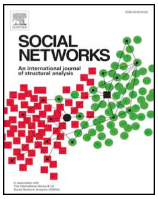
Social Networks 56 (2019) 70–80

Contents lists available at ScienceDirect

Social Networks

journal homepage: www.elsevier.com/locate/socnet



Partitioning signed networks using relocation heuristics, tabu search, and variable neighborhood search

Michael J. Brusco[a], Patrick Doreian[b,c,*]

[a] Florida State University, United States
[b] University of Ljubljana, Slovenia
[c] University of Pittsburgh, United States

ARTICLE INFO

Keywords:
Signed networks
Structural balance
Partitioning
Heuristics
Tabu search
Variable neighborhood search

ABSTRACT

Recently, there have been significant advancements in the development of exact methods and metaheuristics for partitioning signed networks. The metaheuristic advancements have led commonly to adverse implications for multiple restart (multistart) relocation heuristics for these networks. Most notably, it has been reported that multistart relocation heuristics are not computationally feasible for large signed networks with thousands or tens of thousands of vertices. In this paper, we show that combining multistart relocation heuristics with tabu search or variable neighborhood search can rapidly produce partitions of the vertices of signed networks that are competitive with those obtained using existing metaheuristics.

1. Introduction

The problem of partitioning the vertices of a signed network has applications in a variety of different scientific contexts. This includes partitioning for signed social networks studied within the general rubric of structural balance theory with clear substantive concerns originally articulated by Heider (1946) and later extended by Cartwright and Harary (1956); Davis (1967), and Doreian and Mrvar (1996, 2009). This partitioning is important for testing substantive theories (Doreian and Mrvar, 2014), adapting methods (Doreian, 2008), and tracking imbalance over time (Doreian and Mrvar, 2015). Of additional interest is the study of signed networks in other contexts and fields (Facchetti et al., 2011, Huffner et al., 2010; Yang et al., 2007; Kim et al., 2014). Further, Aref et al. (2017) and Levorato et al. (2017) recently noted examples in the physical sciences including: (i) chemistry: the study of fullerene graphs pertaining to carbon allotropes (Došlić and Vukičević, 2007), (ii) biology: measuring the distance of a biological network from monotonicity (Iacono et al., 2010), and (iii) physics: the study of energy states (Kasteleyn, 1963). In the arena of political and social science, signed networks have been analyzed to investigate voting patterns of the United Nations General Assembly (Doreian et al., 2013; Doreian and Mrvar, 2015). Recent literature reviews pertaining to the analysis of signed networks are provided by Tang et al. (2016) and Traag et al. (2018).

For all these applications, getting the partitioning done correctly and efficiently is important, especially so when the available signed network data sets are larger. Broadly, there are a variety of alternative approaches for community detection in the context of signed networks. These include, but are not necessarily limited to, modularity-based approaches (Anchuri and Magdon-Ismail, 2012), spectral clustering (Kunegis et al., 2010), mixture-modeling (Chen et al., 2013), and dynamic model-based algorithms (Yang et al., 2007). Our focus herein is on direct partitioning approaches for signed networks (Doreian and Mrvar, 1996; Bansal et al., 2004; Traag and Bruggeman, 2009) with emphasis on methods that have great potential for partitioning larger signed networks.

Our focus in this paper is on undirected signed networks associated with a vertex set $V = \{1,\ldots, n\}$, edge set $E = \{u, v\}$, and edge weights w_{uv} (for all $\{u, v\} \in E$).[1] The edge set is partitioned into two subsets E^+ and E^-, which correspond to positive and negative edges, respectively. There are several common formulations of optimization problems for the partitioning of signed networks. Perhaps the most general of these is the correlation clustering problem, where the goal is to partition the vertex set into clusters to minimize the total level of *frustration* or inconsistency in the network. While recognizing that there are alternative objective functions, such as weighed functions and positive and negative inconsistencies (Doreian and Mrvar, 1996) and Hamiltonians (Traag and Bruggeman, 2009), we restrict our attention to the popular

* Corresponding author.
E-mail addresses: mbrusco@business.fsu.edu (M.J. Brusco), pitpat@pitt.edu (P. Doreian).

[1] The methods described in this paper can also be applied to directed networks. The assumption of undirected networks is made for consistency with the network problems analyzed later in the paper.

https://doi.org/10.1016/j.socnet.2018.08.007

criterion of minimizing the total level of frustration.

We note that the frustration index is formally identical to the line index of imbalance used in structural balance partitioning but only for two clusters. Moving beyond partitions having only two clusters is critical. With correlation clustering, the number of clusters is not specified in advance. Correlation clustering is based on principles of weak structural balance described by Davis (1967) building on the work of Cartwright and Harary (1956). Frustration occurs when two vertices sharing a positive edge are in different clusters, or when two vertices sharing a negative edge are in the same cluster. Mathematical programming formulations for the correlation clustering problem have been described by Figueiredo and Moura (2013) and Aref et al. (2017). Metaheuristics for large networks have also been proposed and include genetic algorithms (Zhang et al., 2010) and iterated local search (Levorato et al., 2017).

Some authors have focused on a variation of the correlation clustering problem whereby a desired number of clusters, K, is prespecified (Brusco and Steinley, 2010; Doreian and Mrvar, 1996; Giotis and Guruswami, 2006). A branch-and-bound algorithm for this *K-balance partitioning* problem was proposed by Brusco and Steinley (2010); however, it is only scalable for small networks ($n < 30$). Preferable exact approaches based on mathematical programming have been proposed by Figueiredo and Moura (2013) and Aref et al. (2017). For larger problems, Doreian and Mrvar (1996) proposed a relocation heuristic. The special case of $K = 2$ for the K-balance partitioning problem is of considerable theoretical interest because it corresponds to minimization of the frustration index via a partition of the vertex set so as to obtain the minimum number of edges that must be removed so as to bring the network into balance. Aref et al. (2017) present a formal treatment of this problem within a graph coloring framework and provide an efficient exact solution approach using integer linear programming.

Despite advancements in the development of exact procedures (Aref et al., 2017; Brusco and Steinley, 2010; Brusco et al., 2011; Figueiredo and Moura, 2013), heuristic procedures remain useful for the partitioning of large signed networks (e.g., $n > 1000$). The relocation heuristic developed by Doreian and Mrvar (1996) is one of the most general heuristic procedures for partitioning signed networks. It begins with an initial (often randomly generated) partition that is refined by two local-search operations: (i) relocation of each vertex from its current cluster to one of the other clusters, and (ii) pairwise interchanges (or exchanges) of the cluster memberships for each pair of vertices that are not currently in the same cluster. The implementation of the relocation heuristic in the Pajek software system (see de Nooy et al., 2011) has been shown to perform well on small real-world instances of K-balance partitioning problems (Brusco and Steinley, 2010; Figueiredo and Moura, 2013); however, its performance deteriorated somewhat for synthetic random networks with roughly 50 vertices (Figueiredo and Moura, 2013). Of greater concern is the reported computational impracticality of the relocation heuristic for problems with $n > 1000$ vertices (Levorato et al., 2017). This problem needs to be addressed.

In an effort to tackle large instances of correlation clustering and K-balance partitioning problems, several researchers have focused on the development of metaheuristics such as genetic algorithms (Goldberg, 1989), greedy randomized adaptive search procedure (GRASP: Feo and Resende, 1995), variable neighborhood search (Hansen and Mladenovic, 1997), and iterated local search (Lourenco et al., 2003, 2010). More specifically, genetic algorithm approaches have been designed by Zhang et al. (2008) and Ma et al. (2015). Drummond et al. (2013) obtained promising results with the GRASP method. Most recently, however, Levorato et al. (2017) showed that their implementation of iterated local search convincingly outperformed the greedy relocation heuristics of Doreian and Mrvar (1996) and Elsner and Schudy (2009), GRASP, and an implementation of variable neighborhood search.

Our specific operational goals in this paper are twofold. First, we demonstrate that an efficient multistart implementation of the relocation heuristic is often scalable for networks with far more than 1000 vertices. Second, and more importantly, we show that the relocation heuristic is also an effective engine for metaheuristics for K-balance partitioning and, by extension, correlation clustering. More specifically, we recommend a two-phase approach. The first phase uses the multistart relocation heuristic to establish a good initial solution and either tabu search (Glover, 1989, 1990; Glover and Laguna, 1993) or variable neighborhood search (Mladenovic and Hansen, 1997) is used in the second phase to refine the solution. We evaluate this procedure using test problems considered by Levorato et al. (2017), which are slices of the Slashdot zoo data (Leskovec et al., 2010). We also apply the method to the full Slashdot zoo network and the Wiki elections data (Leskovec et al., 2010). Our results compare very favorably to the methods evaluated by Levorato et al. (2017). Moreover, we show that high-quality solutions are achievable using *far fewer* clusters than reported by Levorato et al. (2017).

Section 2 presents a formal presentation of the K-balance partitioning problem. This section also includes a description of the relocation heuristic, tabu search, and variable neighborhood search procedures. Section 3 presents an evaluation of the two-phase procedure using the test problems from the Levorato et al. (2017) study. The paper concludes in section 4 with a summary of the findings and suggestions for future research.

2. Generalized structural balance partitioning (K-balance partitioning)

2.1. The optimization problem

We recall our previous definitions of V, E^+, E^-, and w_{uv} as the set of n vertices, set of positive undirected edges, set of undirected edges, and edge weights, respectively. The K-balance partitioning problem seeks a partition, $P = \{S_1, \ldots, S_K\}$, of the vertex set into $K \geq 2$ clusters, where S_k contains the vertices assigned to cluster k for all $1 \leq k \leq K$ clusters. The goal to find the partition P from the set of all possible partitions (Π) of n vertices into K clusters so as minimize the following objective criterion function:

$$Z(P) = \sum_{k=1}^{K} \sum_{\substack{(\{u,v\} \in E^-) \wedge \\ (\{u,v\} \in S_k)}} w_{uv} + \sum_{1 \leq k < l \leq K} \sum_{\substack{(\{u,v\} \in E^+) \wedge \\ \left[\begin{smallmatrix} (u \in S_k \wedge v \in S_l) \vee \\ (u \in S_l \wedge v \in S_k) \end{smallmatrix}\right]}} w_{uv} \tag{1}$$

The first term in Eq. (1) is a summation of the weights of vertex pairs $\{u, v\}$ whereby u and v are both in the same cluster and are associated with a negative edge. The second term in the equation is the summation of the weights of vertex pairs $\{u, v\}$ whereby u and v are in different clusters and are associated with a positive edge. Therefore, the objective criterion function value, Z(P), is a measure of the total amount of inconsistency with perfect structural balance.

There are several possible exact solution approaches for finding the partition P that minimizes $Z(P)$. One approach, complete enumeration, is to compute $Z(P)$ for all partitions $P \in \Pi$ and select the partition that yields the minimum value. The number of partitions in Π is a Stirling number of the second kind and precludes complete enumeration for even modest values of n and K (e.g., $n = 20$ and $K = 4$). Alternatively, an implicit enumeration scheme based on branch-and-bound programming was developed by Brusco and Steinley (2010); however, it too is limited to relatively small problems. Perhaps the most robust exact procedure is based on mixed integer linear programming (Figueiredo and Moura, 2013), yet this approach is also limited to problems with $n < 50$. In light of the limitations of extant exact procedures, there is a necessary reliance on heuristic methods for large problems.

2.2. Relocation heuristic (RH)

Doreian and Mrvar (1996) presented a relocation heuristic for partitioning signed networks based on the generalized structural balance criterion in Eq. (1). Their method begins with an initial partition, P, as input. Next all possible transfers of a vertex from its current cluster to each of the other clusters are evaluated. Any transfer that improves the value of $Z(P)$ is accepted. This transfer phase is complete when no vertex can be moved from its current cluster to one of the clusters and thereby improve $Z(P)$. The next phase evaluates all possible exchanges (or pairwise interchanges) of cluster memberships for pairs of vertices not currently in the same cluster. Again, any vertex exchange that improves $Z(P)$ is accepted. This phase terminates when no possible exchange of vertices can further reduce $Z(P)$. The relocation heuristic terminates when there is no transfer or exchange that will reduce $Z(P)$. The resulting partition is, therefore, locally optimal with respect to transfers and exchanges, but is not guaranteed to be a global minimum. There is also the problem of having multiple equally well-fitting partitions for specific values of K.

Brusco and Steinley (2011) and Brusco et al. (2013) have determined that the exchange phase of the relocation heuristic is ineffective and should not be incorporated in the relocation heuristic. There are two reasons for their recommendation. First, the exchange process seldom improves the result obtained by the transfer heuristic. Second, and more importantly, the exchange heuristic exhausts a large amount of computation time, even for modestly-sized n (e.g., a few hundred vertices). The number of trial solutions evaluated by each complete cycle of the transfer heuristic is $n(K\text{-}1)$. The number of trial solutions associated with the exchange heuristic depends on the relative sizes of the clusters. Brusco and Steinley (2011) noted that, assuming equal cluster sizes of n/K, the number of exchanges evaluated in a complete cycle is $(K(K\text{-}1)/2) \times (n/K)^2$. In addition to fewer trial solutions, transfers also tend to require fewer computational operations than exchanges with respect to the evaluation of a trial solution. When a vertex is transferred from one cluster to another, the effect on the criterion function can be evaluated by examining the changes in the number of inconsistencies that might accrue due the edges that vertex shares with other vertices in its old cluster and it new cluster. By contrast, an exchange requires the relocation of two vertices simultaneously and accordingly, the computational requirement to assess changes in the criterion function is greater. In light of these computational differences, if an analyst is implementing a multistart relocation heuristic for some fixed time limit, then it is far more advantageous to exclude the exchange phase and allow the heuristic to get many more restarts within the time limit. Increasing the number of restarts is essential when using heuristics.

Here, we implement a multistart version of the relocation heuristic that excludes the exchange phase which has been shown to not improve the partitioning outcomes (Brusco and Steinley, 2007, 2011). The pseudo code for the heuristic is displayed in Fig. 1. To generate the initial partition for each restart, K vertices are randomly selected based on a uniform distribution with equal probability for each unselected vertex. A check is made to assure that no two vertices having the exact same edge set are selected. The K selected vertices are referred to as cluster exemplars. To form the initial partition, the remaining $n - K$ vertices are assigned to the cluster associated with the exemplar to which their neighborhood is most similar. Neighborhood similarity between two vertices u and v is computed by counting the number of other vertices to which u and v each share a positive (or negative) edge. The initial partition is then refined using the transfer phase of the relocation heuristic. The relocation heuristic is permitted to restart with a different randomly-constructed initial solution until a time limit is reached. The partition, P^*, resulting in the minimum value of the criterion function, $Z(P^*)$, is stored.

The multiple restart implementation of the relocation heuristic performs very well for n and K of modest size. However, as n and K increase, the performance begins to degrade, most often, seriously. This degradation in performance is not unique to structural balance partitioning, but also occurs for other clustering problems, such as K-means clustering (Brusco and Steinley, 2007), p-median clustering (Hansen and Mladenovic, 1997), and clique partitioning (Brusco and Kohn, 2009). It is also the case for combinatorial optimization problems in general (Lourenco et al., 2003, 2010). It is for this reason that metaheuristics are commonly recommended to improve performance.

2.3. Tabu search

Originally developed by Glover (1989, 1990), tabu search is a metaheuristic approach that facilitates the escape from local optima by forbidding some neighborhood moves for a prescribed number of local-search operations (see Glover and Laguna, 1993 for an extensive treatment of tabu search). Tabu search algorithms have been designed for a variety of partitioning problems, including K-means clustering (Pacheco and Valencia, 2003), p-median clustering (Hansen and Mladenovic, 1997), and clique partitioning (De Amorim et al., 1992). Within the context of network analysis, Borgatti and Everett (1997) applied tabu search in their FACTIONS program for analyzing two-mode network data. Subsequently, Brusco and Steinley (2011) developed a tabu search algorithm for deterministic two-mode blockmodeling of social networks.

Tabu search implementations for clustering problems commonly build a tabu list that forbids the movement of a given vertex v to a given cluster k for a fixed number of iterations, τ (see Pacheco and Valencia, 2003; Brusco and Steinley, 2011). We originally implemented tabu search in this manner but found the performance to be unacceptable, presumably because of the sparsity of the networks in our study. An alternative implementation that placed vertex v on the tabu list for a fixed number of iterations, τ, performed much better. That is, each time a vertex was moved from one cluster k to cluster l, that vertex was placed on the tabu list and could not be relocated to any cluster (not just cluster k) for τ iterations. The pseudo code for the tabu search heuristic is provided in Fig. 2.

The best-found partition obtained by the relocation heuristic serves as the initial partition for the tabu search heuristic. An n-dimensional vector, $\mathbf{g}$, with elements $g(v)$ representing the number of iterations for which vertex v remains forbidden is initialized to zero. At each iteration, all possible transfers of vertices $(v : g(v) = 0)$ to a different cluster are evaluated and the transfer that provides the largest improvement (or smallest worsening) of the criterion function is implemented. Denoting v^* as the vertex transferred, we set $g(v^*) = \tau$. If a new best-found criterion function value is obtained, then all elements of $\mathbf{g}$ are reset to zero; otherwise, all non-zero values of $\mathbf{g}$ are reduced by one. We tested several values for τ using guidelines from previous studies (De Amorim et al., 1992; Pacheco et al., 2003; Brusco and Kohn, 2009; Brusco and Steinley, 2011) and ultimately selected $\tau = n/8$ for our implementation. It is typically common to apply a termination criterion based on the number of iterations with no improvement in the criterion function. However, in our implementation, we apply tabu search to a starting solution obtained by the relocation heuristic and let the algorithm run until a prespecified time limit is reached.

2.4. Variable neighborhood search

There are several variants of heuristic procedures that seek to circumvent multiple restarts by engaging a vigorous search within the neighborhood of a local search. One such method is iterated greedy search, which destroys a solution by making it infeasible and then applies a construction heuristic to the solution to return it to feasibility (Jacobs and Brusco, 1995; Ruiz and Stutzle, 2007, 2008). A closely related alternative is iterated local search (Lourenco et al., 2003, 2010), which does not create an infeasible solution, but does perturb the solution to suboptimality and then returns it to local optimality using a

```
Set time_limit, $Z(P^*) = \infty$
While time < time_limit
    randomly generate a partition $P$
    ***BEGIN RELOCATION HEURISTIC***
    improv = 0
    while improv = 0
        improv = 1
        for $v = 1$ to $n$
            $h = l : v \in S_l$
            if $|S_h| > 1$ then
              for $1 \leq k \neq h \leq K$
                  $Q = P \setminus \{S_h \cup S_k\} \cup \{S_h \setminus \{v\}\} \cup \{S_k \cup \{v\}\}$
                  if $Z(Q) < Z(P)$ then
                      $P = Q$
                      $h = k$
                      improv = 0
                  end if
              next $k$
            end if
        next $v$
    end while
    ***END RELOCATION HEURISTIC***
    if $Z(P) < Z(P^*)$ then
        $Z(P^*) = Z(P)$
        $P^* = P$
    end if
end While
```

Fig. 1. Pseudo code for the multistart relocation heuristic.

local-search heuristic. As noted previously, Levorato et al. (2017) have recently reported excellent results using iterated local search for the closely related correlation clustering problem.

Variable neighborhood search, which was developed by Mladenovic and Hansen (1997), is similar to iterated local search; however, it enforces greater control on the size of the neighborhood around the current incumbent solution wherein the search takes place. The method has received tremendous attention in combinatorial optimization in general, as well cluster analysis in particular (Hansen and Mladenovic, 1997; Hansen and Mladenovic, 2001). Variable neighborhoods search has also been successfully employed for two-mode blockmodeling of social networks (Brusco et al., 2013) and was also evaluated for correlation clustering by Levorato et al. (2017).

Here, we adapt the variable neighborhood search procedure used by Brusco et al. (2013) for two-mode blockmodeling to refine the solutions obtained by the multistart relocation heuristic for generalized structural

```
Input $Z(P^*)$, $P^*$
Set time_limit, $\tau = n/8$, $g(v) = 0$ for $1 \leq i \leq n$.
$P = P^*$
While time < time_limit
    $\Delta^* = \infty$
    for $v = 1$ to $n$
        $h = l : v \in S_l$
        if $|S_h| > 1$ and $g(v) = 0$ then
          for $1 \leq k \neq h \leq K$
              $Q = P \setminus \{S_h \cup S_k\} \cup \{S_h \setminus \{v\}\} \cup \{S_k \cup \{v\}\}$
              if $Z(Q) - Z(P) < \Delta^*$ then
                  $Q^* = Q$
                  $\Delta^* = Q^*$
                  $v^* = v$
              end if
          next $k$
        end if
    next $v$
    $P = Q^*$
    $g(v^*) = \tau$
    if $Z(P) < Z(P^*)$ then
        $Z(P^*) = Z(P)$
        $P^* = P$
        $g(v) = 0$ for $1 \leq v \leq n$
    else
        $g(v) = g(v) - 1$ for $v : g(v) > 0$
    end
end while
```

Fig. 2. Pseudo code for the Tabu search heuristic.

```
Input $Z(P^*)$, $P^*$
Set time_limit, $ymin$ = .005, $ymax$ = .20, $ystep$ = .005, $ypert$ = $ymin$
While time < time_limit
    $P = P^*$
    for $v$ = 1 to $n$
        $h = l : v \in S_l$
        if $|S_h| > 1$ then
            $rnd$ = uniform random number on [0,1]
            if $rnd < ypert$ then
                randomly choose $k$ from the values $1 \le k \ne h \le K$
                $P = P \setminus \{S_h \cup S_k\} \cup \{S_h \setminus \{v\}\} \cup \{S_k \cup \{v\}\}$
            end if
        end if
    next $v$
    ***BEGIN RELOCATION HEURISTIC to refine $P$
    while improv = 0
        improv = 1
        for $v$ = 1 to $n$
            $h = l : v \in S_l$
            if $|S_h| > 1$ then
                for $1 \le k \ne h \le K$
                    $Q = P \setminus \{S_h \cup S_k\} \cup \{S_h \setminus \{v\}\} \cup \{S_k \cup \{v\}\}$
                    if $Z(Q) < Z(P)$ then
                        $P = Q$
                        $h = k$
                        improv = 0
                    end if
                next $k$
            end if
        next $v$
    end while
    ***END RELOCATION HEURISTIC***
    if $Z(P) < Z(P^*)$ then
        $Z(P^*) = Z(P)$
        $P^* = P$
        $ypert = ymin$
    else
        $ypert = ypert + ystep$
        if $ypert > ymax$ then
            $ypert = ymin$
        end if
    end if
end while
```

Fig. 3. Pseudo code for the variable neighborhood search heuristic.

balance partitioning. One of the key reasons for selecting this method for adaptation is that it can embed the relocation heuristic described in the previous section as the engine of the search process. This helps strengthen our claim that relocation heuristics should *not* be abandoned for blockmodeling because they are often a key component of metaheuristics.

The pseudo code for the variable neighborhood search algorithm is displayed in Fig. 3. The algorithmic process begins by taking the best-found partition obtained by the relocation heuristic as the incumbent partition. Next the parameters of the variable neighborhood search algorithm are initialized. The parameter *ypert* is the key parameter of the process that perturbs the current incumbent partition into a neighboring partition. It is the probability that any given vertex will be relocated from its current cluster to a different cluster. Accordingly, this parameter controls the size of the neighborhood that will be searched. If *ypert* = .01, then we would expect 1% of the vertices to be moved from their current cluster to a different cluster. However, if *ypert* = .10, then we would expect 10% of the vertices to be relocated and thus the move is to a more distant neighborhood. The parameter *ypert* is initialized to *ymin*, which corresponds to the minimum neighborhood size. The parameters controlling the maximum probability (*ymax*) and the step size for moving from the minimum to the maximum probability (*ystep*) are also initialized.

The next step is the perturbation process. A probabilistic process is used to determine if each vertex is to be moved from its current cluster. The probability of a move is *ypert*. If a decision is made to relocate a vertex from its current cluster (*h*), then one of the other *K*-1 clusters is selected at random (equal probability for each of these possible clusters) to be the new location for the vertex. Applying this process to each vertex creates a new partition, *P*, that is almost always not a local minimum. To refine *P* to a local minimum, the same relocation heuristic described in Fig. 1 is applied to return *P* to local optimality.

Next, *P* is compared to the incumbent partition, *P**. If $Z(P) < Z(P^*)$, then *P* replaces *P** as the new incumbent partition and *ypert* is reset to *ymin*. If $Z(P) \ge Z(P^*)$, then *ypert* is incremented by *ystep* so as to result in the search of a larger neighborhood. If the increment results in *ypert* > *ymax*, then *ypert* is reset to *ymin*. If the time limit has not been reached, then control returns to perturbation process. Once the time limit is reached, *P** is returned as best-found partition with criterion function value $Z(P^*)$.

There are two critical differences in our implementation of variable neighborhood search relative to the implementation by Brusco et al. (2013) for two-mode blockmodeling. First, in the Brusco et al. (2013) implementation, variable neighborhood search was applied after each restart of the relocation heuristic for a random initial partition. This process was found to be much too wasteful for the appreciably larger networks that we consider. Second, because of this distinction, the termination of the variable neighborhood search process in the Brusco et al. (2013) implementation occurred when *ymax* was reached the first time. In the current implementation, attaining *ymax* just results in resetting *ypert* to *ymin*. Termination occurs when the time limit is reached. Third, we found that reducing the *ymin*, *ystep*, and *ymax* parameter settings relative to those in the Brusco et al. (2013) led to better performance. In particular, the values of the parameters used herein are *ymin* = .005, *ymax* = .2, and *ystep* = .005.

3. Computational analysis

3.1. Test problems

There are two goals associated with our computational analyses. The first goal is to demonstrate that the multistart relocation heuristic is computationally feasible for large signed networks, particularly when the exchange portion of the algorithm is eliminated. The second goal is to show that the variable neighborhood search heuristic, when using a starting solution obtained by the multistart relocation heuristic, can produce partitions that are competitive with the best-known solutions for these larger networks. Both goals were realized. Admittedly, the term 'large' is rather subjective and some clarification is required. Our definition of small signed networks would include those obtained for relational ties in dormitories (Lemann and Solomon, 1952), monasteries (Sampson, 1968), or fraternities (Newcomb, 1961), which have been analyzed by a number of authors (Brusco and Steinley, 2010; Brusco et al., 2011; Doreian et al., 2005; Doreian and Mrvar, 1996, 2009; Figueiredo and Moura, 2013). The networks typically have roughly $n = 20$ vertices, and we would consider network with $n = 50$ or fewer vertices as small. The United Nations voting networks commonly have a few hundred vertices and would be considered medium-sized. We define large signed networks as those with roughly $n = 1000$ or more vertices. These would include the Wiki elections network data ($n \approx 7000$ and $|E| \approx 100{,}000$) and the Slashdot zoo network ($n \approx 80{,}000$ and $|E| \approx 500{,}000$). See also Batagelj et al. (2014) for a classification of the sizes of networks.

Levorato et al. (2017) have provided a tremendous service by making suite of large signed networks readily available for researchers (http://www.ic.uff.br/~yuri/files/CCinst.zip).[2] These authors produced an undirected network for the Slashdot zoo data and subsequently sliced the full network into smaller undirected networks of the following sizes: (i) $n = 600$, $|E| = 1917$, (ii) $n = 1000$, $|E| = 5991$, (iii) $n = 2000$, $|E| = 20{,}815$, (iv) $n = 4000$, $|E| = 49{,}532$, (v) $n = 8000$, $|E| = 109{,}705$, and (vi) $n = 10{,}000$, $|E| = 139{,}071$. Our primary focus is on these six test problems. For comparison purposes, we also use average criterion function values, confidence intervals, and numbers of clusters reported by Levorato et al. (2017) for their implementation of iterated local search. In addition, we also report results for the full undirected Slashdot network ($n = 82{,}144$, $|E| = 498{,}532$) and an undirected version of the Wiki elections data ($n = 8298$, $|E| = 106{,}548$) produced by Levorato et al. (2017).

3.2. Hardware and software platforms

The relocation heuristic and variable neighborhood search procedure were written in Fortran and compiled using the GNU Fortran compiler that is available with the R Tools software system. All computational results were obtained using a desktop computer with an i7 processor and 16 GB of RAM.

For each test problem, a time limit of up to 10 min was specified. The first half of the available time limit was allocated to the multistart relocation heuristic. The best objective criterion function value obtained by the heuristic within 10, 30, 60, 120, 180, 240, and 300 s was stored. The total number of restarts realized within the time limit was also recorded. The variable neighborhood search process was executed for the second half of the allotted time limit and the best value was stored.

The smallest of the test problems ($n = 600$ and $n = 1000$) were allotted a time limit of 60 s. The somewhat larger problems corresponding to $n = 2000$ and $n = 4000$ were allotted 300 s. The $n = 8000$ and $n = 10{,}000$ test problems were allotted 600 s. Likewise, the full Slashdot and Wiki elections networks were also allotted 600 s.

The number of clusters for the K-balance partitioning problem must be prespecified. For each test problem, we began with $K = 2$ clusters and continued incrementing the number of clusters until several successive increments of K did not produce any further improvement in the objective criterion function value.

3.3. Results for the Slashdot zoo slices

Table 1 reports results illustrating the advantages of excluding the exchange routine from the algorithm from the relocation heuristic. For the $n = 2000$ and $n = 4000$ test problems, the top panel of Table 1 reports the criterion function values after 150 s for the relation heuristic when using only transfers (RH) in the neighborhood search process, as well as when using both transfers and exchanges (RHwE). Similar results are reported for the $n = 8000$ and $n = 10{,}000$ test problems for a 300-second time limit. The bottom panel of Table 1 reports the total number of restarts achieved by RH and RHwE within the allotted time limits.

There are only three instances in Table 1 where RHwE yielded a better criterion function value than RH: (i) $K = 14$ for the $n = 4000$ dataset, (ii) $K = 2$ for $n = 8000$ dataset, and $K = 2$ for $n = 10{,}000$ dataset. The difference in the $Z(P)$ values obtained by RH and RHwE for these three instances were 6, 1, and 2 inconsistencies, respectively. Contrastingly, RH produced a better criterion function value than RHwE for 44 of the remaining 45 instances in Table 1 (the two algorithms produced the same value of $Z(P) = 2190$ for the $n = 2000$ dataset at $K = 5$). Moreover, for 27 of the 44 instances, the $Z(P)$ value obtained by RH was 10 or more inconsistences better than the corresponding value achieved by RHwE. The RH algorithm outperformed RHwE by 20 or more inconsistencies for 12 of the instances. While the numerical differences are sometimes smaller, it seems that the inclusion of exchanges compromises the minimization of the criterion function.

For the $n = 2000$ test problems, the exclusion of the exchange routine results in RH achieving 11.7–44.0 times the number of restarts within the 150-second limit when compared to RHwE. For the $n = 4000$, $n = 8000$, and $n = 10{,}000$ test problems, the ratio of RH restarts to RHwE restarts is even more dramatic, with ranges (across the different values of K) of 15.7–81.8, 29.8–164.1, and 36.8–243.7, respectively. It is also interesting to observe that a doubling of the number of vertices commonly lead to *roughly* a doubling of the restart ratio. For example, at $K = 5$ for the $n = 2000$ dataset, the number of restarts for RH and RHwE were 9527 and 435, respectively, for a ratio of $9527/435 = 21.9$. Similarly, at $K = 5$ for the $n = 4000$ dataset, the number of restarts for RH and RHwE were 3686 and 81, respectively, for a ratio of $3686/81 = 45.5$. When n doubles again, at $K = 5$ for the $n = 8000$ dataset, the number of restarts for RH and RHwE were 3119 and 36, respectively, for a ratio of $3119/36 = 86.6$. For the $n = 10{,}000$ test problems, the RHwE implementation is only achieving 20–25 restarts within the 300-second time limit for most values of K. Accordingly, for larger n, the use of the exchange routine becomes computationally impractical. Our comparison of methods hereafter is limited to RH and its integration with tabu search and variable neighborhood search.

The results for the Slashdot zoo tests problems on the interval $600 \leq n \leq 4000$ are reported in Table 2. For the $n = 600$ test problem, the best-known criterion function value of $Z(P) = 109$ was obtained by the variable neighborhood search for all values of K on the interval $3 \leq K \leq 6$. The relocation heuristic also matched this criterion function value for all values of K on this interval within its 30-second time limit, as

[2] The explicit provision of test problems is important because different versions of the Slashdot zoo and Wiki elections data are available from different sources. Moreover, authors commonly use different adaptations of these networks (e.g., by applying different rules to transform the networks from directed to undirected).

Table 1
An evaluation of the impact of excluding the exchange routine.

	n = 2000		n = 4000		n = 8000		n = 10,000	
K	RH	RHwE	RH	RHwE	RH	RHwE	RH	RHwE
2	2298	2300	6543	6548	16959	16958	21792	21790
3	2214	2218	6274	6284	16262	16272	20830	20850
4	2191	2196	6227	6228	16149	16166	20660	20680
5	2190	2190	6212	6227	16115	16134	20612	20618
6	2187	2188	6213	6216	16102	16121	20598	20606
7	2187	2189	6213	6242	16092	16113	20590	20614
8	2190	2191	6210	6214	16087	16088	20587	20600
9	2191	2193	6208	6221	16089	16118	20584	20596
10	2189	2191	6211	6224	16090	16114	20591	20608
11	–		6209	6221	16086	16114	20583	20612
12	–		6213	6215	16093	16113	20583	20598
13	–		6214	6215	16091	16110	20579	20600
14	–		6215	6209	16087	16134	20586	20599
2	29243	664	11371	139	9681	59	7312	30
3	17013	480	6567	95	5563	36	4227	21
4	12121	406	4697	80	3974	39	3056	21
5	9527	435	3686	81	3119	36	2413	20
6	7836	441	3062	81	2567	36	1983	22
7	6669	431	2604	71	2176	36	1689	22
8	5718	419	2259	78	1905	38	1477	21
9	5093	416	1989	76	1684	37	1301	23
10	4539	387	1776	78	1496	38	1170	20
11			1607	84	1352	33	1051	22
12			1464	78	1229	37	958	23
13			1336	67	1128	34	879	22
14			1228	78	1044	35	810	22

Note: The top panel contains the $Z(P)$ values for the relocation heuristic with transfers only (RH) and the relocation heuristic with transfers and exchanges (RHwE). The bottom panel contains the total number of restarts obtained within a time limit of 150 s for the n = 2000 and n = 4000 test problems, or within a 300-second limit for the n = 8000 and n = 10,000 test problems.

well as within 10 s for 3 ≤ K ≤ 5. Within its 30 s time limit. The multistart relocation heuristic realized 109,824 restarts for K = 2. Although the number of restarts within the limit declines as K increases, there were still 23,443 restarts realized at K = 6.

The key column to focus on is the one labeled "$Z(P)$ VNS". For n = 600, the first instance of the best solution, $Z(P)$ = 109, is at K = 3. For n = 1000, the first instance of the best solution, $Z(P)$ = 600, is at K = 4. For n = 2000, the first instance of the best solution, $Z(P)$ = 2184, is at K = 5. One important theorem reported in Doreian et al. (2005, p 305) is that the plot of the value of the criterion function plotted against the number of clusters has a U-shaped distribution. These results are consistent with this. For n = 4000, the first instance of the best solution, $Z(P)$ = 6191, is at K = 7. Notice that 6191 is also found at K = 8, K = 10, and K = 11, but not at K = 9. This is not due to a violation of the U-shaped rule, it is just that the algorithm is an imperfect heuristic. At K = 9 it failed to find 6191 within the allotted time limit. Still, for a 4000-node problem, it is impressive that either 6191 or 6192 was found for all K values between 7 and 12. To be parsimonious, in the sense of using the smallest number of clusters, it seems reasonable to accept the K = 7 solution for further interpretations.

It is also useful to compare the results in the "$Z(P)$ TS" column to those in the "$Z(P)$ VNS" column. Like variable neighborhood search, tabu search frequently improved the solution obtained by the relocation heuristic. However, the tabu search criterion values, although quite good, were frequently slightly worse (larger) than the variable neighborhood search criterion values. For example, tabu search matched the best-found criterion value of $Z(P)$ = 2184 for the n = 2000 test problem at K = 5 and K = 6 but failed to do for K ≥ 7. For the n = 4000 test problem, tabu search failed to match the best-found criterion value of $Z(P)$ = 6191 using any K but did find $Z(P)$ = 6192 for K = 11 and K = 13.

For the n = 1000 test problem, the results reported by Levorato et al. (2017, Table 3, p. 485) indicate the average $Z(P)$ value realized after two hours of execution of their iterated local search procedure (SeqILS) was 600 and there was no variance. The results reported by Levorato et al. (2017, Table 4, p. 488) show that this criterion function value was obtained using K = 18 clusters. Our results in Table 2 reveal that variable neighborhood search obtained the same criterion function value of $Z(P)$ = 600 for 5 ≤ K ≤ 8 clusters. This is an important finding because it shows that, although correlation clustering has the advantage of not requiring the pre-specification of K, it can *use far more clusters than necessary* to get the best criterion function value. At a minimum, this points to a serious problem with their method. The relocation heuristic also produced the best criterion value of $Z(P)$ = 600 for K = 5 and was very close for 6 ≤ K ≤ 8 clusters. Table 1 also shows that relocation heuristic produced at least 4657 restarts (K = 8) within 30 s for the 1000 vertex problem

From the results for the n = 2000 test problem reported by Levorato et al. (2017, Table 3, p. 485 and Table 4, p. 488), we observe an average $Z(P)$ value of 2184.14, a 95% confidence interval of [2183.99 to 2184.29], and K = 31 clusters for a two-hour implementation of their SeqILS method. Table 2 reveals that the variable neighborhood search heuristic found $Z(P)$ = 2184 for all values of K on the interval 5 ≤ K ≤ 10. Therefore, it seems that the variable neighborhood search heuristic is matching the objective function value from correlation clustering but doing with *far fewer* clusters. The tabu search heuristic also found 2184 twice. The relocation heuristic did not find the best criterion function value of 2184 within its allotted 150-second time limit. However, it did produce an excellent starting point for variable neighborhood search with criterion values ranging from 2187 to 2191 for the interval 5 ≤ K ≤ 10.

From the results for the n = 4000 test problem reported by Levorato et al. (2017, Table 3, p. 485 and Table 4, p. 488), we observe an average $Z(P)$ value of 6193.35, a 95% confidence interval of [6192.78 to 6193.92], and K = 44 clusters for a two-hour implementation of their SeqILS method. Table 2 reveals that the variable neighborhood search heuristic found partitions with $Z(P)$ values outside this confidence interval (i.e., $Z(P)$ = 6191 or $Z(P)$ = 6192) for all values of K on the interval 7 ≤ K ≤ 14. For the same interval, the relocation heuristic produced objective criterion function values ranging from 6208 to 6215 within its 150-second time limit.

The results for the n = 8000 and n = 10,000 Slashdot zoo tests problems are reported in Table 3. From the results for the n = 8000 test problem reported by Levorato et al. (2017, Table 3, p. 485 and Table 4, p. 488), we observe an average $Z(P)$ value of 16,060.55, a 95% confidence interval of [16,058.89 to 16,062.21], and K = 80 clusters for a two-hour implementation of their SeqILS method. Table 3 reveals that the variable neighborhood search heuristic found partitions with $Z(P)$ values outside this confidence interval (i.e., 16,046 ≤ $Z(P)$ ≤ 16,054) for all values of K on the interval 7 ≤ K ≤ 14. Tabu search also performed extremely well, yielding values outside the confidence interval (i.e., 16,050 ≤ $Z(P)$ ≤ 16,054) for 8 ≤ K ≤ 14. For the interval 7 ≤ K ≤ 14. the relocation heuristic produced objective criterion function values ranging from 16,086 to 16,093 within its 300-second time limit.

From the results for the n = 10,000 test problem reported by Levorato et al. (2017, Table 3, p. 485 and Table 4, p. 488), we observe an average $Z(P)$ value of 20,571.35, a 95% confidence interval of [20,565.95 to 20,576.75], and K = 95 clusters for a two-hour implementation of their SeqILS method. Table 3 reveals that the variable neighborhood search heuristic found partitions with $Z(P)$ values well outside this confidence interval (i.e., 20,542 ≤ $Z(P)$ ≤ 20,559) for all values of K on the interval 6 ≤ K ≤ 14. The performance of the tabu search heuristic was also outstanding, yielding values outside the confidence interval (i.e., 20,542 ≤ $Z(P)$ ≤ 20,555) for all values of K on the interval 7 ≤ K ≤ 14. For the interval 6 ≤ K ≤ 14, the relocation heuristic produced objective criterion function values ranging from 20,579 to 20,598 within its 300-second time limit.

Table 2
Results for Slashdot subproblems of size $600 \le n \le 4000$.

		Best $Z(P)$ for RH after time in seconds					$Z(P)$	$Z(P)$
	K	10	30	60	120	150	TS	VNS
$n = 600$	2	112	112				112	110
$\|E\| =$ 1917	3	109	109				109	109
60-s limit	4	109	109				109	109
	5	109	109				109	109
	6	111	109				109	109
$n = 1000$	2	628	627				627	627
$\|E\| =$ 5991	3	603	603				603	603
60-s limit	4	601	601				601	601
	5	602	600				600	600
	6	602	601				600	600
	7	602	602				602	600
	8	602	602				600	600
$n = 2000$	2	2302	2300	2299	2299	2298	2298	2298
$\|E\| =$ 20815	3	2216	2216	2214	2214	2214	2212	2210
300-s limit	4	2198	2192	2192	2192	2191	2190	2188
	5	2190	2190	2190	2190	2190	2184	2184
	6	2189	2189	2189	2188	2187	2184	2184
	7	2194	2189	2189	2187	2187	2185	2184
	8	2193	2193	2191	2191	2190	2186	2184
	9	2196	2194	2191	2191	2191	2186	2184
	10	2192	2192	2192	2191	2189	2189	2184
$n = 4000$	2	6546	6546	6543	6543	6543	6542	6542
$\|E\| =$ 49532	3	6279	6279	6278	6277	6274	6268	6268
300-s limit	4	6235	6235	6235	6230	6227	6217	6211
	5	6218	6216	6216	6216	6212	6203	6196
	6	6220	6220	6214	6214	6213	6194	6193
	7	6217	6217	6213	6213	6213	6202	6191
	8	6218	6211	6211	6210	6210	6196	6191
	9	6226	6220	6209	6208	6208	6193	6192
	10	6218	6218	6211	6211	6211	6194	6191
	11	6227	6215	6215	6209	6209	6192	6191
	12	6220	6214	6214	6213	6213	6196	6192
	13	6218	6218	6217	6217	6214	6192	6192
	14	6218	6218	6218	6218	6215	6194	6191

Note: Multiple restarts of RH were completed for the first half of the allotted time limit and the best-found $Z(P)$ values were stored at 10, 30, and (where applicable) 60, 120, and 150 s. The second half of the allotted time limit was used to apply TS or VNS using the best-found RH solution as the starting solution. The TS and VNS columns report the best-found $Z(P)$ values from the two-stage (RH + TS) and (RH + VNS) processes, respectively.

3.4. Results for the Wiki election and full Slashdot zoo data

The results for the Wiki election and (full) Slashdot zoo networks are reported in the top and bottom panels of Table 4, respectively. Unlike the results for the slices from the Slashdot zoo dataset discussed in subsection 3.3, we do not have benchmark values of $Z(P)$ for the Wiki election and full Slashdot zoo networks from previous research to report for comparative purposes. Although Levorato et al. (2017) report $Z(P)$ as a percentage of the total number of edges for these networks, the divisor for computing this percentage is not clear. For example, the undirected version of the Wiki election network that is available from http://www.ic.uff.br/~yuri/files/CCinst.zip has some edges with zero weights, which we excluded from the data and reduced the edge count accordingly. In addition, there were a large number of edges at the end of the file with '*' denoted as the edge weight. Apparently, these were edges where cases where $\{u, v\} \in E^+$ and $\{v, u\} \in E^-$. We excluded these edges as well and reduced the edge count. The full Slashdot zoo network also had a large number of edges with '*' denoted as the edge weight and these were discarded as well.

For the Wiki elections network, the total number of restarts realized by the relocation heuristic within the 300-second time limit ranged from 847 at $K = 14$ to 11,356 at $K = 2$. The $Z(P)$ values obtained by the relocation heuristic for the Wiki elections data fell within a narrow band of $14{,}165 \le Z(P) \le 14{,}173$ for all numbers of clusters on the interval $5 \le K \le 14$. The variable neighborhood search algorithm improved the $Z(P)$ values for all values of K. Moreover, the $Z(P)$ values obtained by the variable neighborhood search algorithm also fell within a narrow band of $14{,}142 \le Z(P) \le 14{,}148$ for all numbers of clusters on the interval $5 \le K \le 14$. The best objective criterion value of 14,142 was obtained at $K = 6$ clusters. The criterion function values for the tabu search heuristic revealed a systematic improvement over those obtained by the relocation heuristic; however, they were systematically inferior to the variable neighborhood search results. The tabu search criterion values ranged from $14{,}154 \le Z(P) \le 14{,}162$ for all numbers of clusters on the interval $5 \le K \le 14$.

For the full Slashdot zoo network, the total number of restarts realized by the relocation heuristic within the 300-second time limit ranged from 229 at $K = 14$ to 2628 at $K = 2$. The $Z(P)$ values obtained by the relocation heuristic for the Slashdot zoo data fell within a somewhat wider band of $68{,}649 \le Z(P) \le 68{,}689$ for all numbers of clusters on the interval $9 \le K \le 14$. The variable neighborhood search algorithm substantially improved the $Z(P)$ values for all values of K. The $Z(P)$ values obtained by the variable neighborhood search algorithm also fell within a band of $67{,}782 \le Z(P) \le 67{,}820$ for all numbers of clusters on the interval $9 \le K \le 14$. The three best objective criterion values identified were 67,782 ($K = 12$), 67,784 ($K = 9$), and 67,785 ($K = 11$). The tabu search heuristic did not perform well for the full Slashdot zoo network, yielding only modest improvement over the relocation heuristic but criterion function values markedly inferior to those associated with variable neighborhood search. Possible explanations for the results are: (i) the need for a different τ parameter for networks this large, or (ii) the need for the search operations used by

Table 3
Results for Slashdot subproblems of size n = 8000 and n = 10,000.

		Best $Z(P)$ for RH after time in seconds					$Z(P)$	$Z(P)$
	K	60	120	180	240	300	TS	VNS
n = 8000	2	16962	16959	16959	16959	16959	16958	16956
$\lvert E\rvert$ = 109705	3	16262	16262	16262	16262	16262	16252	16249
600-s limit	4	16151	16150	16150	16149	16149	16113	16116
	5	16115	16115	16115	16115	16115	16076	16073
	6	16103	16102	16102	16102	16102	16064	16062
	7	16092	16092	16092	16092	16092	16065	16054
	8	16088	16088	16087	16087	16087	16053	16049
	9	16110	16099	16098	16089	16089	16050	16050
	10	16108	16094	16094	16090	16090	16054	16046
	11	16101	16100	16086	16086	16086	16053	16053
	12	16095	16095	16093	16093	16093	16053	16051
	13	16105	16096	16091	16091	16091	16053	16052
	14	16112	16108	16108	16108	16087	16054	16051
n = 10,000	2	21797	21796	21792	21792	21792	21788	21787
$\lvert E\rvert$ = 139071	3	20830	20830	20830	20830	20830	20814	20815
600-s limit	4	20672	20672	20671	20660	20660	20628	20627
	5	20623	20623	20612	20612	20612	20581	20574
	6	20610	20600	20598	20598	20598	20567	20559
	7	20591	20591	20591	20591	20590	20545	20551
	8	20596	20596	20595	20587	20587	20542	20546
	9	20591	20591	20591	20584	20584	20555	20547
	10	20594	20594	20594	20594	20591	20547	20542
	11	20588	20583	20583	20583	20583	20545	20546
	12	20595	20592	20592	20586	20583	20544	20549
	13	20604	20579	20579	20579	20579	20544	20546
	14	20598	20588	20586	20586	20586	20550	20545

Note: Multiple restarts of RH were completed for the first half (300-s) of the allotted 600-s time limit and the best-found $Z(P)$ values were stored at 60, 120, 180, 240, and 300 s. The second half of the allotted time limit was used to apply TS or VNS using the best-found RH solution as the starting solution. The TS and VNS columns report the best-found $Z(P)$ values from the two-stage (RH + TS) and (RH + VNS) processes, respectively.

Table 4
Results for undirected Wiki elections (n = 8298) and Slashdot (n = 82,144) networks.

		Best $Z(P)$ for RH after time in seconds					$Z(P)$	$Z(P)$	*RH*
	K	60	120	180	240	300	TS	VNS	Restarts
Wiki elections	2	14578	14574	14574	14574	14574	14559	14556	11356
n = 8298	3	14223	14223	14220	14219	14219	14195	14190	6299
$\lvert E\rvert$ = 106548	4	14181	14180	14178	14178	14178	14167	14154	4238
600-s limit	5	14173	14173	14173	14173	14173	14155	14145	3173
	6	14177	14171	14170	14169	14169	14159	14142	2507
	7	14174	14174	14170	14170	14170	14156	14145	2027
	8	14175	14171	14171	14171	14171	14162	14146	1707
	9	14170	14170	14170	14170	14165	14154	14146	1466
	10	14178	14173	14173	14172	14172	14157	14147	1287
	11	14176	14169	14169	14169	14169	14155	14144	1129
	12	14175	14174	14170	14170	14170	14158	14145	1013
	13	14177	14173	14173	14171	14171	14161	14143	929
	14	14170	14170	14170	14170	14170	14159	14148	847
Slashdot zoo	2	74579	74579	74579	74560	74560	74352	73885	2648
n = 82,144	3	69809	69798	69772	69697	69672	69520	68934	1594
$\lvert E\rvert$ = 498532	4	69165	69137	69071	69067	69048	68939	68230	1166
600-s limit	5	68952	68897	68874	68874	68801	68694	67915	903
	6	68747	68737	68737	68737	68737	68673	67870	732
	7	68747	68747	68747	68726	68700	68612	67829	600
	8	68767	68716	68716	68586	68586	68519	67829	509
	9	68754	68726	68726	68649	68649	68605	67784	431
	10	68811	68687	68677	68663	68663	68607	67815	366
	11	68776	68721	68692	68649	68649	68576	67785	315
	12	68690	68690	68690	68675	68675	68621	67782	281
	13	68754	68723	68723	68689	68689	68612	67820	246
	14	68773	68683	68683	68661	68661	68627	67807	229

Note: Multiple restarts of RH were completed for the first half (300-s) of the allotted 600-s time limit and the best-found $Z(P)$ values were stored at 60, 120, 180, 240, and 300 s. The total number of RH restarts realized within the allotted time is shown in the far-right column. The second half of the allotted time limit was used to apply TS or VNS using the best-found RH solution as the starting solution. The TS and VNS columns report the best-found $Z(P)$ values from the two-stage (RH + TS) and (RH + VNS) processes, respectively.

variable neighborhood search that allow for large jumps to escape local optima.

4. Conclusions

4.1. Summary

Levorato et al. (2017) recently published results indicating that the Pajek implementation of the relocation heuristic for generalized structural balance developed by Doreian and Mrvar (1996) is not practical for networks with $n > 1000$ vertices because of the enormous computation time requirements involved. We hypothesize that these findings are likely attributable to two factors. First, the relocation heuristic uses both single-vertex reassignments (transfers) from one cluster to another and pairwise interchanges (exchanges) of vertices in different clusters. Evidence from other partitioning problems (Brusco and Steinley, 2007, 2011) has shown that pairwise interchanges produce little if any refinement of the solutions obtained the via vertex reassignment process and are also computationally demanding for problems where n is large. Second, it is possible that the heuristic code in Pajek is not customized to exploit the sparse nature of many large networks and this can also significantly increase the computational time.

We implemented a modified version of the relocation heuristic developed by Doreian and Mrvar (1996) that excludes pairwise interchanges. The heuristic was applied to the Slashdot subnetworks considered by Levorato et al. (2017), which ranged in size from 600 to 10,000 vertices. Whereas Levorato et al. (2017) noted that the Pajek implementation of the relocation heuristic could not achieve 100 restarts within two hours of CPU time for test problems where $n > 1000$, our implementation was typically able to realize thousands of restarts within five minutes for test problems with 2000, 4000, 8000, and even 10,000 vertices. Moreover, our implementation of the relocation heuristic was able to realize hundreds of restarts within five minutes for the full Slashdot network, which has 82,144 vertices.

The general quality of the partitions obtained by the multistart relocation heuristic was good, but not great. The relocation heuristic was able to match the best-known objective criterion values for the $n = 600$ and $n = 1000$ Slashdot networks, but was unable to do so for the larger networks. Nevertheless, the relocation heuristic commonly produced partitions with objective criterion values that were not too distant from the best-known values, which made it easier for the variable neighborhood search heuristic to refine them efficiently.

The tabu search and variable neighborhood search heuristics embed the relocation heuristic within the algorithmic process. Taking a partition obtained by the multistart relocation heuristic as initial incumbent solution, the tabu search and variable neighborhood search algorithms generally provide a refinement of the solution that results in significant improvement of the criterion function values. The variable neighborhood search heuristic found objective criterion values that were competitive with the results obtained by Levorato et al. (2017) and revealed that these criterion values could be obtained with a modest number of clusters.

4.2. Limitations and extensions

The limitations of the methods reported fall into several categories. First, as noted previously, the methods considered are heuristic in nature. There is no guarantee that a global optimum has been obtained. Nevertheless, Levorato et al. (2017) have noted that heuristics are generally required for problem instances of correlation clustering and generalized structural balance when n exceeds 50 and, therefore, the development of better heuristics remains a worthwhile endeavor. This is an obvious conclusion.

A second limitation that could be raised is that, because we focus on generalized structural balance partitioning and not correlation clustering, our algorithms must be run for different values of K. By contrast, correlation clustering procedures select K as part of the solution process. Nevertheless, our results suggest that there is some merit for using generalized K-balance partitioning for the purpose of correlation clustering, as our results showed that the best objective criterion value was often achievable using a very small number of clusters.

A third limitation is that our comparative analyses cannot definitively demonstrate that one metaheuristic is better than another. For example, Levorato et al. (2017) reported results showing *their implementation* of variable neighborhood search was outperformed by *their implementation* of iterated local search. However, *our implementation* of variable neighborhood search seemed to outperform both of these Levorato et al. (2017) implementations in many instances. Along similar lines, it would be a mistake to conclude based on our findings that variable neighborhood search is better than tabu search for partitioning signed networks. Both methods have unique strengths. Tabu search is excellent for providing a robust search of the near-neighborhood of a solution, whereas variable neighborhood search facilitates jumps to distant neighborhoods. It is for this reason that tabu search and variable search are sometimes integrated into a single metaheuristic for combinatorial optimization problems (Pérez et al., 2003; Brusco and Kohn, 2009).

Finally, a fourth limitation pertains to the computational feasibility of our methods. This limitation can actually be subdivided into: (i) issues pertaining to computer memory, and (ii) issues pertaining to computation time. Tackling large networks requires the dimensioning of large arrays. Although it is not necessary to generate $n \times n$ arrays for sparse networks, it is necessary to generate arrays that are $n \times d$, where d is the maximum degree across all edges. If the Slashdot zoo data had even a slightly larger d, we could not have compiled our Fortran program given the limitations of our hardware platform.

Our Fortran implementations of the multistart relocation heuristic and variable neighborhood search are generally quite efficient. Even for the 82,144-vertex Slashdot zoo network, we were able to get hundreds of restarts of the multistart relocation heuristic within a five-minute time limit. Moreover, variable neighborhood search efficiently refined these solutions in only five additional minutes. Nevertheless, it must be acknowledged that networks with greater density could require appreciably more computation time. In light of this limitation, one potentially interesting extension might be the use of variable neighborhood search after replacing transfers with more efficient search algorithms such as Louvain-type methods (Blondel et al., 2008; Traag, 2015). In any case, computational limitations are a given but we emphasize that having well designed algorithms for partitioning signed networks is not just an end in itself, while being very important, they are integral to testing, in a serious fashion, hypotheses about the operation of social processes.

References

Anchuri, P., Magdon-Ismail, J., 2012. Communities and balance in signed networks: A spectral approach. In: Proceedings of the 2010 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONSM '12). IEEE. pp. 235–242.

Aref, S., Mason, A.J., Wilson, M.C., 2017. An Exact Method for Computing the Frustration index in Signed Networks Using Binary Programming. asXiv preprint arXiv:1611.09030 (2017). https://arxiv.org/pdf/1611.09030.

Bansal, N., Blum, A., Chawla, S., 2004. Correlation clustering. Mach. Learn. 56 (1–3), 89–113.

Batagelj, V., Doreian, P., Ferligoj, A., Kejzar, N., 2014. Understanding Large Temporal Networks and Spatial Networks. Wiley, Chichester.

Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E., 2008. Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp iopscience.iop.org/article/10.1088/1742-5468/2008/10/P10008/meta.

Borgatti, S.P., Everett, M.G., 1997. Network analysis of 2-mode data. Soc. Netw. 19, 243–269.

Brusco, M.J., Kohn, H.-F., 2009. Clustering qualitative data based on binary equivalence relations: a neighborhood search heuristic for the clique partitioning problem. Psychometrika 74, 685–703.

Brusco, M.J., Steinley, D., 2007. A comparison of heuristic procedures for minimum within-cluster sums of squares partitioning. Psychometrika 72, 583–600.

Brusco, M.J., Steinley, D., 2010. K-balance partitioning: an exact method with application

to generalized structural balance and other psychological contexts. Psychol. Methods 15, 145–157.
Brusco, M., Steinley, D., 2011. A tabu search heuristic for deterministic two-mode blockmodeling of binary network matrices. Psychometrika 76, 612–633.
Brusco, M., Doreian, P., Mrvar, A., Steinley, D., 2011. Linking theory, models, and data to understand social network phenomena: two algorithms for relaxed structural balance partitioning. Sociol. Methods Res. 40, 57–87.
Brusco, M., Doreian, P., Lloyd, P., Steinley, D., 2013. A variable neighborhood search method for a two-mode blockmodeling problem in social network analysis. Netw. Sci. 1, 191–212.
Cartwright, D., Harary, F., 1956. Structural balance: a generalization of Heider's theory. Psychol. Rev. 63, 277–293.
Chen, Y., Wang, X.-L., Yuan, B., 2013. Overlapping community detection in signed networks. arXiv preprint: https://arxiv.org/ftp/arxiv/papers/1310/1310.4023.pdf.
Davis, J.A., 1967. Clustering and structural balance in graphs. Hum. Relat. 20, 181–187.
de Nooy, W., Mrvar, A., Batagelj, V., 2011. Exploratory Social Network Analysis with Pajek, 2nd edition. Cambridge University Press, New York.
De Amorim, S.G., Barthélemy, J.-P., Ribeiro, C.C., 1992. Clustering and clique partitioning: simulated annealing and tabu search approaches. J. Class. 9, 17–41.
Doreian, P., 2008. A multiple indicator approach to blockmodeling signed networks. Soc. Netw. 30, 247–258.
Doreian, P., Mrvar, A., 1996. A partitioning approach to structural balance. Soc. Netw. 18, 149–168.
Doreian, P., Mrvar, A., 2009. Partitioning signed social networks. Soc. Netwo. 31, 1–11.
Doreian, P., Mrvar, A., 2014. Testing two theories for generating signed networks using real data. Metodoloski Zvezki: Adv. Methodol. Stat. 11, 31–63.
Doreian, P., Mrvar, A., 2015. Structural balance and signed international relations. J. Soc. Struct. 16, 1–49.
Doreian, P., Batagelj, V., Ferligoj, A., 2005. Generalized Blockmodeling. Cambridge University Press, Cambridge, UK.
Doreian, P., Lloyd, P., Mrvar, A., 2013. Partitioning large signed two-mode networks: problems and prospects. Soc. Netw. 35, 1–21.
Došlić, T., Vukičević, D., 2007. Computing the bipartite edge frustration of fullerene graphs. Discrete Appl. Math. 155, 1294–1301.
Drummond, L., Figueiredo, R., Frota, Y., Levorato, M., 2013. Efficient solution of the correlation clustering problem: an application to structural balance. YanTang, D., Herv, P. (Eds.), OTM 2013 Workshops, Lecture Notes in Computer Science, vol. 8186, 674–683.
Elsner, M., Schudy, W., 2009. Bounding and comparing methods for correlation clustering beyond ILP. ILP -09, Proceedings of the Workshop on integer Linear Programming for Natural Language Processing. pp. 19–27.
Feo, T.A., Resende, M.G.C., 1995. Greedy randomized adaptive search procedures. J. Glob. Optim. 6, 109–133.
Facchetti, G., Iacono, G., Altafini, C., 2011. Computing global structural balance in large-scale signed networks. Proc. Natl. Acad. Sci. U. S. A. 108, 20953–20958.
Figueiredo, R., Moura, G., 2013. Mixed integer programming formulations for clustering problems related to structural balance. Soc. Netw. 35, 639–651.
Giotis, I., Guruswami, V., 2006. Correlation clustering with a fixed number of clusters. Proceedings of the Seventeenth Annual ASM-SIAM Symposium on Discrete Algorithms. pp. 1167–1176.
Glover, F., 1989. Tabu search – Part I. ORSA J. Comp. 1, 190–206.
Glover, F., 1990. Tabu search – Part II. ORSA J. Comp. 2, 4–32.
Glover, F., Laguna, M., 1993. Tabu search. In: Reeves, C. (Ed.), Modern Heuristic Techniques for Combinatorial Problems. Blackwell, Oxford, pp. 70–141.
Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York.
Hansen, P., Mladenovic, N., 1997. Variable neighborhood search for the p-median. Locat. Sci. 5, 207–226.
Hansen, P., Mladenovic, N., 2001. J-means: a new local search heuristic for minimum sum of squares clustering. Pattern Recognit. 34, 405–413.
Heider, F., 1946. Attitudes and cognitive organization. J. Psychol. 21, 107–112.
Huffner, F., Betzler, N., Niedermeier, R., 2010. Separator-based data reduction for signed graph balancing. J. Comb. Opt. 20, 335–360.
Iacono, G., Ramezani, F., Soranzo, N., Altafini, C., 2010. Determining the distance to monotonicity of a biological network: a graph-theoretical approach. IET Syst. Biol. 4 (3), 223–235.
Jacobs, L.W., Brusco, M.J., 1995. Note: a local-search heuristic for large set-covering problems. Nav. Res. Logist. 42 (7), 1129–1140.
Kasteleyn, P.W., 1963. Dimer statistics and phase transitions. J. Mathematical Phys. 4 (2), 287–293.
Kim, S., Yoo, C.D., Nowoin, S., Kohli, P., 2014. Image segmentation using higher-order correlation clustering. IEEE Trans. Pattern Anal. Mach. Intell. 36, 1761–1774.
Kunegis, J., Schmidt, S., Lommatzsch, A., Lerner, J., De Luca, E.W., Albayrak, S., 2010. Spectral analysis of signed graphs for clustering, prediction, and visualization. In: Proceedings of the 2010 SIAM International Conference on Data Mining. Vol. 10, SIAM. pp. 559–570.
Lemann, T.B., Solomon, R.L., 1952. Group characteristics as revealed in sociometric patterns and personality ratings. Sociometry 15, 7–90.
Leskovec, J., Huttenlocher, D., Kleinberg, J., 2010. Signed Networks in Social Media. 28th ACM Conference on Human Factors in Computing Systems (CHI) 2010.
Levorato, M., Figueiredo, R., Frota, Y., Drummond, L., 2017. Evaluating balancing on social networks through the efficient solution of correlation clustering problems. EURO J. Comput. Opt. 5, 467–498.
Lourenco, H.R., Martin, O.C., Stutzle, T., 2003. Iterated local search. In: In: Glover, F., Kochenberger, G.A. (Eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science Vol. 57. Springer, New York, pp. 320–352.
Lourenco, H.R., Martin, O.C., Stutzle, T., 2010. Iterated local search: framework and applications. In: 2nd edition. In: Gendreau, M., Potvin, J.-Y. (Eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science Vol. 146. Springer, New York, pp. 363–397.
Ma, L., Gong, M., Du, H., Shen, B., Jiao, L., 2015. A memetic algorithm for computing and transforming structural balance in signed networks. Knowl.-Based Systems 85, 169–209.
Mladenovic, N., Hansen, P., 1997. Variable neighborhood search. Comput. Oper. Res. 24, 1097–1100.
Newcomb, T.N., 1961. The Acquaintance Process. Holt Rinehart and Winston, New York.
Pacheco, J., Valencia, O., 2003. Design of hybrids for the minimum sum-of-squares clustering problem. Comput. Stat. Data Anal. 43, 235–248.
Pérez, J.A.M., Moreno-Vega, J.M., Martin, I.R., 2003. Variable neighborhood tabu search ad its application to the median cycle problem. Eur. J. Oper. Res. 151, 361–378.
Ruiz, R., Stutzle, T., 2007. A simple and effective greedy algorithm for the permutation flowshop scheduling problem. Eur. J. Oper. Res. 177 (3), 2033–2049.
Ruiz, R., Stutzle, T., 2008. An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives. Eur. J. Oper. Res. 187 (3), 1143–1159.
Sampson, S.F., 1968. A Novitiate in a Period of Change: An Experimental Case Study of Relationships. Unpublished Ph.D. dissertation. Department of Sociology, Cornell University, Ithaca, NY.
Tang, J., Chang, Y., Aggarwal, C., Liu, H., 2016. A survey of signed network mining in social media. ACM Comp. Surv. 49 (3) Article 42.
Traag, V.A., 2015. Faster unfolding of communalities: speeding up the Louvain algorithm. Phys. Rev. E 92, 032801.
Traag, V.A., Bruggeman, J., 2009. Community detection in networks with positive and negative links. Phys. Rev. E 80 (3), 036115.
Traag, V.A., Doreian, P., Mrvar, A., 2018. Partitioning signed networks. arXiv:1803.02082.
Yang, B., Cheung, W., Liu, J., 2007. Community mining from signed social networks. IEEE Trans. Knowl. Data Eng. 19, 1333–1348.
Zhang, Z., Cheng, H., Chen, W., Zhang, S., Fang, Q., 2008. Correlation clustering based on genetic algorithm for documents clustering. IEEE Congress on Evolutionary Computation. pp. 3191–3198.
Zhang, Z., Cheng, H., Chen, W., Zhang, S., Fang, Q., 2010. Correlation clustering based on genetic algorithm for documents clustering. IEEE Congr. Evol. Comp. 3191–3198.