** 57–87. 8

Brusco, Michael J, Hans-Friedrich Köhn. 2009. Clustering qualitative data based on binary equivalence relations: neighborhood search heuristics for the clique partitioning problem. *Psychometrika* **74**(4) 685–703. 3, 9, 16

Cartwright, D., F. Harary. 1956. Structural balance: A generalization of heider's theory. *Psychological Review* **63** 277–293. 2

Charikara, M., V. Guruswamib, A. Wirtha. 2005. Clustering with qualitative information. *Journal of Computer and System Sciences* **71** 360–383. 6

Chiang, Kai-Yang, Cho-Jui Hsieh, Nagarajan Natarajan, Ambuj Tewari, Inderjit S Dhillon. 2013. Prediction and clustering in signed networks: A local to global perspective. *arXiv preprint arXiv:1302.5145* . 25, 26

Crainic, Teodor Gabriel, Michel Toulouse. 2010. Parallel meta-heuristics. *Handbook of meta-heuristics*. Springer, 497–541. 12

DasGupta, B., G. A. Encisob, E. Sontag, Y. Zhanga. 2007. Algorithmic and complexity results for decompositions of biological networks into monotone subsystems. *BioSystems* **90** 161–178. 2, 6

Davis, J.A. 1967. Clustering and structural balance in graphs. *Human Relations* **20** 181–187. 2

De Nooy, Wouter, Andrej Mrvar, Vladimir Batagelj. 2011. *Exploratory social network analysis with Pajek: Revised and Expanded. 2nd Edition.*, vol. 27. Cambridge University Press. 6, 13

Demaine, E. D., D. Emanuel, A. Fiat, N. Immorlica. 2006. Correlation clustering in general weighted graphs. *Theoretical Computer Science* **361** 172–187. 6

Den Besten, Matthijs, Thomas Stützle, Marco Dorigo. 2001. Design of iterated local search algorithms. *Applications of Evolutionary Computing*. Springer, 441–451. 10

Doreian, P., A. Mrvar. 1996a. A partitioning approach to structural balance. *Social Networks* **18** 149–168. 2, 3, 6

Doreian, P., A. Mrvar. 2009. Partitioning signed social networks. *Social Networks* **31** 1–11. 2, 3, 4, 7, 8, 14

Doreian, Patrick, David Krackhardt. 2001. Pre-transitive balance mechanisms for signed networks*. *Journal of Mathematical Sociology* **25**(1) 43–67. 2

Doreian, Patrick, Andrej Mrvar. 1996b. Structural balance and partitioning signed graphs. *Developments in data analysis* 195–208. 3, 7, 13

Dowdall, Aaron T. 2009. The birth and death of a tar baby: Henry kissinger and southern africa. Ph.D. thesis, University of Missouri–Columbia. 24

Drummond, Lucia, Rosa Figueiredo, Yuri Frota, Mario Levorato. 2013. Efficient solution of the correlation clustering problem: An application to structural balance. YanTang Demey, Herv Panetto, eds., *OTM 2013 Workshops*, *LNCS*, vol. 8186. Springer Berlin Heidelberg, 674–683. 3, 4, 7, 13, 20

Duch, Jordi, Alex Arenas. 2005. Community detection in complex networks using extremal optimization. *Physical review E* **72**(2) 027104. 2

Ekşioglu, Sandra Duni, Panos M Pardalos, Mauricio GC Resende. 2002. Parallel metaheuristics for combinatorial optimization. *Models for Parallel and Distributed Computation*. Springer, 179–206. 12

Elsner, M., W. Schudy. 2009. Bounding and comparing methods for correlation clustering beyond ilp. *ILP'09 Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*. 19–27. 3, 6, 13

Epinions. 1999. Website. URL http://www.epinions.com. Accessed on March 2015. 3

Esmailian, Pouya, Seyed Ebrahim Abtahi, Mahdi Jalili. 2014. Mesoscopic analysis of online social networks: The role of negative ties. *Physical Review E* **90**(4) 042817. 2, 7

Facchetti, G., G. Iacono, C. Altafini. 2011. Computing global structural balance in large-scale signed social networks. *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108. 20953–20958. 2, 3, 4, 15, 25, 26

Feo, Thomas A, Mauricio GC Resende. 1995. Greedy randomized adaptive search procedures. *Journal of global optimization* **6**(2) 109–133. 3, 7, 9, 13

Figueiredo, Rosa, Yuri Frota. 2014. The maximum balanced subgraph of a signed graph: Applications and solution approaches. *European Journal of Operational Research* **236**(2) 473 – 487. 15

Figueiredo, Rosa, Gisele Moura. 2013. Mixed integer programming formulations for clustering problems related to structural balance. *Social Networks* **35**(4) 639–651. 3, 6, 8, 14, 16, 19, 22

Gendreau, M., J.Y. Potvin. 2010. *Handbook of Metaheuristics*. International Series in Operations Research & Management Science, Springer. 12

Giotis, Ioannis, Venkatesan Guruswami. 2006. Correlation clustering with a fixed number of clusters. *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. ACM, 1167–1176. 6

Golan, Galia. 2010. *Yom Kippur and After: The Soviet Union and the Middle East Crisis*, vol. 19. Cambridge University Press. 24

Golani, Motti. 1995. The historical place of the czech-egyptian arms deal, fall 1955. *Middle Eastern Studies* **31**(4) 803–827. 24

Gülpinar, N., G. Gutin, G. Mitra, A. Zverovitch. 2004. Extracting pure network submatrices in linear programs using signed graphs. *Discrete Applied Mathematics* **137** 359–372. 2, 6

Harary, F., M. Lim, D. C. Wunsch. 2003. Signed graphs for portfolio analysis in risk management. *IMA Journal of Management Mathematics* **13** 1–10. 6

Heider, F. 1946. Attitudes and cognitive organization. *Journal of Psychology* **21** 107–112. 2

Huffner, F., N. Betzler, R. Niedermeier. 2010. Separator-based data reduction for signed graph balancing. *Journal of Combinatorial Optimization* **20** 335–360. 6

Inohara, T. 1998. On conditions for a meeting not to reach a deadlock. *Applied Mathematics and Computation* **90** 1–9. 2

Kim, Sungwoong, Choong D Yoo, Sebastian Nowozin, Pushmeet Kohli. 2014. Image segmentation usinghigher-order correlation clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **36**(9) 1761–1774. 2, 6

Kreps, Sarah E. 2007. The 2006 lebanon war: Lessons learned. *Parameters* **37**(1) 72. 24

Kunegis, J., A. Lommatzsch, C. Bauckhage. 2009. The slashdot zoo: mining a social network with negative edges. *WWW'09 Proceedings of the 18th international conference on World wide web*. 741–750. 3, 4

Kunegis, Jérôme, Stephan Schmidt, Andreas Lommatzsch, Jürgen Lerner, Ernesto William De Luca, Sahin Albayrak. 2010. Spectral analysis of signed graphs for clustering, prediction and visualization. *SDM*, vol. 10. SIAM, 559–559. 4

Leskovec, J., D. Huttenlocher, J. Kleinberg. 2010. Signed networks in social media. *CHI'10 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1361–1370. 2, 3, 4, 15, 25, 26

Leskovec, Jure, Andrej Krevl. 2014. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`. 4, 12, 15, 19, 23, 25

Levorato, Mario, Lucia Drummond, Yuri Frota, Rosa Figueiredo. 2015. An ils algorithm to evaluate structural balance in signed social networks. *Symposium on Applied Computing, SAC 2015, Salamanca, Spain - April 13 - 17*. 1117–1122. 4, 7

Lourenço, Helena R, Olivier C Martin, Thomas Stützle. 2003. *Iterated local search*. Springer. 4, 7, 9

Macon, K.T., P.J. Mucha, M.A. Porter. 2012. Community structure in the united nations general assembly. *Physica A: Statistical Mechanics and its Applications* **391** 343–361. 2, 4, 6, 15, 23

McGreal, Chris. 2006. Brothers in arms-israel's secret pact with pretoria. *The Guardian* **7**. 24

Mearsheimer, John J, Stephen M Walt. 2006. The israel lobby and us foreign policy. *Middle East P.* **13**(3) 29–87. 24

Mehrotra, Anuj, Michael A Trick. 1998. Cliques and clustering: A combinatorial approach. *Oper. Res. Lett.* **22**(1) 1–12. 5, 6, 16

Mladenović, N., P. Hansen. 1997. Variable neighborhood search. *Computers & Operations Research* **24**(11) 1097–1100. 9, 10, 13

Munem, Dr. Eng. Baker Abdel. 2008. Canada and peace in the middle east. URL `http://www.palestine1.net/can&p-e.htm`. Accessed on Jan 2015. 25

Nascimento, M. C.V., L. Pitsoulis. 2013. Community detection by modularity maximization using grasp with path relinking. *Computers Operations Research* Available online on March 2013. 13

Nesbitt, Francis Njubi. 2004. *Race for sanctions: African Americans against apartheid, 1946-1994*. Indiana University Press. 24

Newman, Mark EJ. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences* **103**(23) 8577–8582. 2

Pérez-Stable, Marifeli. 1993. *The Cuban revolution: Origins, course, and legacy*. Oxford University Press New York. 23

Ruiz, Rubén, Thomas Stützle. 2007. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research* **177**(3) 2033–2049. 9

Slashdot. 1997. Website. URL http://slashdot.org. Accessed on March 2015. 3

Smith, Charles D. 2010. *Palestine and the Arab-Israeli conflict:[a history with documents]*. Bedford/St. Martin's. 24

Srinivasan, A. 2011. Local balancing influences global structure in social networks. *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108. 1751–1752. 2

Stinnett, Douglas M., Jaroslav Tir, Paul F. Diehl, Philip Schafer, Charles Gochman. 2002. The Correlates of War (Cow) Project Direct Contiguity Data, Version 3.0. *Conflict Management and Peace Science* **19** 59–67. 23

Swamy, Chaitanya. 2004. Correlation clustering: maximizing agreements via semidefinite programming. *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 526–527. 6

Traag, V.A., J. Bruggeman. 2009. Community detection in networks with positive and negative links. *Physical Review E* **80** 036115. 2, 4, 6, 23

Wang, Ning, Jie Li. 2013. Restoring: A greedy heuristic approach based on neighborhood for correlation clustering. *Advanced Data Mining and Applications*. Springer, 348–359. 7

Yang, B., W.K. Cheung, J. Liu. 2007. Community mining from signed social networks. *IEEE Transactions on Knowledge and Data Engineering* **19** 1333–1348. 2, 7, 14, 15

Zhang, Shihua, Rui-Sheng Wang, Xiang-Sun Zhang. 2007. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications* **374**(1) 483–490. 6

Zhang, Z., H. Cheng, W. Chen, S. Zhang, Q. Fang. 2008. Correlation clustering based on genetic algorithm for documents clustering. *IEEE Congress on Evolutionary Computation*. 3193–3198. 3, 7, 14