PSYCHOMETRIKA—VOL. 76, NO. 4, 612–633
OCTOBER 2011
DOI: 10.1007/S11336-011-9221-9

A TABU-SEARCH HEURISTIC FOR DETERMINISTIC TWO-MODE BLOCKMODELING OF BINARY NETWORK MATRICES

MICHAEL BRUSCO

FLORIDA STATE UNIVERSITY

DOUGLAS STEINLEY

UNIVERSITY OF MISSOURI-COLUMBIA

Two-mode binary data matrices arise in a variety of social network contexts, such as the attendance or non-attendance of individuals at events, the participation or lack of participation of groups in projects, and the votes of judges on cases. A popular method for analyzing such data is two-mode blockmodeling based on structural equivalence, where the goal is to identify partitions for the row and column objects such that the clusters of the row and column objects form blocks that are either complete (all 1s) or null (all 0s) to the greatest extent possible. Multiple restarts of an object relocation heuristic that seeks to minimize the number of inconsistencies (i.e., 1s in null blocks and 0s in complete blocks) with ideal block structure is the predominant approach for tackling this problem. As an alternative, we propose a fast and effective implementation of tabu search. Computational comparisons across a set of 48 large network matrices revealed that the new tabu-search heuristic always provided objective function values that were better than those of the relocation heuristic when the two methods were constrained to the same amount of computation time.

Key words: clustering, two-mode networks, blockmodeling, tabu search, heuristics.

1. Introduction

Most clustering applications focus on problems where the goal is to cluster a single set of objects. For example, given an $N \times M$ matrix of metric variable measurements for N objects on M variables, a common approach is to cluster the objects based on their distances with respect to the variables. Methods for accomplishing this task include (but are not limited to) Ward's (1963) hierarchical clustering method, K-means clustering (Steinhaus, 1956) and p-median clustering (Mulvey & Crowder, 1979). We refer to the clustering problem associated with each of these methods as *one-mode* because only the N objects are clustered. However, in some applications, it is important to simultaneously establish clusters for both the N objects associated with the rows and the M variables associated with the columns, which results in a *two-mode* clustering problem that is also commonly referred to as *biclustering* in the biological sciences (Madeira & Oliveira, 2004; Prelić, Blueler, Zimmerman, Wille, Bühlmann, Gruissem, Hennig, Thiele, & Zitzler, 2006; van Uitert, Meuleman, & Wessels, 2008). Since the pioneering work of Hartigan (1972) in this area, there have been a number of deterministic (Krolak-Schwerdt, 2003; van Rosmalen, Groenen, Trejos, & Castillo, 2009) and stochastic (Govaert & Nadif, 2003; Kaiser & Leisch, 2008) approaches designed for two-mode clustering.

Two-mode clustering problems are especially relevant within the context of social network data (Borgatti & Everett, 1997; Davis, Gardner, & Gardner, 1941; Doreian, Batagelj, & Ferligoj, 2004, 2005, Chapter 8; Wasserman & Faust, 1994). In these applications, there are two sets of objects: (1) the N objects corresponding to the rows, and (2) the M objects corresponding to the

Requests for reprints should be sent to Michael Brusco, Department of Marketing, College of Business, Florida State University, Tallahassee, FL 32306-1110, USA. E-mail: mbrusco@fsu.edu

columns. The row and column objects define the two distinct sets of a bipartite graph with the elements of the $N \times M$ binary network matrix, $\mathbf{X} = [x_{ij}]$, assuming a value of $x_{ij} = 1$ if there is a tie or bond between row object i and column object j and $x_{ij} = 0$ otherwise. As an example from the social network literature, Mische and Pattison (2000) considered a network matrix where there are $N = 29$ organizations that either participate ($x_{ij} = 1$) or do not participate ($x_{ij} = 0$) in each of $M = 22$ community projects. Davis et al. (1941) described a social network consisting of $N = 18$ women in a southern community and the $M = 14$ social events they attended ($x_{ij} = 1$) or did not attend ($x_{ij} = 0$), which has been widely studied in the literature (Borgatti & Everett, 1997; Brusco & Steinley, 2006, 2007a; Doreian et al. 2004, 2005; Freeman, 2003; Homans, 1950). Another popular social network matrix examines the voting behavior of the $N = 9$ Supreme Court justices on a set of $M = 26$ cases during the year 2000 (Brusco & Steinley, 2007a; Doreian et al. 2004, 2005), where $x_{ij} = 1$ and $x_{ij} = 0$ reflect votes with the majority and minority, respectively. Much larger two-node networks occur in other applications, such as document summarization and clustering (Xu, Liu, & Gong, 2003), where N words and M documents comprise the rows and columns of a network matrix, respectively.

The ubiquity of two-mode binary network data has sparked a number of research efforts dedicated to methodological approaches for modeling these data. Some approaches focus on establishing permutations of the row and column objects to uncover structure (Arabie, Hubert, & Schleutermann, 1990; Brusco & Steinley, 2006). Other methods attempt to provide Boolean decompositions of the network matrix (Pattison & Breiger, 2002). Of particular importance in recent years are two-mode *blockmodeling* methods that generate partitions of the row and column objects to uncover structure in the two-mode network data (Brusco & Steinley, 2007a, 2009; Doreian et al. 2004, 2005). The row objects in each cluster of the partition of row objects form a block with the column objects in each cluster of the partition of column objects and, together, these blocks define the structure present in the network data. In this paper, we limit our focus to blockmodeling of two-mode network data.

The principles of structural equivalence (Breiger, Boorman, & Arabie, 1975; Lorrain & White, 1971) provide a strong foundation for two-mode blockmodeling. A blockmodel that is in perfect concordance with structural equivalence will contain blocks that are either *complete* (all 1s) or *null* (all 0s). The clusters for the row and column objects yield a $K \times L$ *image matrix*, $\mathbf{Q} = [q_{kl}]$, with values of $q_{kl} = 1 (q_{kl} = 0)$ if the block defined by row cluster k and column cluster l is complete (null), for $1 \le k \le K$ and $1 \le l \le L$. In applications of *confirmatory* (or *deductive*) *blockmodeling* (see Doreian et al., 2005, Chapter 8), the image matrix is assumed to be known in advance. More commonly, however, analysts are interested in *exploratory* (or *inductive*) *blockmodeling*, where the image matrix must be obtained as part of the solution process. We focus on this more challenging problem of inductive two-mode blockmodeling.

As an example, consider a situation where the row objects are persons and the column objects are events. The matching of a cluster r of rows (persons) and a cluster c of columns (events) forms a block. If all persons in cluster r attended every event in cluster c, then the block would be complete. Contrastingly, if no person in cluster r attended any of the events in cluster c, the block would be null. Values of 0 in blocks that have mostly 1s, as well as values of 1 in blocks that have mostly zeros, are *inconsistencies* with ideal block structure based on structural equivalence. In practice, there is typically no clustering of row and column objects that yields perfect structural equivalence; however, a popular optimization problem is to seek to identify partitions that minimize the total number of inconsistencies. Although a globally optimal solution obviously exists in light of the finite solution space, this is an inherently challenging discrete optimization problem because it requires partitions of two distinct object sets.

Doreian et al. (2004, 2005) developed a relocation heuristic for blockmodeling based on structural equivalence. The algorithm refines initial partitions of the row and column objects using two operations: (1) transfers of objects from their current cluster to one of the other clusters, and (2) exchanges of cluster memberships for pairs of objects. Accordingly, the relocation algorithm converges to a blockmodel that is locally optimal in the sense that there is no transfer or exchange that will reduce the number of inconsistencies with ideal block structure. To avoid the potential for a poor local optimum, the algorithm is restarted for a large number of random initial partitions, which we refer to as a *multistart* implementation of the heuristic.

A multistart implementation of the relocation heuristic has several desirable properties. One of its advantages is that it is relatively efficient and easy to implement. A second advantage, which is particularly beneficial, is that the heuristic need not require an analyst to pre-specify the image matrix and, accordingly, it is well suited for exploratory blockmodeling. The image matrix evolves during the execution of the algorithm because blocks can change from complete to null (or null to complete) as the number of 0s exceeds the number of 1s (or the number of 1s exceeds the number of 0s). A third desirable characteristic of the relocation algorithm is that it is very flexible and can be adapted for objective functions related to other forms of equivalence in one- and two-mode blockmodeling (Doreian et al., 2005), as well as blockmodeling of signed networks (Brusco, Doreian, Mrvar, & Steinley, 2011; Brusco & Steinley, 2010; Doreian & Mrvar, 1996; Mrvar & Doreian, 2009).

Multistart implementations of the relocation heuristic are apt to yield excellent performance for small- to modestly sized networks of up to about 50 row and column objects, regardless of the properties of the network matrix. For networks with 100 or more row and column objects, the likelihood of finding a globally optimal blockmodel using a multistart relocation heuristic diminishes rapidly, particularly for network matrices that do not have a strong ideal block structure. Moreover, a "catch-22" (Heller, 1961) situation arises in the sense that larger problems need more restarts to find good local optima, yet the computation time for each restart becomes increasingly larger as problem size increases. Thus, for large problems where many restarts are required, the computational cost of using a large number of restarts is often prohibitive. The inherent limitations of multistart relocation algorithms for blockmodeling are comparable to those encountered in other clustering contexts. For example, metaheuristics such as variable neighborhood search (Mladenović & Hansen, 1997) and genetic algorithms (Goldberg, 1989) have been shown to outperform multistart implementations of the K-means algorithm (Steinhaus, 1956) for within-cluster sum-of-squares partitioning (see Brusco & Steinley, 2007b). Similarly, simulated annealing (Aarts & Korst, 1989) and tabu search (Glover & Laguna, 1993) have proven superior to multiple restarts of relocation heuristics for the clique partitioning problem (see Brusco & Köhn, 2009; De Amorim, Barthélemy, & Ribeiro, 1992).

Our goal in this paper is to offer a new heuristic procedure for two-mode blockmodeling based on tabu search. The tabu-search heuristic uses the same search processes (object transfers and exchanges) of the relocation heuristic developed by Doreian et al. (2004), yet at the same time allows for a more thorough exploration of the neighborhoods of locally optimal solutions. Part of the motivation for the selection of tabu search is its strong performance within the context of other partitioning problems such as K-means clustering (Pacheco & Valencia, 2003), p-median clustering (Rolland, Schilling, & Current, 1996), and clique partitioning (De Amorim et al., 1992). Moreover, it is well-known that tabu search can be integrated with other types of metaheuristics such as variable neighborhood search.

In Section 2 of this paper, we offer a brief coverage of classification schemes for blockmodeling, which is then used to position our contribution to the literature. Section 3 presents a formal mathematical statement of the two-mode blockmodeling problem and discusses extant methods for this problem, including the relocation heuristic. Section 4 provides the details of the proposed tabu-search algorithm. Section 5 offers results for several small empirical networks to demonstrate that the proposed tabu-search heuristic obtains partitions yielding the best-known objective function value for a greater percentage of restarts. A simulation study is provided in

Section 6, revealing that the tabu-search algorithm substantially outperforms the multistart relocation heuristic for large networks. A two-mode blockmodeling application for a psychometric journal citation matrix is provided in Section 7. A brief summary is provided in Section 8.

2. A Classification of Blockmodeling Approaches

Blockmodeling methods can be classified based on a number of different characteristics. In this section, we briefly review some possibilities for distinguishing among approaches. Our classifications are not exhaustive, nor are they mutually exclusive. Nevertheless, they provide a foundation for positioning our work within a broader literature base. The classification schemes that we consider in the following subsections are: (1) deterministic vs. stochastic blockmodeling, (2) one-mode vs. two-mode blockmodeling, (3) exploratory vs. confirmatory blockmodeling, and (4) unsigned network vs. signed network blockmodeling. This section concludes with a subsection that positions the current paper within this framework.

2.1. *Deterministic vs. Stochastic Blockmodeling*

One important classification scheme categorizes approaches based on the presence or absence of an underlying probabilistic model. For example, Goldenberg, Zheng, Fienberg, and Airoldi (2009) use the terms "model-based" and "algorithmic" to distinguish between blockmodeling methods that do and do not assume an underlying probabilistic model, respectively. Similarly, Bickel and Chen (2009) use the terms "stochastic" and "deterministic" to differentiate among methods in a similar manner. Deterministic models based on structural equivalence are grounded in the pioneering work of Lorrain and White (1971) and Breiger et al. (1975), whereas deterministic approaches for regular equivalence can be traced to White and Reitz (1983). An excellent summarization of deterministic blockmodeling is provided by Doreian et al. (2005), and the foundation for most the models and methods described in their book is the minimization of inconsistency with ideal block structure.

Stochastic blockmodeling is rooted in the work of Holland and Leinhardt (1976, 1977, 1981) and Holland, Laskey, and Leinhardt (1983), who formalized blockmodeling within the context of random graphs. Nowicki and Snijders (2001) expounded on these earlier efforts by laying the groundwork for model estimation in the context of stochastic blockmodeling. More recently, Handcock, Raftery, and Tantrum (2007) proposed a model-based approach to blockmodeling that is based on latent spaces. Airoldi, Blei, Fienberg, and Xing (2008) have developed mixed-membership stochastic blockmodels, which are especially appropriate for relational data where independence and exchangeability assumptions are not tenable. Bickel and Chen (2009) consider the "Newman-Girvan modularity function" (Newman & Girvan, 2004) and propose an alternative modularity based on likelihood that is demonstrated superior, both asymptotically and via simulation. A more in-depth coverage of many of these recent developments in stochastic blockmodeling is provided by Goldenberg et al. (2009, Section 3.8).

2.2. *One-Mode vs. Two-Mode Blockmodeling*

One-mode blockmodeling pertains to situations where there is a single set of objects and the goal is to partition the object set to uncover block structure. There are many possibilities for deterministic blockmodeling of one-mode data. One approach is to convert the raw network data to a dissimilarity matrix and apply traditional clustering procedures such as K-means or Ward's method. Spectral clustering (von Luxburg, 2007) offers another option, whereby an eigen-decomposition is performed on the network and eigenvectors are clustered using K-means or Ward's method. A more specific procedure for one-mode networks is Breiger et al.'s (1975)

CONCOR method. Batagelj, Ferligoj, and Doreian (1992) offered a "direct" deterministic approach to one-mode blockmodeling that uses objective functions associated with minimizing inconsistency with ideal block structure. Stochastic approaches for one-mode blockmodeling have their foundation in random graphs (Fienberg, Meyer, & Wasserman, 1985; Holland et al., 1983; Lauritzen, 2008; Nowicki & Snijders, 2001), and include more recent approaches such as the mixed-membership model of Airoldi et al. (2008) and the latent space approach of Handcock et al. (2007).

Two-mode blockmodeling requires the establishment of a partition for each of two distinct sets of objects (e.g., a set of clusters for women, and a set of clusters for the events attended). Although two-mode clustering dates back at least to Hartigan (1972), deterministic approaches to two-mode blockmodeling of binary network data have only recently received considerable attention. Doreian et al. (2004) generalized their earlier work in one-mode blockmodeling to the two-mode case, and Brusco and Steinley (2007a, 2009) expounded on their models and methods. Two-mode clustering approaches have also been popular for modeling gene expression data in the biological sciences, where they are known as "biclustering" methods. In a recent survey, van Uitert et al. (2008) indicated that there were very few two-mode clustering procedures that were well suited for binary data, most notably those of Koyutürk, Szpanowski, and Grama (2004) and Prelić et al. (2006).

2.3. *Exploratory vs. Confirmatory Blockmodeling*

Doreian et al. (2005, pp. 349–350) present another important distinction for blockmodeling methods that hinges on the issue of pre-specification. In the context of exploratory blockmodeling, an analyst identifies the permissible block types and posits the presence of clusters; however, the number of clusters and the locations of the block types are unknown. This is the more common (and more challenging) type of blockmodeling problem tackled by most deterministic and stochastic approaches. Nevertheless, Doreian et al. propose that analysts often have more information available when fitting blockmodels. That is, analysts might be able to hypothesize the presence of certain block types in some (or all) of the various regions of the blockmodel structure. Such information allows analysts to pre-specify the location of some (or all) block types prior to implementation of the blockmodeling method. The degree of pre-specification that can be afforded in any given context results in a gravitation from an exploratory analysis toward a confirmatory analysis.

2.4. *Unsigned Network vs. Signed Network Blockmodeling*

Many applications in the social network literature pertain to binary networks with values of '1' or '0' representing the presence or absence of a tie, respectively. These applications may be classified as unsigned networks because there are no negative elements in the network matrix. However, there are some important applications where the elements of the network matrix are trinary ('+1', '0', or '−1'). An example would be affect ties in a group of individuals, where each person could identify each of the other group members as either a friend (+1), an enemy (−1), or a neutral acquaintance (0). The resulting network is known as a signed network, and Doreian et al. (2005, p. 295) caution that "*. . . structural equivalence is singularly inappropriate for analyzing signed networks*" (italics emphasis in accordance with the source). Accordingly, deterministic methods for analyzing signed graphs tend to emphasize different types of blockmodeling problems that are more commonly known as structural balance partitioning (Brusco & Steinley, 2010; Doriean & Mrvar, 1996) and relaxed structural balance partitioning (Brusco et al., 2011; Doreian & Mrvar, 2009).

2.5. *Positioning Our Contribution in this Framework*

Throughout the remainder of this paper, we focus exclusively on deterministic, two-mode, exploratory blockmodeling of binary networks. Moreover, our emphasis is geared toward the objective criterion of minimizing inconsistency with ideal block structure, which is the foundation for much of the work in deterministic blockmodeling (Doreian et al., 2005). The goal is to ascertain whether the tabu-search method is a viable alternative to the more popular relocation heuristics used in deterministic blockmodeling. In accordance with this goal, experimental analyses are limited to a comparison of heuristic procedures all seeking to optimize the same objective function (minimizing inconsistencies).

For completeness and clarity, it is critical to note that the use of tabu search for blockmodeling of two-mode network data is not unprecedented. Borgatti and Everett (1997, pp. 265–267) used the FACTIONS program of Ucinet (Borgatti, Everett, & Freeman, 2002) to cluster the bipartite graph for the southern women social event data from Davis et al. (1941). Borgatti and Everett noted that this program used tabu search to find a K-cluster partition that maximizes the correlation between the clustered data and an ideal pattern corresponding to 100% density (all 1s) within clusters and 0% density (all 0s) between clusters. The implementation details of the tabu-search heuristic were not provided by Borgatti and Everett. More importantly, based on the results provided for the application to the southern women social event data, it is clear that the tabu-search heuristic was actually used to obtain a solution for a *one-mode partitioning problem* with a pre-specified ideal block structure (i.e., 100% density within groups and 0% density between groups). The two modes, women and events, were collapsed to form a single mode of 'objects', and then the tabu-search heuristic in FACTIONS produced a *single partition* of these objects into two clusters, which each contained both women and events. It is imperative to clarify that this implementation is very different from our tabu-search implementation, which establishes *two partitions* (one for each mode); and the number of clusters for the partition of one mode (e.g., the women) need not be the same as the number of clusters for the partition of the second mode (e.g., the events). Moreover, like the relocation heuristic of Doreian et al. (2004, 2005), the ideal structure need not be pre-specified in our implementation of tabu search.

3. A Deterministic Two-Mode Blockmodeling Problem

3.1. *Formulation of Two-Mode Blockmodeling (Problem P1)*

Two-mode blockmodeling is formulated as a discrete optimization problem. A formal description of this two-mode clustering problem uses the following notation:

$\mathbf{X}$:	an $N \times M$ binary data matrix with elements $x_{ij} = 1$ if there is a bond between row object i and column object j and $x_{ij} = 0$ otherwise, for $1 \le i \le N$ and $1 \le j \le M$;
K:	the number of clusters for the row objects of $\mathbf{X}$;
L:	the number of clusters for the column objects of $\mathbf{X}$;
Π_K:	the set of all K-cluster partitions of the row objects;
π_K:	$\pi_K = \{R_1, R_2, \dots, R_K\}$ is a K-cluster partition of the row objects ($\pi_K \in \Pi_K$) where R_k is the set of row objects assigned to cluster k and $N_k = \lvert R_k \rvert$ is the number of row objects assigned to cluster k, for $1 \le k \le K$;
Ω_L:	the set of all L-cluster partitions of the column objects;
ω_L:	$\omega_L = \{C_1, C_2, \dots, C_L\}$ is an L-cluster partition of the column objects ($\omega_L \in \Omega_L$) where C_l is the set of column objects assigned to cluster l and $M_l = \lvert C_l \rvert$ is the number of column objects assigned to cluster l, for $1 \le l \le L$.

Given the above definitions, the optimization problem, P1, for two-mode blockmodeling can be expressed as follows:

$$\underset{(\pi_K \in \Pi_K, \omega_L \in \Omega_L)}{\text{Minimize:}} \quad g(\pi_K, \omega_L) = \sum_{k=1}^{K} \sum_{l=1}^{L} \min\{\lambda_{kl}, \rho_{kl}\}, \tag{1}$$

where:

$$\lambda_{kl} = \sum_{i \in R_k} \sum_{j \in C_l} x_{ij}, \quad \forall 1 \le k \le K \text{ and } 1 \le l \le L, \tag{2}$$

and

$$\rho_{kl} = \sum_{i \in R_k} \sum_{j \in C_l} (1 - x_{ij}), \quad \forall 1 \le k \le K \text{ and } 1 \le l \le L. \tag{3}$$

The optimization problem posed by P1 is to find a K-cluster partition of the row objects, π_K, and an L-cluster partition of the column objects, ω_L, that minimizes the total number of inconsistencies with an ideal structure where all blocks are either complete or null. The value of λ_{kl} is the number of 1s in the block defined by the objects in row cluster k and column cluster l, where ρ_{kl} is the number of zeros in the block. If $\lambda_{kl} \ge \rho_{kl}$, then the block would be deemed complete and ρ_{kl} would represent the number of inconsistencies in the block that would be collected in the objective function. Similarly, if $\lambda_{kl} < \rho_{kl}$, then the block would be deemed null and λ_{kl} would represent the number of inconsistencies in the block that would be collected in the objective function. Summing over all $K \times L$ blocks in the objective function (1) represents the total number of inconsistencies with an ideal block structure.

3.2. *Exact Solution Procedures for Problem P1*

The solution space associated with problem P1 is enormous for networks of practical size. The number of ways to partition the row objects can be computed using formulas for the Stirling number of the second kind (Clapham, 1996; Hand, 1981), denoted $SN(N, K)$:

$$SN(N, K) = \frac{1}{K!} \sum_{k=0}^{K} (-1)^k \binom{K}{k} (K - k)^N. \tag{4}$$

This same formula can be used to compute the number of ways to partition the column objects as $SN(M, L)$. Assuming that any partition of row objects can be matched with any partition of the column objects; there are $SN(N, K) \times SN(M, L)$ solutions to the two-mode blockmodeling problem. Even for a small two-mode network such as the southern women social event data (Davis et al., 1941), where $N = 18$ and $M = 14$, the number of solutions for a blockmodel with $K = 3$ and $L = 3$ row and column clusters, respectively, is roughly 50.8 trillion.

In light of the massive solution space for two-mode blockmodeling problems, complete enumeration of all solutions is practical only for very small problems. Brusco and Steinley (2009) formulated the two-mode blockmodeling problem as an integer program, which was used to obtain exact solutions for modestly sized networks. However, this formulation requires a pre-specified image matrix, and is therefore not directly suited for the problem at hand.

3.3. *A Relocation Heuristic for Problem P1*

Doreian et al. (2004, 2005, Chapter 8) presented a relocation heuristic for problem P1. The heuristic applies two neighborhood search operations to an initial blockmodeling solution and continues until there is no operation that will further reduce the number of inconsistencies. The first neighborhood search operation involves transfers of objects from their current cluster to one of the other clusters. The second operation examines exchanges of objects in two different clusters. These operations are evaluated for both row and column objects. Some neighborhood search heuristics accept solutions that improve the objective function as they are generated, which affords efficiency but does not necessarily result in the best move being accepted on any given iteration. Contrastingly, our implementations of the relocation heuristic operate in a greedy fashion. That is, for a given solution, all possible transfers and (if permitted) exchanges of row and column objects are evaluated, and the neighborhood move that provides the greatest reduction in the number of inconsistencies is implemented. This process is repeated until no neighborhood search operation will further reduce the number of inconsistencies. Although the greedy process ensures that an optimal choice is made on each iteration, a globally optimal solution does not necessarily result from this process. The precise steps of the relocation algorithm are as follows:

Step 0. *Initialization*. Randomly generate an initial K-cluster partition of the row objects, π_K, and an initial L-cluster partition for the column objects, ω_L. Compute $g(\pi_K, \omega_L)$ using Equations (1) through (3).

Step 1. *Evaluate all transfers of row objects*.
Step 1a. Compute $g(\pi'_K, \omega_L)$ $\forall \pi'_K = \pi_K \backslash \{R_h, R_k\} \cup \{R_h \backslash \{i\}\} \cup \{R_k \cup \{i\}\} : 1 \le h \le K$ $(i \in R_h : |R_h| > 1)$, and $1 \le k \ne h \le K$.
Step 1b. Set $\delta_1(i', h', k') = \operatorname{argmin}\{g(\pi'_K, \omega_L) : 1 \le h \le K$ $(i \in R_h : |R_h| > 1)$, and $1 \le k \ne h \le K\}$.

Step 2. *Evaluate all transfers of column objects*.
Step 2a. Compute $g(\pi_K, \omega'_L)$ $\forall \omega'_L = \omega_L \backslash \{C_u, C_l\} \cup \{C_u \backslash \{j\}\} \cup \{C_l \cup \{j\}\} : 1 \le u \le L$ $(j \in C_u : |C_u| > 1)$, and $1 \le l \ne u \le L$.
Step 2b. Set $\delta_2(j', u', l') = \operatorname{argmin}\{g(\pi_K, \omega'_L) : 1 \le u \le L$ $(j \in C_u : |C_u| > 1)$, and $1 \le l \ne u \le L\}$.

Step 3. *Evaluate all exchanges of row objects*.
Step 3a. Compute $g(\pi''_K, \omega_L)$ $\forall \pi''_K = \pi_K \backslash \{R_h, R_k\} \cup \{R_h \backslash \{i\} \cup \{v\}\} \cup \{R_k \cup \{i\} \backslash \{v\}\} :$ $1 \le h < k \le K, i \in R_h$, and $v \in R_k$.
Step 3b. Set $\delta_3(i'', v'', h'', k'') = \operatorname{argmin}\{g(\pi''_K, \omega_L) : 1 \le h < k \le K, i \in R_h$, and $v \in R_k\}$.

Step 4. *Evaluate all exchanges of column objects*.
Step 4a. Compute $g(\pi_K, \omega''_L)$ $\forall \omega''_L = \omega_L \backslash \{C_u, C_l\} \cup \{C_u \backslash \{j\} \cup \{w\}\} \cup \{C_l \cup \{w\} \backslash \{j\}\} :$ $1 \le u < l \le K, j \in C_u$, and $w \in C_l$.
Step 4b. Set $\delta_4(j'', w'', u'', l'') = \operatorname{argmin}\{g(\pi_K, \omega''_L) : 1 \le u < l \le L, j \in C_u$, and $w \in C_l\}$.

Step 5. *Find the best neighborhood move*.
Step 5a. Set $\Delta = \min\{\delta_1(i', h', k'), \delta_2(j', u', l'), \delta_3(i'', v'', h'', k''), \delta_4(j'', w'', u'', l'')\}$. If $\Delta \ge g(\pi_K, \omega_L)$, then return π_K, ω_L, and $g(\pi_K, \omega_L)$ and STOP; otherwise proceed to Step 5b.
Step 5b. Let *index* $= z : \delta_z(\bullet) = \Delta$ (break ties based on the first value of z achieving this condition).

Step 6. *Make a relocation*.
Step 6a. If $z = 1$, then set $g(\pi_K, \omega_L) = \Delta$, modify π_K by setting $R_{h'} = R_{h'} \backslash \{i'\}$ and $R_{k'} = R_{k'} \cup \{i'\}$, and return to Step 1.
Step 6b. If $z = 2$, then set $g(\pi_K, \omega_L) = \Delta$, modify ω_L by setting $C_{u'} = C_{u'} \backslash \{j'\}$ and $C_{l'} = C_{l'} \cup \{j'\}$, and return to Step 1.
Step 6c. If $z = 3$, then set $g(\pi_K, \omega_L) = \Delta$, modify π_K by setting $R_{h''} = R_{h''} \backslash \{i''\} \cup \{v''\}$ and $R_{k''} = R_{k''} \cup \{i''\} \backslash \{v''\}$, and return to Step 1.

Step 6d. If $z = 4$, then set $g(\pi_K, \omega_L) = \Delta$, modify ω_L by setting $C_{u''} = C_{u''} \backslash \{j''\} \cup \{w''\}$ and $C_{l''} = C_{l''} \cup \{j''\} \backslash \{u''\}$, and return to Step 1.

The relocation algorithm begins in Step 0 with the generation of an initial blockmodeling solution via assignment of each row and column object to randomly selected clusters. All row-object transfers, column-object transfers, row-object exchanges, and column-object exchanges are evaluated in Steps 1, 2, 3, and 4, respectively. If the best of these neighborhood search operations does not result in a reduction in the number of inconsistencies, then the algorithm terminates in Step 5a. Otherwise, the neighborhood search operation yielding the greatest reduction in the number of inconsistencies is implemented in Step 6 and another iteration of search operations is initiated by returning to Step 1.

It is important to highlight some salient differences between transfers and exchanges. Of the two operations, object transfers are more powerful and efficient. First, object transfers change the sizes of the clusters, whereas exchanges leave the cluster sizes unchanged. Thus, using exchanges alone would be wholly ineffective as a neighborhood search strategy. A second, and related, advantage is that two successive object transfers can replicate what would be achieved by an exchange, whereas two successive exchanges cannot equate to a transfer. In most applications, there are also far fewer transfers to evaluate than exchanges on any given iteration. The number of possible transfers of row objects would typically be $N(K - 1)$ because there are N objects that can be moved to any of the $K - 1$ clusters of which they are not a member. The number of possible exchanges for row objects depends on the cluster sizes. In the case of equal cluster sizes ($N_k = N/K$ for $1 \le k \le K$), there are $(N/K)^2 K(K - 1)/2$ possible exchanges because there are $K(K - 1)/2$ pairs of clusters and, for each pair, any of the N/K objects in one cluster can be exchanged with any of the N/K objects in the other cluster. As an example, consider $N = 200$ row objects and $K = 4$ with 50 objects in each cluster. On each iteration, $200(4 - 1) = 600$ transfers would be evaluated, whereas $4(4 - 1)/2(200/4)^2 = 15{,}000$ exchanges would be tested. In light of the computational costliness of exchanges and their potential for limited value added above and beyond what can be achieved with transfers alone, we will consider versions of the relocation heuristic without exchanges (RH1) and with exchanges (RH2). Our multistart implementations of RH1 and RH2 are constrained by time limit. A time limit constraint makes more sense than constraining the number of restarts because it ensures that the heuristics are allotted the same computational burden in light of the fact that RH2 requires appreciably more time than RH1 per restart. The selection of an appropriate time limit will depend largely on the size of the network matrix. For several small empirical network matrices, we limited the computation time to 1 minute, which was more than sufficient to ensure that each heuristic obtained the best-found solution on at least one restart. Based on the fact that the problems in our simulation study were larger and more difficult, we selected a 10-minute time limit. If an analyst were running only one method for one network matrix, then a much greater time limit could be used for such an implementation. Nevertheless, analysts should be aware that even a run time measured in days or weeks will not guarantee that the global optimum will be found.

4. A Tabu-Search Heuristic for Problem P1

4.1. *The Tabu-Search Algorithm*

Tabu search is a metaheuristic for combinatorial optimization problems (see Glover and Laguna, 1993, for an excellent overview). Like some other metaheuristic approaches (e.g., simulated annealing), the key aspect of the tabu-search process is that it permits escape from a locally optimal solution by accepting solutions that worsen the objective function value. For example, in our minimization context, a neighborhood search heuristic such as RH1 or RH2 can become stuck at a local minimum (or valley). The acceptance of neighborhood moves that increase the objective function can allow the search process to climb out of the valley and then, subsequently, descend in a *deeper* valley that has a better local minimum. In the case of tabu-search process, this is accomplished by forbidding neighborhood moves that would return to the escaped local optimum for a fixed number of iterations. The forbidden moves are referred to as tabu. In our two-mode clustering context, we maintain two tabu tables: (1) $\mathbf{D} = [d_{ik}]$ is an $N \times K$ table with elements d_{ik} representing the number of iterations for which row object i cannot be moved to row cluster k, and (2) $\mathbf{E} = [e_{jl}]$ is an $M \times L$ table with elements e_{jl} representing the number of iterations for which column object j cannot be moved to column cluster l. The two key parameters of the tabu-search heuristic are ψ_0 and η_0. Each time an object is relocated from a cluster, the corresponding tabu table element is set to ψ_0, which defines the number of iterations that returning the object to the cluster will be forbidden. All non-zero tabu elements are reduced by one after each iteration of neighborhood search operations and, if a new best-found solution is identified during an iteration, then all tabu elements are reset to zero. The algorithm terminates when η_0 iterations of the algorithm elapse without improvements in the best-found objective function. The precise steps of our implementation are as follows:

Step 0. *Initialization*. Randomly generate an initial K-cluster partition of the row objects, π_K, and an initial L-cluster partition for the column objects, ω_L. Compute $g(\pi_K, \omega_L)$ using Equations (1) through (3). Set $\pi_K^* = \pi_K, \omega_L^* = \omega_L, g^* = g(\pi_K, \omega_L), \psi_0 = (N + M)/8, \eta_0 = 2 * (N + M)$, $\mathbf{D} = \mathbf{0}_{N\times K}$ (i.e., an $N \times K$ matrix of zeros), and $\mathbf{E} = \mathbf{0}_{M\times L}$ (i.e., an $M \times L$ matrix of zeros) Initialize the iteration counter, $\eta = \eta_0$.

Step 1. *Evaluate all transfers of row objects*.
Step 1a. Compute $g(\pi_K', \omega_L)$ $\forall \pi_K' = \pi_K \backslash \{R_h, R_k\} \cup \{R_h \backslash \{i\}\} \cup \{R_k \cup \{i\}\} : 1 \le h \le K$ $(i \in R_h : |R_h| > 1), 1 \le k \ne h \le K$, and $d_{ik} = 0$.
Step 1b. Set $\delta_1(i', h', k') = \text{argmin}\{g(\pi_K', \omega_L) : 1 \le h \le K$ $(i \in R_h : |R_h| > 1), 1 \le k \ne h \le K$, and $d_{ik} = 0\}$.

Step 2. *Evaluate all transfers of column objects*.
Step 2a. Compute $g(\pi_K, \omega_L')$ $\forall \omega_L' = \omega_L \backslash \{C_u, C_l\} \cup \{C_u \backslash \{j\}\} \cup \{C_l \cup \{j\}\} : 1 \le u \le L$ $(j \in C_u : |C_u| > 1), 1 \le l \ne u \le L$, and $e_{jl} = 0\}$.
Step 2b. Set $\delta_2(j', u', l') = \text{argmin}\{g(\pi_K, \omega_L') : 1 \le u \le L$ $(j \in C_u : |C_u| > 1), 1 \le l \ne u \le L$, and $e_{jl} = 0\}$.

Step 3. *Evaluate all exchanges of row objects*.
Step 3a. Compute $g(\pi_K'', \omega_L)$ $\forall \pi_K'' = \pi_K \backslash \{R_h, R_k\} \cup \{R_h \backslash \{i\} \cup \{v\}\} \cup \{R_k \cup \{i\} \backslash \{v\}\}$: $1 \le h < k \le K, i \in R_h, v \in R_k, d_{vh} = 0$; and $d_{ik} = 0$.
Step 3b. Set $\delta_3(i'', v'', h'', k'') = \text{argmin}\{g(\pi_K'', \omega_L) : 1 \le h < k \le K, i \in R_h, v \in R_k, d_{vh} = 0$; and $d_{ik} = 0\}$.

Step 4. *Evaluate all exchanges of column objects*.
Step 4a. Compute $g(\pi_K, \omega_L'')$ $\forall \omega_L'' = \omega_L \backslash \{C_u, C_l\} \cup \{C_u \backslash \{j\} \cup \{w\}\} \cup \{C_l \cup \{w\} \backslash \{j\}\}$: $1 \le u < l \le K, j \in C_u, w \in C_l, e_{wu} = 0$; and $e_{jl} = 0$.
Step 4b. Set $\delta_4(j'', w'', u'', l'') = \text{argmin}\{g(\pi_K, \omega_L'') : 1 \le u < l \le L, j \in C_u, w \in C_l, e_{wu} = 0$ and $e_{jl} = 0\}$.

Step 5. *Find the best neighborhood move*.
Step 5a. Set $\Delta = \min\{\delta_1(i', h', k'), \delta_2(j', u', l'), \delta_3(i'', v'', h'', k''), \delta_4(j'', w'', u'', l'')\}$.
Step 5b. Let *index* $= z : \delta_z(\bullet) = \Delta$ (break ties based on the first value of z achieving this condition).

Step 6. *Make a row object transfer*?
If $z \ne 1$, then go to Step 7; otherwise proceed to Step 6a.
Step 6a. Set $g(\pi_K, \omega_L) = \Delta$ and modify π_K by setting $R_{h'} = R_{h'} \backslash \{i'\}$ and $R_{k'} = R_{k'} \cup \{i'\}$.

Step 6b. If $g(\pi_K, \omega_L) < g^*$, then set $\eta = \eta_0$, $\mathbf{D} = \mathbf{0}_{N\times K}$, $\mathbf{E} = \mathbf{0}_{M\times L}$, $g^* = g(\pi_K, \omega_L)$, $\pi_K^* = \pi_K$, and go to Step 1; otherwise, proceed to Step 6c.

Step 6c. Set $d_{ik} = \max\{0, d_{ik} - 1\}$ for $1 \le i \le N$ and $1 \le k \le K$, and $e_{jl} = \max\{0, e_{jl} - 1\}$ for $1 \le i \le M$ and $1 \le l \le L$. Set $d_{i'h'} = \omega_0$ and $\eta = \eta - 1$. If $\eta > 0$, then go to Step 1; otherwise return π_K^*, ω_L^*, and g^* and STOP.

Step 7. *Make a column object transfer*?

If $z \neq 2$, then go to Step 8; otherwise proceed to Step 7a.

Step 7a. Set $g(\pi_K, \omega_L) = \Delta$ and modify ω_L by setting $C_{u'} = C_{u'} \backslash \{j'\}$ and $C_{l'} = C_{l'} \cup \{j'\}$.

Step 7b. If $g(\pi_K, \omega_L) < g^*$, then set $\eta = \eta_0$, $\mathbf{D} = \mathbf{0}_{N\times K}$, $\mathbf{E} = \mathbf{0}_{M\times L}$, $g^* = g(\pi_K, \omega_L)$, $\omega_L^* = \omega_L$, and go to Step 1; otherwise proceed to Step 7c.

Step 7c. Set $d_{ik} = \max\{0, d_{ik} - 1\}$ for $1 \le i \le N$ and $1 \le k \le K$, and $e_{jl} = \max\{0, e_{jl} - 1\}$ for $1 \le i \le M$ and $1 \le l \le L$. Set $e_{j'u'} = \omega_0$ and $\eta = \eta - 1$. If $\eta > 0$, then go to Step 1; otherwise return π_K^*, ω_L^*, and g^* and STOP.

Step 8. *Exchange row objects*?

If $z \neq 3$, then go to Step 9; otherwise proceed to Step 8a.

Step 8a. Set $g(\pi_K, \omega_L) = \Delta$ and modify π_K by setting $R_{h''} = R_{h''} \backslash \{i''\} \cup \{v''\}$ and $R_{k''} = R_{k''} \cup \{i''\} \backslash \{v''\}$.

Step 8b. If $g(\pi_K, \omega_L) < g^*$, then set $\eta = \eta_0$, $\mathbf{D} = \mathbf{0}_{N\times K}$, $\mathbf{E} = \mathbf{0}_{M\times L}$, $g^* = g(\pi_K, \omega_L)$, $\pi_K^* = \pi_K$, and go to Step 1; otherwise proceed to Step 8c.

Step 8c. Set $d_{ik} = \max\{0, d_{ik} - 1\}$ for $1 \le i \le N$ and $1 \le k \le K$, and $e_{jl} = \max\{0, e_{jl} - 1\}$ for $1 \le i \le M$ and $1 \le l \le L$. Set $d_{i''h''} = \omega_0$, $d_{v''k''} = \omega_0$, and $\eta = \eta - 1$. If $\eta > 0$, then go to Step 1; otherwise return π_K^*, ω_L^*, and g^* and STOP.

Step 9. *Exchange column objects*?

Step 9a. Set $g(\pi_K, \omega_L) = \Delta$ and modify ω_L by setting $C_{u''} = C_{u''} \backslash \{j''\} \cup \{w''\}$ and $C_{l''} = C_{l''} \cup \{j''\} \backslash \{u''\}$.

Step 9b. If $g(\pi_K, \omega_L) < g^*$, then set $\eta = \eta_0$, $\mathbf{D} = \mathbf{0}_{N\times K}$, $\mathbf{E} = \mathbf{0}_{M\times L}$, $g^* = g(\pi_K, \omega_L)$, $\omega_L^* = \omega_K$, and go to Step 1; otherwise proceed to Step 9c.

Step 9c. Set $d_{ik} = \max\{0, d_{ik} - 1\}$ for $1 \le i \le N$ and $1 \le k \le K$, and $e_{jl} = \max\{0, e_{jl} - 1\}$ for $1 \le i \le M$ and $1 \le l \le L$. Set $e_{w''l''} = \omega_0$, $e_{j''u''} = \omega_0$, and $\eta = \eta - 1$. If $\eta > 0$, then go to Step 1; otherwise return π_K^*, ω_L^*, and g^* and STOP.

The tabu-search algorithm begins in Step 0 with initialization of parameters and the generation of an initial blockmodeling solution via assignment of each row and column object to randomly selected clusters. All row-object transfers, column-object transfers, row-object exchanges, and column-object exchanges are evaluated in Steps 1, 2, 3, and 4, respectively. Steps 1a, 2a, 3a, and 4a incorporate an additional condition not found in the relocation heuristic described in Section 3.3, namely, that tabu moves are not evaluated. The best neighborhood search move across Steps 1 through 4 is identified in Step 5. Unlike the relocation heuristic, moves that worsen the current objective function value may be accepted in Steps 6 through 9. It is this process that enables the tabu-search heuristic to escape from a local optimum. Steps 6 through 9 contain substeps that update the tabu tables and the iteration counter. The algorithm terminates in one of these Steps once the iteration limit is reached.

4.2. *Implementation of the Tabu Search Algorithm*

Like the relocation heuristic, we consider two multistart implementations of the tabu-search heuristic. Version TS1 evaluates only object transfers in Steps 1 and 2 and is, therefore, much faster because it avoids the computationally costly exchanges in Steps 3 and 4. Version TS2 evaluates all transfers and exchanges in Steps 1 through 4. Tabu-search results from previous applications, as well as some experimentation, were used to guide our selection of the ψ_0 and η_0

parameters. De Amorim et al. (1992) used modest parameter settings of $\psi_0 = 5$ and $\eta_0 = N/2$ in their tabu-search application for clique partitioning; however, Brusco and Köhn (2009) found that more aggressive parameter settings of $\psi_0 = N/2$ and $\eta_0 = 6N$ yielded substantially better performance for larger, more difficult problems. Our parameter experiments yielded similar findings for two-mode blockmodeling. Although modest parameter settings are quite sufficient for small network matrices, we adopted settings of $\psi_0 = (N + M)/8$ and $\eta_0 = 2(M + N)$. Increasing the value of ψ_0 much beyond the selected settings often resulted in too many tabu operations at one time, and the solution often began to systematically deteriorate after initial improvements in the local minimum. Larger values of η_0 tended to be prohibitively costly with respect to computation time.

5. Comparisons for Three Empirical Network Matrices

We begin by comparing the performances of RH1, RH2, TS1 and TS2 across three well-known social network matrices from the literature: (1) Davis et al.'s (1941) southern women social event data, (2) the 2000 Supreme Court Justice voting data, and (3) Mische and Pattison's (2000) organization and community project data. In light of the fact that these network matrices are small, we expect that all heuristic methods will obtain optimal blockmodels on at least one restart. Accordingly, our focus will be on the percentage of restarts for which each method obtains the best-found objective function value across all methods. These results will provide some information regarding the attraction of the heuristic methods to the best-found solution.

We applied multistart versions of each heuristic algorithm to each data set for a variety of different value of K and L. The total computation time limit for all methods was one minute on a 2.4 GHz Core 2 Duo Processor with 3 GB of SDRAM. The results are presented in Table 1. All four of the heuristic methods obtained the best-found objective function value (g^*) on at least one of their restarts. The RH1 heuristic is exceptionally fast and achieved far more restarts within the 60-second limit for all test problems. The TS2 heuristic is the least efficient of the methods and always resulted in the fewest restarts. The number of restarts for RH2 was always greater than the corresponding measure for TS1; however, the differences were very modest for the Mische and Pattison (2000) test problems, which were the largest in the set.

With respect to attraction to the best-found objective function value, the results can be summarized as follows: (1) With only one exception (Supreme Court voting data for $K = 5$ and $L = 3$), RH2 obtained the best-found objective function value for a greater percentage of its restarts than RH1, (2) TS1 always obtained the best-found solution for a much greater percentage of its restarts than either RH1 or RH2, and (3) With only two exceptions (social event data $K = 2$ and $L = 3$, and Supreme Court voting data $K = 5$ and $L = 3$), TS2 obtained the best-found objective function value for a greater percentage of its restarts than TS1.

The results in Table 1 clearly reveal that the two versions of the tabu-search heuristic have a much greater attraction rate to the best-found objective value than their relocation heuristic competitors. Nevertheless, given that the relocation heuristics achieve many more restarts within the same time frame, it can be argued that the *number of restarts* (rather than the *percentage of restarts*) on which the best-found objective function value was obtained is a preferred measure of relative performance. If this is the case, then RH1 and TS1 are the best-performing methods, a finding that stems from the computational savings afforded by dropping the exchange routine. The RH1 heuristic obtained the best-found objective function value for more restarts on 8 problems, whereas TS1 did so for 7 problems. The results for the organization and community organization network problems from Mische and Pattison (2000) are especially noteworthy because they foreshadow what will happen as problem size increases further. For the $K = L = 4$ instance of this network, RH1 and TS1 achieved 71,793 and 11,986 restarts, respectively, within

TABLE 1.
Computational results for empirical network matrices.

				Number of restarts[*]				Number of best-found objective values[**]			
	K	L	g	RH1	RH2	TS1	TS2	RH1	RH2	TS1	TS2
Social event data from Davis et al. (1941)	2	3	52	1669649	349319	144883	28364	30548 (1.83)	8936 (2.56)	35926 (24.80)	6841 (24.12)
	3	3	40	722546	171859	85490	17660	14155 (1.96)	7460 (4.34)	21513 (25.16)	8526 (48.28)
	4	3	37	402402	100736	59526	14366	41951 (10.43)	30126 (29.91)	28925 (48.59)	9550 (66.48)
	4	4	33	270420	76266	43833	11810	12750 (4.71)	6839 (8.97)	17776 (40.55)	5877 (49.76)
	5	4	33	193972	62057	34330	10030	26105 (13.46)	13801 (22.24)	14862 (43.29)	5499 (54.83)
Supreme Court voting data from Doreian et al. (2004)	3	3	22	443777	92820	77480	14944	108039 (24.35)	31730 (34.18)	32919 (42.49)	7049 (47.17)
	5	3	20	227386	68783	40817	9664	12177 (5.36)	2369 (3.44)	5165 (12.65)	778 (8.05)
	5	4	16	188011	58458	32352	8198	5486 (2.92)	1997 (3.42)	2117 (6.54)	651 (7.94)
	7	3	20	148599	53420	25318	7202	34290 (23.08)	14829 (27.76)	7740 (30.57)	2842 (39.46)
	7	4	11	116457	41136	20068	5984	424 (0.36)	250 (0.61)	403 (2.01)	191 (3.19)
Organization and community project data from Mische and Pattison (2000)	2	3	130	284159	37767	37279	4451	6664 (2.35)	1004 (2.66)	6609 (17.73)	1314 (29.52)
	3	3	119	154274	24183	23684	3344	8633 (5.60)	3031 (12.53)	10330 (43.62)	1735 (51.88)
	4	4	102	71793	15616	11986	2044	178 (0.25)	69 (0.44)	3691 (30.79)	929 (45.45)
	4	5	95	53712	12921	9003	1797	45 (0.08)	11 (0.09)	2801 (31.11)	760 (42.29)
	5	5	90	40305	10432	7265	1618	586 (1.45)	241 (2.31)	3451 (47.50)	854 (52.78)

[*]The number of restarts for each heuristic within a 1 CPU minute limit.

[**]The number (% in parentheses) of restarts for which each heuristic obtained the best-found objective value.

the 60-second limit. However, despite the fact that RH1 had nearly 6 times the number of restarts, the TS1 heuristic obtained the best-found objective function value for drastically more restarts than RH1 (3,691 vs. 178). A similar result is observed for $K = 4$ and $L = 5$, where TS1 obtained the best-found objective function value for 2,801 of its 9,003 restarts (31.11%), whereas RH1 yielded the best-found objective function value for only 45 of its 53,712 restarts (0.08%).

6. Simulation Experiment—Large Network Matrices

6.1. *Experimental Test Problems*

The relative performances of the heuristic methods in the previous section demonstrate that the relocation and tabu-search heuristics are both capable of yielding optimal (or best-found) solutions for small networks; however, the tabu search has a greater attraction rate to these solutions. In this section, we explore the relative performances of the methods on much larger ill-structured data sets. We conducted an experiment that manipulated four factors at two levels each: (1) the number of row objects at levels of $N = 200$ and $N = 400$, (2) the number of column objects at levels of $M = 200$ and $M = 400$, (3) the number of row clusters at levels of $K = 3$ and $K = 6$, and (4) the number of column clusters at levels of $L = 3$ and $L = 6$. A complete crossing of these four factors yields $2^4 = 16$ cells and we generated 3 replicates (labeled A, B, and C in Table 2) at each cell for a total of 48 unique test problems. For all test problems, the matrix elements were independently generated based on a uniform distribution with equal probability for a value of 0 or 1. The random generation of matrix elements has also been used in other clustering contexts because of its ability to produce difficult test problems that can differentiate the performance of competitive methods (Charon & Hudry, 2006; Brusco & Köhn, 2009).

6.2. *Experimental Results*

We applied multistart versions of each heuristic algorithm to each of the 48 data sets using a total computation time limit of 600 seconds (10 minutes). The results of the simulation experiment are provided in Table 2. The table reports, for each heuristic, the best-found objective function value and the total number or restarts within the 600-second time limit. The best objective function value (across all methods) is highlighted in bold for each test problem. Table 2 unequivocally reveals TS1 as the best-performing method. The TS1 heuristic obtained the best-found objective function value for all 48 problems and none of the other three methods obtained a best-found objective function value for any problem. The TS2 heuristic yielded the second best performance, obtaining a better (worse) objective function value than RH1 for 28 (20) of the test problems. The RH1 heuristic drastically outperformed RH2 for the third place, yielding a better objective function value for 45 of the 48 problems.

The disparity of performance between the TS1 heuristic and its competitors was affected by the levels of the factors in the experiment. For the smallest network matrices in the experiment ($N = 200$, $M = 200$), the number of inconsistencies obtained by RH1, RH2, and TS2 exceeded the number associated with TS1 by averages of 166, 241, and 106, respectively. Contrastingly, for the largest network matrices in the experiment ($N = 400$, $M = 400$), the RH1, RH2, and TS2 averages were 404, 721, and 701, respectively. Although the increased disparity in the number of inconsistencies for all methods for the largest problems is not surprising, it is interesting that degradation in performance when moving from smallest to largest was much more pronounced for TS2 than for RH1. The relative performances of RH1, RH2, and TS2 also declined as the number of clusters increased; however, the effect was much smaller. For the smallest numbers of clusters in the experiment ($K = 3$, $L = 3$), the number of inconsistencies obtained by RH1, RH2, and TS2 exceeded the number associated with TS1 by averages of 226, 374, and 231, respectively. Contrastingly, for the largest numbers of clusters in the experiment ($K = 6$, $L = 6$), the RH1, RH2, and TS2 averages were 390, 598, and 291, respectively. Clearly, the relative performance of TS2 was much more affected by increases in the number of objects than increases in the number of clusters. This is not surprising in light of the massive growth in the number of possible exchanges as N and/or M increase.

Once again, RH1 obtained drastically more restarts than its competitors. The TS2 heuristic incurred the fewest restarts for all problems, often obtaining only one restart for the largest test

TABLE 2.
Computational results for large synthetic network matrices.

Problem*	Best-found objective function value**				Number of restarts***			
	RH1	RH2	TS1	TS2	RH1	RH2	TS1	TS2
A_200/200/3/3	17515	17557	**17422**	17485	2688	61	371	8
B_200/200/3/3	17532	17570	**17434**	17455	2615	61	329	6
C_200/200/3/3	17530	17526	**17408**	17453	2674	64	372	6
A_200/200/3/6	17187	17329	**17040**	17135	1199	40	178	6
B_200/200/3/6	17189	17254	**17031**	17288	1213	41	179	3
C_200/200/3/6	17206	17330	**17053**	17084	1216	41	199	5
A_200/200/6/3	17224	17327	**17004**	17148	1184	40	196	7
B_200/200/6/3	17217	17328	**17065**	17142	1166	41	182	7
C_200/200/6/3	17170	17270	**17017**	17223	1151	40	174	7
A_200/200/6/6	16828	16893	**16600**	16699	646	27	75	2
B_200/200/6/6	16808	16869	**16571**	16671	640	27	71	3
C_200/200/6/6	16815	16865	**16578**	16713	646	26	74	4
A_200/400/3/3	35888	36072	**35683**	35814	843	14	117	2
B_200/400/3/3	35930	35924	**35659**	35705	829	14	102	1
C_200/400/3/3	35846	35960	**35641**	35757	849	13	117	1
A_200/400/3/6	35248	35341	**34973**	35100	385	8	78	1
B_200/400/3/6	35244	35548	**34971**	35636	386	9	80	2
C_200/400/3/6	35298	35448	**34986**	35408	389	9	78	2
A_200/400/6/3	35416	35536	**35090**	35218	376	10	32	2
B_200/400/6/3	35404	35650	**35055**	35383	370	9	37	1
C_200/400/6/3	35412	35576	**35048**	35533	367	10	31	1
A_200/400/6/6	34661	34908	**34297**	34528	206	7	22	1
B_200/400/6/6	34684	34797	**34282**	34471	207	6	21	1
C_200/400/6/6	34616	34777	**34217**	34408	203	6	24	1
A_400/200/3/3	35880	36170	**35618**	35932	873	14	105	1
B_400/200/3/3	35788	36028	**35674**	35715	844	13	105	2
C_400/200/3/3	35816	35944	**35632**	35756	862	15	112	1
A_400/200/3/6	35454	35608	**35044**	35911	402	10	44	2
B_400/200/3/6	35366	35444	**35011**	35722	397	10	47	2
C_400/200/3/6	35356	35320	**35166**	35437	398	9	45	1
A_400/200/6/3	35374	35441	**34932**	35680	384	9	74	2
B_400/200/6/3	35304	35445	**34985**	35497	378	9	79	2
C_400/200/6/3	35276	35324	**34978**	35394	378	9	74	2
A_400/200/6/6	34626	34794	**34289**	34870	214	7	24	1
B_400/200/6/6	34513	34893	**34182**	34394	212	6	24	1
C_400/200/6/6	34649	34821	**34215**	34329	211	6	25	1
A_400/400/3/3	73181	73335	**72811**	73066	312	4	43	1
B_400/400/3/3	73092	73574	**72704**	74223	308	4	44	1
C_400/400/3/3	73152	73264	**72755**	72857	298	4	45	1
A_400/400/3/6	72227	72975	**71853**	72815	139	3	22	1
B_400/400/3/6	72256	72714	**72028**	72681	137	3	23	1
C_400/400/3/6	72210	72509	**72051**	72610	137	2	21	1
A_400/400/6/3	72402	72423	**71834**	71864	129	3	26	1
B_400/400/6/3	72289	72531	**71847**	73276	133	3	18	1
C_400/400/6/3	72116	72335	**71905**	73176	130	3	22	1
A_400/400/6/6	71069	71425	**70528**	70993	73	2	6	1
B_400/400/6/6	71136	71310	**70534**	70689	74	2	8	1
C_400/400/6/6	71142	71685	**70571**	71588	73	2	8	1

*The convention for the problem label is 'replicate_$N/M/K/L'$', where the replicate is A, B or C.

**The best-found objective function value for each heuristic within the 600-second time limit.

***The number of restarts for each heuristic within the 600-second time limit.

problems. It is clear from these results that the greater number of restarts afforded by the relocation heuristic versions is not sufficient to offset the much more vigorous neighborhood search process provided by tabu search. The RH1 heuristic typically had 5 to 10 times more restarts than TS1, but the latter heuristic always yielded a better objective function value.

7. A Two-Mode Blockmodeling Application

To demonstrate the principles of two-mode blockmodeling, we use co-citation data originally analyzed by Groenen and Heiser (1996) using multidimensional scaling and subsequently studied by Brusco and Stahl (2001, 2005) using seriation and unidimensional scaling procedures. In their original form (Groenen & Heiser, p. 547), the data consist of 1991–1992 co-citations among 28 journals that are tied to the field of psychometrics. Although discretizing the raw citation data discards potentially useful information (see, for example, Žiberna, 2007), this is necessary given our focus on binary data. For the application herein, we converted this matrix to binary form by setting $x_{ij} = 1$ if the number of times that journal j cited journal i equaled or exceeded 10, or $x_{ij} = 0$ otherwise, for $1 \leq i \leq N = 28$ and $1 \leq j \leq M = 28$. At face value it appears that the co-citation matrix is one-mode data; however, we adopt the practice of treating the data as two mode, with $N = 28$ receiving (or cited) journals and $M = 28$ sending (or citing journals). This practice has precedent in the blockmodeling literature (Doreian et al., 2005, Chapter 8).

The RH1 and TS1 algorithms (60-second time limit) were applied to the network matrix for a variety of different values of K and L. The results of this analysis are reported in Table 3. For each combination of K and L, both heuristics obtained the best-found objective function value on at least one restart. As expected, RH1 achieved drastically more restarts than TS1 within the 60-second time. Nevertheless, in most instances, TS1 obtained the best-found objective function

TABLE 3.
Computational results for the co-citation data at different values of K and L.

K	L	Best-found objective value		Total number of restarts[*]		Number of best-found objective values[**]	
		RH1	TS1	RH1	TS1	RH1	TS1
2	2	166	166	3239414	56961	8350	653
2	3	163	163	1857717	37221	30	60
3	2	138	138	1432789	32334	242	369
3	3	129	129	591559	22714	655	870
3	4	121	121	229976	15371	510	475
4	3	115	115	264511	16347	63	205
4	4	105	105	110687	10804	51	340
4	5	99	99	58231	7724	37	478
5	4	94	94	66274	7974	25	424
5	5	89	89	37242	5741	2	255
5	6	86	86	24384	4497	32	467
6	5	81	81	24734	4436	40	294
6	6	78	78	17434	3493	25	328
7	7	71	71	10679	2339	4	81

[*]The number of restarts for each heuristic within a 1 CPU minute limit.

[**]The number of restarts for which each heuristic obtained the best-found objective function values.

value on far more restarts than RH1. Moreover, the number of times that RH1 obtained the best-found objective function value was disconcertingly low for some combinations of K and L. For example, at $K = L = 5$, RH1 obtained the best-found objective function value for only 2 of its 37,242 restarts, whereas TS1 did so for 255 of its 5,741 restarts. This finding highlights the importance of having an effective heuristic procedure as problem size increases.

The best-found number of inconsistencies for a blockmodel with $K = 2$ row clusters and $L = 2$ column clusters was 166. Increasing the number of column clusters to $L = 3$ (with K remaining at two) provided only a small reduction to 163 inconsistencies; however, increasing the number of row clusters to $K = 3$ (with L remaining at two) provided a sharp reduction to 138 inconsistencies. This pattern of stronger reductions stemming from increases in the number of row clusters continues throughout Table 3. There were 89 inconsistencies for the best-found blockmodel at $K = L = 5$. Increasing the number of row clusters to $K = 6$ (holding at $L = 5$) provided a fairly sharp reduction to 81 inconsistencies. Moving to $K = L = 6$ only reduced the number of inconsistencies to 78 and, even at $K = L = 7$, the number of inconsistencies dipped to only 71. Based on the last fairly sizable reduction occurring at $K = 6$ and $L = 5$, as well as solution interpretability, we adopted the ($K = 6, L = 5$) blockmodel for further analysis. The image matrix corresponding to this solution is

Complete	Complete	Null	Null	Null
Complete	Complete	Complete	Complete	Complete
Null	Complete	Complete	Null	Complete
Null	Null	Null	Complete	Complete
Null	Null	Null	Null	Complete
Null	Null	Null	Null	Null

A permuted version of the network matrix is provided in Table 4 to reveal the journal clusters and the block structure. Recall that the row objects are the cited journals and the column objects are the citing journals. The first cluster of the row objects (cited journals) consists of the premiere statistical outlets (*Annals of Statistics*, *Applied Statistics*, *Biometrics*, *Biometrika*, *Journal of the American Statistical Association*, *Journal of the Royal Statistical Society*). These journals are cited heavily by the first two clusters of column objects, which consist principally of statistical journals and statistical psychology journals, respectively, resulting in near-complete blocks in these two instances. The premiere statistical journals are not cited heavily by the non-statistically oriented psychology outlets in the latter three clusters of citing journals and, therefore, those blocks are mostly null.

The second cluster of the row objects (cited journals) consists only of *Psychometrika*. This journal forms its own cluster of cited journals because it is heavily cited by almost all journals in the sample, including the major statistical outlets, the statistical psychology outlets, the more general psychological outlets, and other journals in the quantitative social sciences. This distinguishes *Psychometrika* from other statistical psychology outlets in the third cluster of cited journals (*Applied Psychological Measurement*, *British Journal of Mathematical and Statistical Psychology*, *Educational and Psychological Measurement*, *Multivariate Behavioral Research*, *Psychological Bulletin*), which are not as heavily cited by the leading statistical journals and/or some of the mainstream psychological outlets.

The fourth cluster of cited journals consists of four elite psychological journals (*Annual Review of Psychology*, *Journal of Mathematical Psychology*, *Perception & Psychophysics*, and *Psychological Review*). These journals are not cited heavily by the more statistically oriented outlets; however, they are cited heavily by two clusters of citing journals with a psychological emphasis: (1) the fourth cluster of citing journals that consists of (*Annual Review of Psychology*, *Journal of Mathematical Psychology*, *Perception & Psychophysics*, *Psychological Review*, and

TABLE 4.
Permuted co-citation matrix (rows are cited journals, columns are citing journals)—inconsistencies shown in bold.

	5	6	7	8	10	11	14	19	28	15	17	1	9	21	3	12	13	16	27	4	18	20	22	26	2	23	24	25
5. Ann Statist	1	1	1	1	1	1	1	1	1	**0**	**0**	1	1	1	0	0	0	0	**1**	0	**1**	0	0	0	0	0	0	0
6. Appl Stat	1	1	1	1	1	1	1	1	1	**0**	**0**	1	1	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7. Biometrics	1	1	1	1	**0**	1	1	1	1	1	**0**	1	1	**0**	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0
8. Biometrika	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	**1**	0	0	0	0	0	0	0	0	0	0	**1**	0
14. J Am Stat A	1	1	1	1	1	1	1	1	1	1	1	1	1	1	**1**	**1**	**1**	0	**1**	0	0	0	0	0	0	0	**1**	0
19. J Roy Sta B	1	1	1	1	1	1	1	1	1	**0**	1	1	1	**0**	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0
1. Psychometrika	**0**	**0**	1	**0**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3. Appl Psych M	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	**0**	1	**0**	0	0	0	0	0	**0**	1	1	**0**
9. Br J Math S	0	0	0	0	0	0	0	0	0	0	**1**	1	1	1	1	1	**0**	**0**	1	**1**	0	0	0	0	**0**	1	1	**0**
12. Educ Psyc M	0	0	0	0	0	0	0	0	0	0	0	**0**	1	1	1	1	**0**	1	**0**	0	0	0	0	0	1	1	1	1
21. Multiv Be R	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	**0**	1	0	0	0	0	0	1	1	1	1
24. Psychol B	0	0	0	0	0	0	0	0	0	0	**1**	1	1	1	1	1	1	1	1	**1**	**1**	0	**1**	**1**	1	1	1	1
4. Ann R Psych	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	**0**	1	1	1	1	1	1
18. J Math Psyc	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	**1**	0	0	0	0	1	1	1	1	1	**0**	**0**	1	**0**
22. Perc Psyc	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	**0**
26. Psychol Rev	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	**1**	0	0	0	0	1	1	1	1	1	1	1	1	1
2. Perc Mot Sk	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	1	1	1	1
23. Pers Indiv	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	1	1	1	1
25. Psychol Rep	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	1	1	1	1
10. Chem Intell	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11. Comput Stat	0	0	0	0	0	**1**	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13. Exp Aging R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	**1**	0	0	0	0	0	0	0	0
15. J Classif	0	0	0	0	0	0	0	0	0	**1**	0	**1**	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0
16. J Educ Meas	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	0	**1**	**1**	0	**1**	0	0	0	0	0	0	0	0	0	**1**
17. J Educ Stat	0	0	0	0	0	0	0	0	0	0	**1**	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20. Math Soc Sc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	0	0	0	0	0
27. Sociol Meth	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0
28. Statisticia	0	0	**1**	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Mathematical Social Sciences), and (2) the fifth cluster of citing journals that consists of (*Perceptual and Motor Skills, Personality and Individual Differences, Psychological Bulletin*, and *Psychological Reports*).

The fifth cluster of cited journals consists of three other psychological journals (*Perceptual and Motor Skills, Personality and Individual Differences,* and *Psychological Reports*). Unlike the elite journals in the fourth cluster of cited journals, the three journals in the fifth cluster of cited journals are cited heavily by only the fifth cluster of citing journals, but not the fourth. That is, although elite journals such as *Psychological Review* and *Annual Review of Psychology* are heavily cited by all of the psychologically oriented outlets, these journals tend to heavily cite only other elite outlets. The sixth cluster of cited journals consists of a mixture of different types of journals that were not heavily cited by any of the clusters of citing journals and, therefore, all blocks for this cluster of row objects are mostly null.

In the blockmodeling literature, it is common to use the number of equally well-fitting partitions as a criterion for blockmodel selection (Doreian et al., 2005). In light of the fact that we are using heuristic algorithms, we cannot definitively determine the number of partitions that yield the same objective function value for each (K, L) pair; however, we only found two equally well-fitting partitions for $(K = 6, L = 5)$ at 81 consistencies. The partition of the cited journals (rows) was the same, but two different partitions were observed for the citing journals (columns). The difference was very subtle, and is indicated by the fact that *Multivariate Behavioral Research* (journal #21) can "float" from the second citing-journal cluster (with *Psychometrika* and *British Journal of Mathematical and Statistical Psychology*) to the third citing journal with (*Applied Psychological Measurement*, *Educational and Psychological Measurement*, *Experimental Aging Research*, *Journal of Educational Measurement*, and *Sociological Methodology*), without increasing the number of inconsistencies. This is easy to observe from Table 4 by envisioning a move of the solid vertical line to the right of column label 21 to the left of column label 21. In light of the fact that only two equally well-fitting partitions were identified, with the difference in the solution being the presence of only one floating journal, we are confident in our selection of the $(K = 6, L = 5)$ blockmodeling solution.

As a final caveat to our results, we return to the potential problems associated with discretizing raw data, as noted by Žiberna (2007). We re-ran our blockmodeling analyses using cutoffs of 5 and 15 citations instead of 10. The increase to a cutoff of 15 reduced the density of the network matrix from 31% to 26%. Although there were some subtle changes in the assignments of citing and cited journals, the basic structure of the blockmodel was comparable. However, the decrease to a cutoff of 5 increased density from 31% to 41% and the changes in the block structure were more pronounced. In light of these findings, the cautions offered by Žiberna (2007) regarding the use of cutoff values to discretize data seem to hold merit for this application.

8. Summary and Extensions

We have developed a new tabu-search heuristic for two-mode blockmodeling and compared two versions of the new procedure to two versions of an existing relocation algorithm. The results showed that all methods performed well for small empirical network matrices; however, the tabu-search heuristics obtained the globally optimal and/or best-found solution on a far greater percentage of their restarts. In a simulation study that focused on 48 large network matrices, a version of the tabu-search heuristic that uses only object transfers (TS1) always yielded a blockmodel with a better objective function value than any of its competitors when all methods were constrained to the same 10-minute time limit. Our conclusion based on these findings, is that the ability of the relocation heuristic to achieve far more restarts than the tabu-search heuristic within the same time limit is more than offset by the more sophisticated neighborhood search capabilities of the tabu-search procedure. Nevertheless, an important limitation of our experiment is that we do not know the proximity of the TS1 objective function value to the globally optimal objective function value.

There are several possible avenues for extending the results of this paper. One such extension is the integration of the tabu-search procedure within the framework of variable neighborhood search (see, for example, Brusco & Köhn, 2009). A second, and potentially important, advancement centers on the development of improved procedures for generating initial blockmodeling solutions. It is clear that some randomly generated starting solutions can lead to poor results regardless of the method employed. Possible strategies for obtaining initial solutions include exchangeable Rasch models (Lauritzen, 2008), or the *Bimax* algorithm developed by Prelić et al. (2006) that is currently available in the R toolbox for biclustering assembled by Kaiser and Leisch (2008). The adaptation and testing of the tabu-search procedure for other blockmodeling contexts (e.g., for signed graphs) would be a worthwhile endeavor.

Finally, comparison of stochastic (model-based) and deterministic (non-model-based) blockmodeling methods remains an important, yet formidable, challenge. Such comparisons often spark considerable debate among different camps of researchers who have developed or advocated methods in each of these classes. A forthcoming example is a recent evaluation of (Gaussian) mixture-model clustering in comparison to the popular non-model-based K-means clustering method (Steinley & Brusco, 2011a), the subsequent commentaries on that comparison (McLachlan, 2011; Vermunt, 2011), and the rejoinder (Steinley & Brusco, 2011b). In light of the need for equity and fairness in such comparisons, great care and considerable thoroughness must be taken in the development of the experimental design. For this reason, we believe that the best comparisons of deterministic and stochastic blockmodels will be offered in contributions dedicated solely to that topic, as opposed to appendages to papers presenting a new methodology.

References

Aarts, E., & Korst, J. (1989). *Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing*. New York: Wiley.

Airoldi, E.M., Blei, D.M., Fienberg, S.E., & Xing, E.P. (2008). Mixed-membership stochastic blockmodels. *Journal of Machine Learning Research*, *9*, 1981–2014.

Arabie, P., Hubert, L., & Schleutermann, S. (1990). Blockmodels from the bond energy algorithm. *Social Networks*, *12*, 99–126.

Batagelj, V., Ferligoj, A., & Doreian, P. (1992). Direct and indirect methods for structural equivalence. *Social Networks*, *14*, 63–90.

Bickel, P.J., & Chen, A. (2009). A nonparametric view of network models and Newman-Girvan and other modularities. *Proceedings of the National Academy of Sciences*, *106*, 21068–21073.

Borgatti, S.P., & Everett, M.G. (1997). Network analysis of 2-mode data. *Social Networks*, *19*, 243–269.

Borgatti, S.P., Everett, M.G., & Freeman, L. (2002). *Ucinet for Windows: software for social network analysis*. Harvard: Analytic Technologies.

Breiger, R.L., Boorman, S.A., & Arabie, P. (1975). An algorithm for clustering relational data with applications to social network analysis and comparison to multidimensional scaling. *Journal of Mathematical Psychology*, *12*, 328–383.

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. *Sociological Methods & Research*, *40*, 57–87.

Brusco, M.J., & Köhn, H.F. (2009). Clustering qualitative data based on binary equivalence relations: neighborhood search heuristics for the clique partitioning problem. *Psychometrika*, *74*, 685–703.

Brusco, M.J., & Stahl, S. (2001). An interactive approach to multiobjective combinatorial data analysis. *Psychometrika*, *66*, 5–24.

Brusco, M.J., & Stahl, S. (2005). Optimal least-squares unidimensional scaling: improved branch-and-bound procedures and comparison to dynamic programming. *Psychometrika*, *70*, 253–270.

Brusco, M., & Steinley, D. (2006). Inducing a blockmodel structure of two-mode binary data using seriation procedures. *Journal of Mathematical Psychology*, *50*, 468–477.

Brusco, M., & Steinley, D. (2007a). A variable neighborhood search method for generalized blockmodeling of two-mode binary matrices. *Journal of Mathematical Psychology*, *51*, 325–338.

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

Brusco, M.J., & Steinley, D. (2009). Integer programs for one- and two-mode blockmodeling based on prespecified image matrices for structural and regular equivalence. *Journal of Mathematical Psychology*, *53*, 577–585.

Brusco, M., & Steinley, D. (2010). K-balance partitioning: an exact method with application to generalized structural balance and other psychological contexts. *Psychological Methods*, *15*, 145–157.

Charon, I., & Hudry, O. (2006). Noising methods for a clique partitioning problem. *Discrete Applied Mathematics*, *154*, 754–769.

Clapham, C. (1996). *The concise Oxford dictionary of mathematics* (2nd ed.). Oxford: Oxford University Press.

Davis, A., Gardner, B., & Gardner, M.R. (1941). *Deep south*. Chicago: University of Chicago Press.

De Amorim, S.G., Barthélemy, J.-P., & Ribeiro, C.C. (1992). Clustering and clique partitioning: simulated annealing and tabu search approaches. *Journal of Classification*, *9*, 17–41.

Doreian, P., Batagelj, V., & Ferligoj, A. (2004). Generalized blockmodeling of two-mode network data. *Social Networks*, *26*, 29–53.

Doreian, P., Batagelj, V., & Ferligoj, A. (2005). *Generalized blockmodeling*. Cambridge: Cambridge University Press.

Doreian, P., & Mrvar, A. (1996). A partitioning approach to structural balance. *Social Networks*, *18*, 149–168.

Doreian, P., & Mrvar, A. (2009). Partitioning signed social networks. *Social Networks*, *31*, 1–11.

Fienberg, S.E., Meyer, M.M., & Wasserman, S.S. (1985). Statistical analysis of multiple sociometric relations. *Journal of the American Statistical Association*, *80*, 51–67.

Freeman, L.C. (2003). Finding social groups: a meta-analysis of the Southern Women data. In R. Brieger, C. Carley, & P. Pattison (Eds.), *Dynamic social network modeling and analysis: workshop summary and papers* (pp. 39–97). Washington: The National Academies Press.

Glover, F., & Laguna, M. (1993). Tabu search. In C. Reeves (Ed.), *Modern heuristic techniques for combinatorial problems* (pp. 70–141). Oxford: Blackwell.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization, and machine learning*. New York: Addison-Wesley.

Goldenberg, A., Zheng, A., Fienberg, S.E., & Airoldi, E.M. (2009). A survey of statistical network models. *Foundations and Trends in Machine Learning*, *2*, 1–117.

Govaert, G., & Nadif, M. (2003). Clustering with block mixture models. *Pattern Recognition*, *36*, 463–473.

Groenen, P.J.F., & Heiser, W.J. (1996). The tunneling method for global optimization in multidimensional scaling. *Psychometrika*, *61*, 529–550.

Hand, D.J. (1981). *Discrimination and classification*. New York: Wiley.

Handcock, M.S., Raftery, A.E., & Tantrum, J. (2007). Model-based clustering for social networks (with discussion). *Journal of the Royal Statistical Society A*, *170*, 301–354.

Hartigan, J. (1972). Direct clustering of a data matrix. *Journal of the American Statistical Association*, *67*, 123–129.

Heller, J. (1961). *Catch-22: a novel*. New York: Simon and Schuster.

Holland, P.W., & Leinhardt, S. (1976). Local structure in social networks. *Sociological Methodology*, *7*, 1–45.

Holland, P.W., & Leinhardt, S. (1977). A dynamic model for social networks. *Journal of Mathematical Sociology*, *5*, 5–20.

Holland, P.W., & Leinhardt, S. (1981). An exponential family of probability distributions for directed graphs (with discussion). *Journal of the American Statistical Association*, *76*, 33–65.

Holland, P.W., Laskey, K.B., & Leinhardt, S. (1983). Stochastic blockmodels: first steps. *Social Networks*, *5*, 109–137.

Homans, G.C. (1950). *The human group*. New York: Harcourt-Brace.

Kaiser, S., & Leisch, F. (2008). *A toolbox for bicluster analysis in R* (Technical Report 028). Munich: Department of Statistics, University of Munich.

Koyutürk, M., Szpanowski, W., & Grama, A. (2004). Biclustering gene-feature matrices for statistically significant patterns. In *Proceedings of the 2004 IEEE computational systems bioinformatics conference* (pp. 480–484). Washington: IEEE Comput. Soc.

Krolak-Schwerdt, S. (2003). Two-mode clustering methods: compare and contrast. In M. Schader, W. Gaul, & M. Vichi (Eds.), *Between data science and applied data analysis* (pp. 270–278). Berlin: Springer.

Lauritzen, S.L. (2008). Exchangeable Rasch matrices. *Rendiconti di Matematica*, *28*, 83–95.

Lorrain, F., & White, H.C. (1971). Structural equivalence of individuals in social networks. *Journal of Mathematical Sociology*, *1*, 49–80.

Madeira, S.C., & Oliveira, A.L. (2004). Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Informatics*, *1*, 24–45.

McLachlan, G.J. (2011). Commentary on Steinley and Brusco (2011): recommendations and cautions. *Psychological Methods*, *16*, 80–81.

Mische, A., & Pattison, P. (2000). Composing a civic arena: publics, projects, and social settings. *Poetics*, *27*, 163–194.

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

Mrvar, A., & Doreian, P. (2009). Partitioning signed two-mode networks. *Journal of Mathematical Sociology*, *33*, 196–221.

Mulvey, J.M., & Crowder, H.P. (1979). Cluster analysis: an application of Lagrangian relaxation. *Management Science*, *25*, 329–340.

Newman, M.E.J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, *69*, 026113. doi:10.1103/PhysRevE.69.026113.

Nowicki, K., & Snijders, T.A.B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, *96*, 1077–1087.

Pacheco, J., & Valencia, O. (2003). Design of hybrids for the minimum sum-of-squares clustering problem. *Computational Statistics and Data Analysis*, *43*, 235–248.

Pattison, P.E., & Breiger, R.L. (2002). Lattices and dimensional representations: matrix decompositions and ordering structures. *Social Networks*, *24*, 423–444.

Prelić, A., Blueler, S., Zimmerman, P., Wille, A., Bühlmann, P., Gruissem, W., Hennig, L., Thiele, L., & Zitzler, E. (2006). A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, *22*, 1122–1129.

Rolland, E., Schilling, D.A., & Current, J.R. (1996). An efficient tabu search procedure for the p-median problem. *European Journal of Operational Research*, *96*, 329–342.

Steinhaus, H. (1956). Sur la division des corps matétiels en parties. *Bulletin de l'Académie Polonaise des Sciences, Classe III*, *IV*(12), 801–804.

Steinley, D., & Brusco, M.J. (2011a). An evaluation of mixture model clustering: recommendations and cautions. *Psychological Methods*, *16*, 63–79.

Steinley, D., & Brusco, M.J. (2011b). K-means clustering and mixture model clustering: reply to McLachlan and Vermunt. *Psychological Methods*, *16*, 89–92.

van Rosmalen, J., Groenen, P.J.F., Trejos, P., & Castillo, W. (2009). Optimization strategies for two-mode partitioning. *Journal of Classification*, *26*, 155–181.

van Uitert, M., Meuleman, W., & Wessels, L. (2008). Biclustering sparse binary genomic data. *Journal of Computational Biology*, *15*, 1329–1345.

Vermunt, J. (2011). K-means may perform as well as mixture model clustering but may also be much worse: comment on Steinley and Brusco (2011). *Psychological Methods*, *16*, 82–88.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistical Computing*, *17*, 395–416.

Ward, J.H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, *58*, 236–244.

Wasserman, S., & Faust, K. (1994). *Social network analysis: methods and applications*. Cambridge: Cambridge University Press.

White, D.R., & Reitz, K.P. (1983). Graph and semigroup homomorphisms on networks of relations. *Social Networks*, *5*, 193–234.

Xu, W., Liu, X., & Gong, Y. (2003). Document clustering based on non-negative matrix factorization. In *SIGIR'03: proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 267–273). New York: Association for Computing Machinery.

Žiberna, A. (2007). Generalized blockmodeling of valued networks. *Social Networks*, *29*, 105–126.

Manuscript Received: 24 MAY 2010
Final Version Received: 18 OCT 2010
Published Online Date: 14 JUL 2011