Journal of Mathematical Sociology, 37: 61–84, 2013
Copyright © Taylor & Francis Group, LLC
ISSN: 0022-250X print/1545-5874 online
DOI: 10.1080/0022250X.2011.597278

AN EXACT ALGORITHM FOR BLOCKMODELING OF TWO-MODE NETWORK DATA

Michael Brusco
College of Business, Florida State University, Tallahassee, Florida, USA

Patrick Doreian
Department of Sociology, University of Pittsburgh, Pittsburgh, Pennsylvania, USA, and Faculty of Social Sciences, University of Ljubljana, Ljubljana, Slovenia

Andrej Mrvar
Faculty of Social Sciences, University of Ljubljana, Ljubljana, Slovenia

Douglas Steinley
Department of Psychological Sciences, University of Missouri–Columbia, Columbia, Missouri, USA

We consider problems where relationships between two sets (or modes) of objects are available in the form of a binary matrix with elements of 1 (0) indicating a bond (lack of a bond) between corresponding row and column objects. The goal is to establish a partition of the row objects and, simultaneously, a partition of the column objects to form blocks that consist of either exclusively 1s or exclusively 0s to the greatest extent possible. This **two-mode blockmodeling problem** ***arises in several scientific domains. In the social sciences, two-mode blockmodeling is particularly relevant for social network analysis, where the goal is to simultaneously partition a set of individuals and another set of objects (e.g., events they attended, organizations they are affiliated with, etc.). The inherent computational challenge of simultaneously constructing partitions for two distinct sets of objects has fostered a reliance on heuristics for two-mode blockmodeling. We offer an exact algorithm and demonstrate its efficacy in a simulation study. Two applications to real-world networks are also provided.***

Keywords: branch-and-bound, exact algorithms, generalized blockmodeling, social networks, two-mode networks

1. INTRODUCTION

The problem of partitioning objects into two or more clusters arises in many areas of scientific inquiry. In most applications, partitioning methods are used to obtain groups for a single set of objects. For this reason, such methods are sometimes referred to as *one-mode partitioning* procedures. Although receiving less

Address correspondence to Patrick Doreian, Department of Sociology, University of Pittsburgh, 2602 WWPH, 230 Bouquet St., Pittsburgh, PA, 15260, USA. E-mail: pitpat@pitt.edu

attention, there are many circumstances in the social sciences where there are *two* distinct sets of objects that are of importance (Borgatti & Everett, 1992, 1997; Latapy, Magnien, & Del Vecchio, 2008). Moreover, in many of these applications, the data are available in the form of a *two-mode* binary matrix, where elements of 1 indicate the presence of a bond (or tie) between the row object and column object, and elements of 0 indicate the lack of a bond. Examples of two-mode binary social network data include the attendance of women at various social events (Davis, Gardner, & Gardner, 1941), affiliation of executives with various clubs/boards (Faust, 1997; Galaskiewicz, 1985), the participation of civic organizations in various community projects (Brusco & Steinley, 2006; Mische & Pattison, 2000), and the voting patterns of Supreme Court justices on a set of cases (Doreian, Batagelj, & Ferlogoj, 2004, 2005, Chapter 8).

In light of the importance of two-mode binary data, it is not surprising that a considerable amount of research has focused on the development of methods for such data (Arabie, Hubert, & Schleutermann, 1990; Batagelj, Mrvar, Ferligoj, & Doreian, 2004; Borgatti & Everett, 1992, 1997; Brieger, 1974; Brusco & Steinley, 2006, 2007, 2009; Doreian, 1979; Doreian et al., 2004, 2005; Hubert, 1974; Latapy et al., 2008; Mrvar & Doreian, 2009; Wasserman & Faust, 1994, Chapter 2). The *two-mode partitioning problem* within the context of binary network data requires the establishment of a partition of the row objects and, simultaneously, the column objects. The intersections of the row and column clusters form blocks, and a common goal is to obtain a partition that tends to produce blocks that consists of all 1s (or all 0s) to the greatest extent possible. This objective is concordant with, and is a generalization of, the principle of structural equivalence in the social network literature (Arabie, Boorman, & Levitt, 1978; Brieger, Boorman, & Arabie, 1975; Lorrain & White, 1971; Sailer, 1978; White, Boorman, & Brieger, 1976). For this reason, we adopt the nomenclature of social network analysis (see, e.g., Doreian et al., 2004, 2005, Chapter 8) and refer to this partitioning context as *two-mode blockmodeling*. Moreover, we limit our focus to the case of *exploratory* two-mode blockmodeling (see Doreian et al., 2005, Chapter 8), where it is assumed that the ideal placement of complete and null blocks is not known a priori and must be determined as part of the solution process.

Like other combinatorial partitioning problems, solution methods for two-mode blockmodeling fall into one of two categories: (a) approximate methods, also known as heuristics, or (b) exact methods. The principal distinction between these two classes of methods is that exact procedures guarantee a globally optimal solution upon convergence, whereas heuristic procedures do not afford such a guarantee. Heuristic methods for partitioning problems abound in the literature and include standard object reassignment methods such as *K*-means clustering (MacQueen, 1967; Steinhaus, 1956), as well as more sophisticated metaheuristics such as simulated annealing (Kirkpatrick, Gelatt, & Vecchi, 1983), tabu search (Glover, 1989, 1990), genetic algorithms (Goldberg, 1989), and variable neighborhood search (Mladenović & Hansen, 1997). Within the context of two-mode blockmodeling, Doreian et al. (2004) developed a relocation heuristic that produces blockmodels that are locally optimal with respect to all possible transfers of row and column objects from their current cluster to one of the other clusters, as well as all possible exchanges of cluster memberships for pairs of objects not in the same cluster. The relocation heuristic has the advantages of computational efficiency and flexibility. Moreover, it does not require the ideal block structure to be specified in advance and is, therefore, well-suited for exploratory blockmodeling. While the relocation heuristic can also be applied for a deductive use of blockmodeling, we do not consider that usage here.

When computationally feasible, exact partitioning methods have several important advantages, which have recently been articulated by Brusco, Doreian, Mrvar, and Steinley (2011). First, there is the obvious benefit of the guarantee of global optimality for applied partitioning problems. Second, the branch-and-bound approach can be modified easily to identify all of the equally well-fitting partitions. That is, the modified program can obtain all partitions that minimize the number of inconsistencies. Third, the ability of exact methods to produce globally optimal solutions for problems of nontrivial size can be used to facilitate the evaluation of heuristic procedures. Recently, there has been considerable progress in the development of exact methods for one-mode clustering problems, such as within-cluster sum of squares partitioning (Brusco, 2006; Du Merle, Hansen, Jaumard, & Mladenović, 2000) and *p*-median partitioning (Avella, Sassano, & Vasil'ev, 2007). However, the inherent computational complexity associated with the need to simultaneously establish partitions for two distinct sets of objects would seem to present a formidable challenge to the construction of exact procedures for two-mode blockmodeling. One step in this direction was recently taken by Brusco and Steinley (2009), who formulated the two-mode blockmodeling problem as an integer linear program. Although their approach does guarantee a globally optimal solution to the two-mode blockmodeling problem, it has several disadvantages: (a) it requires prespecification of the ideal block structure and is not readily suitable for exploratory blockmodeling, (b) integer programming software is required to implement the method, and (c) the linearization of the bilevel integer programming formulation used by Brusco and Steinley offers a weak LP relaxation and is often computationally demanding even for rather small problems.

Our contribution in this article is the development of a standalone exact solution procedure for two-mode blockmodeling of binary matrices that is based on the principles of branch-and-bound programming. In doing so, we confine attention to structural equivalence where only complete and null blocks are permitted. Although the proposed algorithm shares some similarities with branch-and-bound methods proposed for other blockmodeling/partitioning problems (Brusco, 2003, 2006; Brusco et al., 2011; Brusco & Steinley, 2010; Diehr, 1985; Klein & Aronson, 1991; Koontz, Narendra, & Fukunaga, 1975), it is critical to observe that all of these previous applications were for one-mode partitioning. The current extension to two-mode blockmodeling required resolution of nontrivial issues, such as maintaining two distinct object lists, allowing different numbers of clusters for the two object sets, alternating between object sets in the construction of the branching tree, and establishing strong bounds. Although the proposed algorithm does encounter the same scalability limitations faced by the branch-and-bound methods for one-mode blockmodeling/partitioning problems, we have found that it is capable of providing globally optimal two-mode blockmodels for many data sets from the social network literature.

It is important from the outset to note that we do not propose this new algorithm as a replacement for the heuristic relocation algorithm developed by Doreian et al. (2004). In fact, we recommend using the relocation algorithm prior to our branch-and-bound algorithm to provide an initial bound on the objective function. This tandem approach of the relocation heuristic followed by branch-and-bound can be used to obtain guaranteed optimal solutions within the context of exploratory blockmodeling. In Section 2 of this article, we provide a precise mathematical programming formulation of the two-mode blockmodeling problem. Section 3 presents the branch-and-bound algorithm that we propose for two-mode blockmodeling. The results of a simulation study are offered in Section 4. Sections 5 and 6 report results that we obtained when applying the algorithm to two data sets from the social network literature. The first application (Section 5) is for an affiliation network associated with the memberships of CEOs in various clubs and boards (Galaskiewicz, 1985). The second example (Section 6) corresponds to Search and Rescue (SAR) communication network data originally collected by Drabek, Tamminga, Kilijanek, and Adams (1981) and studied by Doreian et al. (2005, Chapter 6). A brief summary and discussion is provided in Section 7.

2. FORMULATION OF THE TWO-MODE BLOCKMODELING PROBLEM

2.1. Formulation of Two-Mode Blockmodeling (Problem P1)

We present a representation of two-mode blockmodeling as a discrete optimization problem. Throughout the remainder of this section, we use notation that is concordant with that used by Brusco and Steinley (2007):

R: is a set of N row objects indexed $\{1, 2, \ldots, N\}$;
C: is a set of M column objects indexed $\{1, 2, \ldots, M\}$;
$\mathbf{X}$: 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: $\pi_K = \{R_1, R_2, \ldots, R_K\}$ 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 $N_k = |R_k|$ 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; and
ω_L: $\omega_L = \{C_1, C_2, \ldots, C_L\}$ 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 $M_l = |C_l|$ 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:

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

where

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

and

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

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, whereas ρ_{kl} is the number of 0s in the block. If $\lambda_{kl} \geq \rho_{kl}$, 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.

2.2. The Number of Feasible Solutions for Problem P1

The number of possible solutions to 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 prespecified image matrix and is, therefore, not directly suited for the problem at hand.

2.3. A Relocation Heuristic for Problem P1

Doreian et al. (2004) devised an effective heuristic algorithm for two-mode partitioning. The heuristic requires an initial feasible solution to problem P1 in the form of initial partitions, π_K and ω_L, for the row and column objects, respectively. These initial partitions are refined by considering all possible relocations of row and columns objects from their current cluster to one of the other clusters, as well as all possible exchanges of objects in different clusters. Transfers or exchanges that improve the objective function (Eq. 1) are implemented to refine partitions π_K and/or ω_L. The relocation heuristic terminates when there is no transfer or exchange that will further reduce the number of inconsistencies. Accordingly, the resulting solution is locally optimal with respect to all possible transfers and exchanges, but it is not guaranteed to be globally optimal. To avoid the potential for a poor local optimum, Doreian et al. (2004) recommend using a large number of repetitions of the heuristic using different initial partitions as input.

Although the relocation heuristic does not *guarantee* a globally optimal solution to problem P1, there is evidence in other blockmodeling contexts that it generally obtains the globally optimal partition on at least some of its repetitions (see, e.g., Brusco et al., 2011). The heuristic also possesses other advantages such as computational efficiency and flexibility. In addition, the relocation heuristic is an important ally for the branch-and-bound algorithm presented in the next section because it can be used to provide strong initial bounds that enhance the computational speed of the branch-and-bound process.

3. A BRANCH-AND-BOUND ALGORITHM FOR P1

3.1. Preliminary Processing

Effective branch-and-bound algorithms have been developed for a variety of single-mode partitioning problems (Brusco, 2006; Brusco & Cradit, 2004; Brusco et al., 2011; Brusco & Steinley, 2010; Hansen & Delattre, 1978; Klein & Aronson, 1991; Koontz et al., 1975; Palubeckis, 1997). In many instances, the performance of these algorithms can be improved by applying heuristic procedures to obtain good initial bounds (Brusco, 2006; Brusco & Cradit, 2004). The branch-and-bound algorithm that we developed for P1 is similar in structure to these one-mode algorithms, and also benefits from the use of information from a heuristic procedure in a preliminary step. Therefore, we recommend obtaining an upper bound for Z using repetitions of the relocation heuristic described by Doreian et al. (2005). Each repetition uses a different, randomly generated, initial two-mode partition. Adding one to the best objective function value obtained across the repetitions provides an initial upper bound, Z_{UB}. The reason that we add one is to ensure that the branch-and-bound algorithm will actually find the optimal solution if the relocation heuristic also obtained the optimum.

3.2. Steps of the Algorithm

Our description of the steps of the branch-and-bound algorithm for P1 employs terminology consistent with those of earlier presentations (Brusco & Stahl, 2005, Chapters 2–5; Brusco & Steinley, 2010; Klein & Aronson, 1991). A complete solution is one where all row and column objects have been assigned to a cluster, whereas as a partial solution corresponds to a situation where not all objects have been assigned. For a given partial solution, we can compute a lower bound on the best possible criterion function that could be achieved by completing the partial solution. If this lower bound equals or exceeds the current upper bound (initially obtained by the relocation heuristic), then the partial solution (and, accordingly, all complete solutions that could be obtained from the partial solution) can be pruned or eliminated from consideration. The following additional notation facilitates the description of the algorithm:

s: the object position pointer, which ranges from $1 \le s \le N + M$;
v: represents the cluster to which object s is being considered for assignment;
p: the position pointer for row objects, which ranges from $1 \le p \le N$;
q: the position pointer for column objects, which ranges from $1 \le q \le N$;
$\boldsymbol{\eta}$: an $(N + M) \times 1$ indicator vector with elements $\eta(s) = 0$ if object s is a row object and $\eta(s) = 1$ if object s is a column object for $1 \le s \le N + M$;
ϕ_r: the number of row clusters that are currently empty;
ϕ_c: the number of column clusters that are currently empty;
Z: the value of the objective criterion function (1) for the current partial solution. Note that $g(\pi_K, \omega_L)$ is defined for complete partitions π_K and ω_L, whereas Z is defined for conditions where these partitions are not necessarily complete. Accordingly, when the branch-and-bound algorithm generates a complete partition, then $Z = g(\pi_K, \omega_L)$;
$Z2$: a bound representing the minimum possible contribution to the objective function that can be realized from the unassigned column objects;
$Z3$: a bound representing the minimum possible contribution to the objective function that can be realized from the unassigned row objects.

The steps of the branch-and-bound algorithm are as follows:

Step 0. INITIALIZATION. Select the number of clusters for the row and column objects, K and L. Apply the relocation heuristic to obtain Z_{UB}. Set $\phi_r = K$, $\phi_c = L$, $s = 0$, $p = 0$, $q = 0$, $\lambda_{kl} = 0$ for $1 \le k \le K$ and $1 \le l \le L$, $\rho_{kl} = 0$ for $1 \le k \le K$ and $1 \le l \le L$.
Step 1. BRANCH FORWARD. Set $v = 1$ and $s = s + 1$. If $[(p \le q \vee p < N) \wedge (q = M)]$, then go to Step 2; otherwise, go to Step 3.
Step 2. ASSIGN A ROW OBJECT
Step 2a. Set $\eta(s) = 0$, $p = p + 1$, and $R_v = R_v \cup \{p\}$. If $|R_v| = 1$, then set $\phi_r = \phi_r - 1$.
Step 2b. Update the λ_{kl} and ρ_{kl} values as follows:

$$\lambda_{vl} = \lambda_{vl} + \sum_{j \in C_1} a_{pj} \qquad \forall 1 \le l \le L,$$
$$\rho_{vl} = \rho_{vl} + \sum_{j \in C_1} (1 - a_{pj}) \quad \forall 1 \le l \le L.$$

Step 2c. If $N - p < \phi_r$, then go to Step 6; otherwise go to Step 4.
Step 3. ASSIGN A COLUMN OBJECT
Step 3a. Set $\eta(s) = 1$, $q = q + 1$ and $C_v = C_v \cup \{q\}$. If $|C_v| = 1$, then set $\phi_c = \phi_c - 1$.

Step 3b. Update the λ_{kl} and ρ_{kl} values as follows:

$$\begin{aligned} \lambda_{kv} &= \lambda_{kv} + \sum_{i \in R_k} a_{iq} && \forall 1 \leq k \leq K, \\ \rho_{kv} &= \rho_{kv} + \sum_{i \in R_k} (1 - a_{iq}) && \forall 1 \leq k \leq K. \end{aligned}$$

Step 3c. If $M - q < \phi_c$, then go to Step 6; otherwise go to Step 4.
Step 4. BOUND TEST.
Step 4a. Compute the objective criterion value for the current partial solution:

$$Z = \sum_{k=1}^{K} \sum_{l=1}^{L} \min\{\lambda_{kl}, \rho_{kl}\}.$$

Step 4b. Compute a bound on the contribution from unassigned column objects:

$$Z2 = \sum_{j=q+1}^{M} \sum_{k=1}^{K} \min\{ZOC(j,k),\ ZZC(j,k)\},$$

where

$$\begin{aligned} ZOC(j,k) &= \sum_{i \in R_k} a_{ij} && \forall q+1 \leq j \leq M \text{ and } 1 \leq k \leq K; \\ ZZC(j,k) &= \sum_{i \in R_k} (1 - a_{ij}) && \forall q+1 \leq j \leq M \text{ and } 1 \leq k \leq K. \end{aligned}$$

Step 4c. Compute a bound on the contribution from unassigned row objects:

$$Z3 = \sum_{i=p+1}^{N} \sum_{l=1}^{L} \min\{ZOR(i,l),\ ZZR(i,l)\},$$

where

$$\begin{aligned} ZOR(i,l) &= \sum_{j \in C_l} a_{ij} && \forall p+1 \leq i \leq N \text{ and } 1 \leq l \leq L; \\ ZZR(i,l) &= \sum_{j \in C_l} (1 - a_{ij}) && \forall p+1 \leq i \leq N \text{ and } 1 \leq l \leq L. \end{aligned}$$

Step 4d. If $Z + Z2 + Z3 \geq Z_{\mathrm{UB}}$, then go to Step 6; otherwise go to Step 5.
Step 5. UPDATE INCUMBENT? If ($p = N$ and $q = M$), then set $Z_{\mathrm{UB}} = Z$, $\pi_K = \{R_1, R_2, \ldots, R_K\}$, $\omega_L = \{C_1, C_2, \ldots, C_L\}$, and proceed to Step 6; otherwise go to Step 1.
Step 6. EVALUATION. If $[(\eta(s) = 0 \vee v = K) \wedge (\eta(s) = 1 \vee v = L) \wedge (\eta(s) = 0 \vee |R_v| = 1 \vee |R_{v+1}| = 0) \wedge (\eta(s) = 1 \vee |C_v| = 1 \vee |C_{v+1}| = 0)]$, then proceed to Step 9; otherwise, proceed to Step 7.

Step 7. BRANCH RIGHT ON ROW OBJECT. If $\eta(s) = 1$, then go to Step 8; otherwise, proceed with Step 7a.

Step 7a. Remove contribution of the assignment of row object p to cluster v:

$$\begin{aligned}\lambda_{vl} &= \lambda_{vl} - \sum_{j \in C_l} a_{pj} && \forall 1 \le l \le L,\\ \rho_{vl} &= \rho_{vl} - \sum_{j \in C_l} (1 - a_{pj}) && \forall 1 \le l \le L.\end{aligned}$$

Step 7b. Set $R_v = R_v - \{p\}$. If $|R_v| = 0$, then set $\phi_r = \phi_r + 1$.

Step 7c. Set $v = v + 1$, $R_v = R_v \cup \{p\}$. If $|R_v| = 1$, then set $\phi_r = \phi_r - 1$.

Step 7d. Add contribution of the assignment of row object p to cluster v:

$$\begin{aligned}\lambda_{vl} &= \lambda_{vl} + \sum_{j \in C_l} a_{pj} && \forall 1 \le l \le L,\\ \rho_{vl} &= \rho_{vl} + \sum_{j \in C_l} (1 - a_{pj}) && \forall 1 \le l \le L.\end{aligned}$$

Step 7e. If $N - p < \phi_r$, then go to Step 6; otherwise, go to Step 4.

Step 8. BRANCH RIGHT ON COLUMN OBJECT.

Step 8a. Remove contribution of the assignment of column object q to cluster v:

$$\begin{aligned}\lambda_{kv} &= \lambda_{kv} - \sum_{i \in R_k} a_{iq} && \forall 1 \le k \le K,\\ \rho_{kv} &= \rho_{kv} - \sum_{i \in R_k} (1 - a_{iq}) && \forall 1 \le k \le K.\end{aligned}$$

Step 8b. Set $C_v = C_v - \{q\}$. If $|C_v| = 0$, then set $\phi_c = \phi_c + 1$.

Step 8c. Set $v = v + 1$, $C_v = C_v \cup \{q\}$. If $|C_v| = 1$, then set $\phi_c = \phi_c - 1$.

Step 8d. Add contribution of the assignment of row object p to cluster v:

$$\begin{aligned}\lambda_{kv} &= \lambda_{kv} + \sum_{i \in R_k} a_{iq} && \forall 1 \le k \le K,\\ \rho_{kv} &= \rho_{kv} + \sum_{i \in R_k} (1 - a_{iq}) && \forall 1 \le k \le K.\end{aligned}$$

Step 8e. If $M - q < \phi_c$, then go to Step 6; otherwise, go to Step 4.

Step 9. DEPTH RETRACTION ON ROW OBJECT. If $\eta(s) = 1$, then go to Step 10; otherwise, proceed with Step 9a.

Step 9a. Remove contribution of the assignment of row object p to cluster v:

$$\begin{aligned}\lambda_{vl} &= \lambda_{vl} - \sum_{j \in C_l} a_{pj} && \forall 1 \le l \le L,\\ \rho_{vl} &= \rho_{vl} - \sum_{j \in C_l} (1 - a_{pj}) && \forall 1 \le l \le L.\end{aligned}$$

Step 9b. Set $R_v = R_v - \{p\}$. If $|R_v| = 0$, then set $\phi_r = \phi_r + 1$.

Step 9c. Set $s = s - 1$ and $p = p - 1$. If $s = 0$, then return Z_{UB}, π_K, ω_L, and STOP; otherwise, proceed to Step 9d.

Step 9d. If $\eta(s) = 0$, then set $v = v' : p \in R_{v'}$; otherwise, set $v = v' : q \in C_{v'}$. Return to Step 6.

Step 10. DEPTH RETRACTION ON COLUMN OBJECT.

Step 10a. Remove contribution of the assignment of column object q to cluster v:

$$\lambda_{kv} = \lambda_{kv} - \sum_{i \in R_k} a_{iq} \qquad \forall 1 \leq k \leq K,$$

$$\rho_{kv} = \rho_{kv} - \sum_{i \in R_k} (1 - a_{iq}) \quad \forall 1 \leq k \leq K.$$

Step 10b. Set $C_v = C_v - \{q\}$. If $|C_v| = 0$, then set $\phi_c = \phi_c + 1$.

Step 10c. Set $s = s - 1$ and $q = q - 1$. If $s = 0$, then return Z_{UB}, π_K, ω_L, and STOP; otherwise, proceed to Step 10d.

Step 10d. If $\eta(s) = 0$, then set $v = v' : p \in R_{v'}$; otherwise, set $v = v' : q \in C_{v'}$. Return to Step 6.

Step 0 of the branch-and-bound algorithm obtains the initial upper bound (Z_{UB}) on the criterion function using the relocation heuristic. A variety of parameters are also initialized at Step 0, including the object position pointer (s), the row-object pointer (p), the column-object pointer (q), the number of empty row and column clusters (ϕ_r and ϕ_c, respectively), and arrays containing the number of 1s (λ_{kl}) and 0s (ρ_{kl}) in each block. This initialization process highlights the need to maintain cluster information for two distinct sets of objects (i.e., the row objects and the column objects). Step 1 of the branch-and-bound algorithm advances the object pointer, s, and makes the determination of whether a cluster membership should be assigned for a row or column object. We designed the algorithm to alternate between assignments of row and column to build good bounds quickly. In some instances, the number of row objects exceeds the number of column objects or vice versa, and this is accommodated by the processing logic. For example, if there are $N = 18$ row objects and $M = 14$ columns objects, then the first 28 assignments will alternate between row and column objects, whereas the last four assignments correspond to the remaining four row objects.

Step 2 augments the contribution to the number of 0s and 1s in blocks that is realized from the assignment of a row object. Step 3 performs a similar function for the assignment of a column object. Moreover, Steps 2c and 3c result in the pruning of partial solutions if there are not a sufficient number of row objects or column objects, respectively, to fill the remaining empty clusters. Step 4 is the key stage of the branch-and-bound algorithm. Part of the lower bound is obtained directly from Z, which represents the contribution to the criterion function realized among assigned objects. The second component of the bound, $Z2$, is computed based on the minimum possible contribution of the $(M - q)$ unassigned column objects and their relationships with the assigned row objects. Similarly, the third component of the bound, $Z3$, is computed based on the minimum possible contribution of the $(N - p)$ unassigned row objects and their relationships with the assigned column objects. To illustrate, suppose we consider a yet unassigned row object, i, and we know that six column objects have been assigned to column cluster l. Further, assume that row object i has ties to exactly four of the six column objects. Given this information, we know that no matter what row cluster object i is assigned, we will pick up at least two inconsistencies based on the entries in row i and column cluster l, and this improves our bound on the best possible criterion function value that can be realized from the completion of the current partial solution.

If $p = N$ and $q = M$ at Step 5, then a new incumbent (best-found) solution is stored; otherwise, processing returns to Step 1 for assignment of the next object. Step 6 determines whether branching or depth retraction should occur. If $v = K$ (or $v = L$ in the case of a column object), then control is passed to depth retraction because all cluster placements for the current object have been exhausted. Depth retraction also occurs if $v < K$ (or $v < L$ in the case of a column object) and the current object is the only remaining object in cluster v and cluster $v + 1$ is empty. In all other cases, the object is moved to the cluster on the right, from v to $v + 1$, hence the term "branch right." Steps 7 and 8 perform the branch right operations for row objects and column objects, respectively. For example, processing at Step 7 removes the contribution to Z stemming from the assignment of row object p to cluster v (this is also required for retraction in Step 9) and then adds the contribution to Z stemming from the assignment of p to cluster $v + 1$. The algorithm terminates in Steps 9 or 10 when the pointer index, s, retracts to 0.

3.3. A Modified Algorithm for Finding Equally Well-Fitting Partitions

The branch-and-bound algorithm described in Subsection 3.2 will find a single set of partitions of the row and column objects that will provide a global minimum for the criterion function. However, it is not uncommon for there to be multiple sets of partitions that will produce the same global minimum for the criterion function. These are known as equally well-fitting partitions. Following Brusco and Steinley (2010) and Brusco et al. (2011), we developed a modified version of the branch-and-bound algorithm that will find all of the equally well-fitting partitions, provided there are 2,000 or fewer of these partitions. The limit of 2,000 was selected to limit the computer memory needed to store the equally well-fitting partitions. This is not a major limitation because most blockmodeling solutions that are apt to be of interest have far fewer than 2,000 sets of equally well-fitting partitions. The principal modification is to change the "$\geq$" sign in Step 4d to "$>$" so that partial solutions will not be pruned if they are capable for providing an optimal solution equal to the current upper bound, as such solutions would need to be stored in the set of equally well-fitting blockmodels.

4. SIMULATION STUDY

4.1. Experimental Test Problems

In this section, we explore the performance of the proposed branch-and-bound algorithm for a set of synthetic data sets. We conducted an experiment that manipulated four data features. The first feature corresponded to pairs representing the number of row and column objects (N, M) and was tested at three levels: ($N = 12$, $M = 12$), ($N = 18$, $M = 12$), and ($N = 18$, $M = 18$). The second feature was associated with pairs for the number of clusters (K, L) and was also manipulated at three levels: ($K = 2$, $L = 2$), ($K = 4$, $L = 2$), and ($K = 4$, $L = 4$). The third feature controlled the relative cluster sizes for the row and column objects and was tested at two levels: approximately equal and skewed. For example, if $N=18$ and $K=4$, then the approximately equal setting would specify row-object cluster sizes of 5, 5, 4, and 4. Contrastingly, the skewed setting for $N=18$ and $K=4$ would specify row-object cluster sizes of 9, 3, 3, and 3. The fourth feature is cluster density, which is controlled by the parameter δ. More specifically, complete blocks are generated with a density of δ and null blocks are generated with a density of $1-\delta$. The two settings for this feature were $\delta=.9$ and $\delta=.8$, such that the former setting produces a stronger distinction between complete and null blocks.

4.2. Implementation Issues

The branch-and-bound algorithm and relocation heuristic for two-mode blockmodeling were written in Fortran 90 and implemented on a 2.4 GHz Core 2 Duo Processor with 3 GB of SDRAM. The relocation algorithm was applied to each of the 108 test problems with a maximum time limit of 10 s. That is, a repetition of the heuristic was performed as long as 10 s had not yet elapsed. For each test problem, the data collected for the relocation heuristic included the number of repetitions realized within the 10-s limit and the number of times that the heuristic obtained the globally optimal criterion value. Dividing the latter measure by the former, we computed the attraction rate, which is the percentage of repetitions for which the optimum was obtained (see Brusco et al., 2011). The branch-and-bound algorithm was also applied to each of the 108 test problems, and the total computation time required to confirm the globally optimal partition was recorded.

4.3. Simulation Results

Table 1 presents a summary of the results of the simulation experiment across feature levels and overall. A more detailed presentation of the results by test problem

TABLE 1 Simulation Results: A Summary for the Levels of Each Feature

	Relocation heuristic		Branch-and-bound
Feature levels	Mean # of restarts	Mean attraction rate	CPU time
$N=12$ and $M=12$	61048	51.03	10.16
$N=18$ and $M=12$	29687	47.76	13.99
$N=18$ and $M=18$	16434	44.33	294.99
$K=3$ and $L=3$	53245	81.84	10.02
$K=6$ and $L=3$	37910	33.30	29.95
$K=6$ and $L=6$	16015	27.97	279.16
Equal spread	31074	67.07	83.49
Skewed spread	40373	28.34	129.26
$\delta=90\%$	36957	56.05	10.30
$\delta=80\%$	34489	39.36	202.46
Overall averages	35723	47.71	106.38

is provided in the Appendix. A summary of the results contained in Table 1 and Appendix follows:

1. The relocation heuristic achieved a large number of repetitions within the modest time limit of 10 s, ranging from roughly 7,000 to 100,000, depending on the problem characteristics (especially, *N, M, K*, and *L*).
2. The relocation heuristic obtained a globally optimal blockmodel for *all* 108 test problems. This finding is consistent with the results of Brusco et al. (2011) within the context of one-mode signed networks, and demonstrates the efficacy of the method.
3. The attraction rate for the relocation heuristic varied drastically across the test problems, from a minimum of 0.22% to a maximum of 97.75%. In general, the attraction rate decreased as the number of row and column objects increased, the number of row and column clusters increased, the spread of objects across clusters became skewed, and as δ decreased.
4. The branch-and-bound algorithm often obtained a confirmed globally optimal blockmodel with only modest computational effort (95 of the 108 test problem were solved exactly within 30 s). However, five problems required more than 10 min and two of these required 45–60 min.
5. Although the problem characteristics that reduce the attraction rate for the relocation heuristic are consistent with those that increase the computation time for the branch-and-bound algorithm, the relationship is not perfectly concordant. For example, for the test problem requiring the greatest branch-and-bound computation time (3526.76 s), the attraction rate for the relocation heuristic was 2.47%. For the test problem with the smallest attraction rate for the relocation heuristic (.22%), the branch-and-bound computation time was 534.04 s.

5. EXAMPLE 1: CEO/CLUB AFFILIATION NETWORK

For our first example, we consider a 26×15 two-mode affiliation network based on data originally collected by Galaskiewicz (1985) as part of a comprehensive study of Minneapolis-St.Paul's urban grants economy. The ties in the network correspond to the participation of $N = 26$ chief-executive-officers (CEOs) in $M = 15$ clubs/boards. Faust (1997) conducted one-mode centrality analyses of the data for CEOs and clubs independently, as well as a two-mode centrality analysis of the bipartite graph. She observed that, although the one-mode centrality analysis for the clubs was reasonably concordant with the two-mode centrality results, the one-mode analysis for the CEOs exhibited marked differences from the two-mode results. In our investigation, we consider the use of two-mode blockmodeling to uncover relationships between the two distinct sets of objects (i.e., CEOs and clubs).

We began our two-mode blockmodeling analysis of the CEO/club affiliation network by fitting blockmodels for all combinations of K and L on the intervals $2 \leq K \leq 5$ and $2 \leq L \leq 6$. For each pair of values for K and L, we used a 10-s implementation of the relocation algorithm, followed by the branch-and-bound algorithm to confirm the global optimality of the loss function value. We then applied the modified branch-and-bound algorithm to identify all of the equally well-fitting partitions. A summary of the results is provided in Table 2.

The exceptional performance of the relocation heuristic in the simulation experiment was reinforced by our results for the affiliation network. The relocation heuristic always obtained the optimal criterion function value within the 10-s time limit. The time required by the branch-and-bound algorithm to confirm global optimality varied markedly as a function of K and L. For $K = L = 2$, global optimality was confirmed in 0.17 s, and only 1.46 and 7.08 s were required for the (2, 3) and (2, 4) blockmodels, respectively. The maximum computation time was 17700.06 s for the (3, 6) blockmodel. Another salient observation from Table 2 is the fact that computation time was generally modest in the instances where there was a unique global optimum (e.g., 473.66 s for the (5, 5) blockmodel) but was frequently excessive when there were many equally well-fitting partitions (e.g., 13705.37 s for the (4, 6) blockmodel).

It was particularly interesting to discover that the optimal (2, 2), (2, 3), and (2, 4) blockmodels were all unique, with 73, 66, and 62 inconsistencies, respectively. There were 197 equally well-fitting partitions for the (3, 3) blockmodel and the corresponding 64 inconsistencies was worse than the corresponding number of 62 for the (2, 4) blockmodel. Moving to a (3, 4) blockmodel lowered the number of inconsistencies to 58, but there were still 67 equally well-fitting partitions. The (5, 5) blockmodel stood out as a promising choice for further analysis because of several factors: (1) the 47 inconsistencies for this blockmodel was a notable improvement over the 52 and 51 inconsistencies realized for the (5, 4) and (4, 5) blockmodels, respectively, (2) increasing the number of clusters for clubs to $L = 6$ only reduced the number of inconsistencies by 1, and (3) the optimal (5, 5) blockmodel was unique. The unique

TABLE 2 Results for Minneapolis/St. Paul CEO/Club Affiliation Network

Row clusters		Column clusters				
		$L = 2$	$L = 3$	$L = 4$	$L = 5$	$L = 6$
$K = 2$	Inconsistencies	73	66	62	62	62
	EWFP	1	1	1	1024	2000+
	Time	.17	1.46	7.08	159.85	3365.38
$K = 3$	Inconsistencies	73	64	58	56	56
	EWFP	2000+	197	67	2000+	2000+
	Time	3.22	19.50	65.53	698.04	17700.06
$K = 4$	Inconsistencies	73	61	54	51	51
	EWFP	2000+	1	65	3	2000+
	Time	26.12	60.73	133.06	607.34	13705.37
$K = 5$	Inconsistencies	73	61	52	47	46
	EWFP	2000+	776	130	1	3
	Time	125.52	188.11	265.56	473.66	3891.60

Note. Inconsistencies is the optimal value of the criterion function. EWFP is the number of equally well-fitting partitions. Time is the total computation time for the branch-and-bound algorithm in seconds for a 2.4 GHz Core 2 Duo processor.

globally optimal (5, 5) blockmodel, which has 47 inconsistencies, is displayed in Figure 1. The image matrix is:

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

The most striking feature of the (5, 5) blockmodel in Figure 1 is that there are three singleton clusters in the partition of the clubs. Every single CEO in each of the first four CEO clusters is affiliated with club "c3," which comprises the first cluster of

	c3	c4	c2	c11	c12	c15	c1	c5	c6	c7	c8	c9	c10	c13	c14
e14	1	1	1	1	1	1	0	0	0	0	0	0	0	**1**	0
e17	1	**0**	1	1	1	1	0	**1**	0	0	0	0	0	0	0
e20	1	1	1	1	**0**	1	0	0	0	0	0	0	0	0	0
e1	1	1	0	0	0	0	0	0	0	0	0	**1**	0	0	0
e7	1	1	0	**1**	0	0	0	0	0	0	0	0	**1**	0	0
e13	1	1	0	0	0	0	0	**1**	0	0	0	**1**	0	0	0
e19	1	1	0	0	0	0	1	0	0	**1**	0	**1**	0	0	0
e21	1	1	0	0	0	0	0	0	0	0	**1**	0	0	0	0
e24	1	1	0	0	0	**1**	**1**	0	**1**	0	0	0	0	0	0
e4	1	0	1	0	0	**1**	0	0	0	0	0	0	0	0	0
e6	1	0	1	0	0	0	0	0	0	0	0	0	0	0	**1**
e11	1	0	1	0	0	0	0	0	0	0	0	**1**	0	0	0
e15	1	0	1	0	0	**1**	0	0	**1**	0	0	0	0	**1**	0
e16	1	0	1	0	0	0	0	0	**1**	0	**1**	0	0	0	**1**
p23	1	0	1	0	0	**1**	0	0	**1**	0	0	0	0	0	0
e25	1	0	1	0	0	0	0	0	0	0	0	0	0	**1**	0
e26	1	0	1	0	**1**	0	0	0	0	0	0	0	0	0	0
e2	1	0	0	0	0	0	0	**1**	0	**1**	0	0	0	0	0
e3	1	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0
e5	1	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**
e10	1	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0
e22	1	0	0	0	0	**1**	0	0	0	0	**1**	0	0	0	0
e8	0	1	0	0	0	0	0	0	0	**1**	0	0	**1**	0	0
e9	0	1	0	0	0	0	**1**	0	0	0	**1**	0	**1**	0	0
e12	0	1	0	0	0	0	0	0	0	**1**	0	0	0	0	0
e18	0	1	0	0	**1**	**1**	0	0	0	0	0	**1**	0	**1**	0

FIGURE 1 The unique globally optimal ($K = 5$, $L = 5$) blockmodel for the CEO/club affiliation network. The row objects are the executives (labeled "e1" to "e26") and the column objects are the clubs labeled ("c1" to "c15"). The horizontal lines delineate the row partition and the vertical lines delineate the column partition. The inconsistencies are shown in bold type.

clubs in Figure 1. Club "c3" is, by far, the most popular club, with 22 of the 26 executives as participants. Club "c4" is the second most popular club, with its 12 participating executives including all six members of CEO cluster 2, all four members of CEO cluster 5, and two of the three members of CEO cluster 1. Club "c3" is close behind with its 11 participating executives including all eight members of CEO cluster 3 and all three members of CEO cluster 1.

The first cluster of CEOs consists of three executives {"e14," "e17," "e20") who are strongly affiliated with the six clubs spanning the first four clusters of clubs {"c3," "c4," "c2," "c11," "c12," "c15"). Moreover, these three executives have only two ties ("e14" with "c13" and "e17" with "c5") to the nine clubs in the fifth cluster of clubs. All six executives in the second cluster of CEOs {"e1," "e7," "e13," "e19," "e21," "e24") are tied to clubs "c3" and "c4," with only sporadic ties to other clubs. Similarly, all eight executives in the third cluster of CEOs {"e4," "e6," "e11," "e15," "e16," "e23," "e25," "e26") are tied to clubs "c3" and "c2," with only sporadic ties to other clubs. The five executives in the fourth cluster of CEOs {"e2," "e3," "e5," "e10," "e22") are each tied to club "c3" but only sporadically with the other 14 clubs. Finally, the four executives in the fifth cluster of CEOs {"e8," "e9," "e12," "e18") are each tied to club "c4" but only sporadically with the other 14 clubs. In summary, the (5, 5) blockmodel provides a revealing picture of the patterning of ties between groups of executives and groups of clubs.

6. EXAMPLE 2: KANSAS SAR COMMUNICATION NETWORK

The next example we consider is actually a one-mode network converted to a two-mode network. The reason is simple: If there are social relations that are inherently nonsymmetric, then there is no reason to expect that the rows and columns are best partitioned in the same way. Specifically, we consider the Kansas SAR communication network data originally collected by Drabek et al. (1981) and subsequently analyzed by Doreian et al. (2005, Chapter 6) and Brusco and Steinley (2009). As with these previous applications, the data are studied in the form of a

TABLE 3 Results for Kansas SAR Data

Row clusters		Column clusters				
		$L=2$	$L=3$	$L=4$	$L=5$	$L=6$
$K=2$	Inconsistencies	76	64	64	—	—
	EWFP	3	6	2000+	—	—
	Time	2.68	13.47	268.18	—	—
$K=3$	Inconsistencies	69	57	49	49	—
	EWFP	1	1	6	755	—
	Time	34.06	101.37	134.14	1102.61	—
$K=4$	Inconsistencies	69	57	49	44	42
	EWFP	2000+	1087	2000+	29	23
	Time	486.16	1591.24	2439.11	3192.78	6845.70

Note. Inconsistencies is the optimal value of the criterion function. EWFP is the number of equally well-fitting partitions. Time is the total computation time for the branch-and-bound algorithm in seconds for a 2.4 GHz Core 2 Duo processor.

binary, asymmetric, one-mode matrix with elements of 1 reflecting communication from one agency to another. However, unlike these previous studies, we apply a two-mode perspective to these inherently one-mode data. More specifically, we consider the rows as message-sending agencies and the columns as message-receiving agencies. Thus, although the rows and columns of the network correspond to the same set of objects, our analysis allows for the possibility of the classification of the sending agencies is not necessarily the appropriate classification for the receiving agencies. In light of the fact that communication of an agency with itself is nonsensical, the main diagonal is ignored in our analyses.

We began our analysis of the Kansas SAR network by considering a ($K=2$, $L=2$) blockmodel and gradually increasing the number of clusters for the row and columns objects. For each pair of values for K and L, we began with a 10-s implementation of the relocation algorithm, followed by the branch-and-bound algorithm to confirm the global optimality of the loss function value. We then

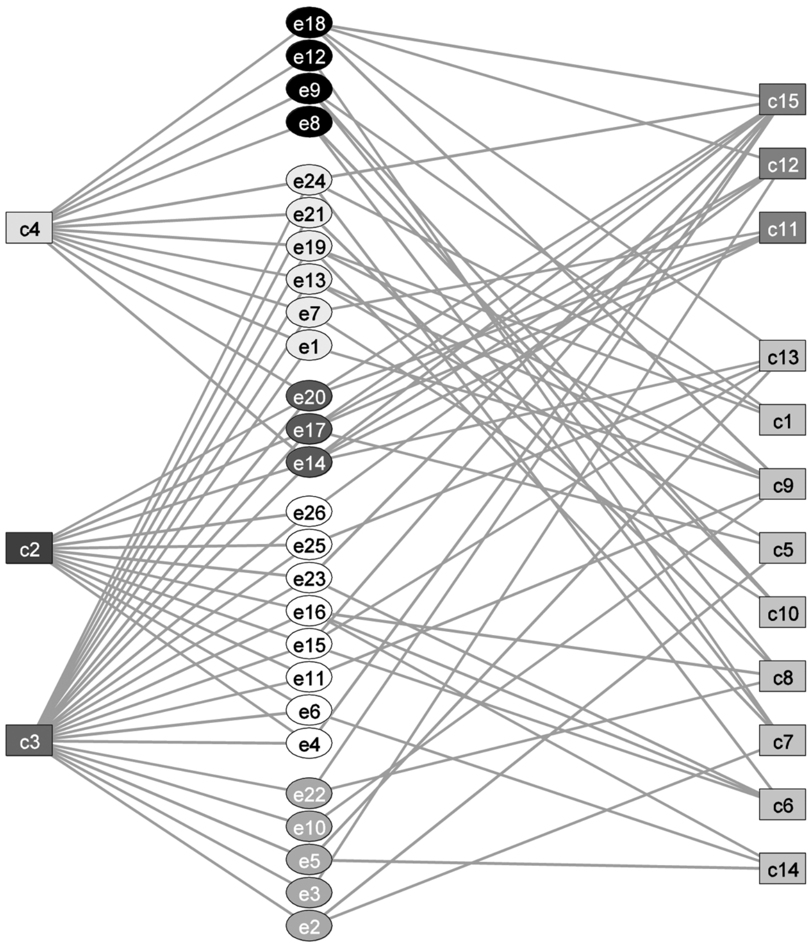

FIGURE 2 The clubs and CEO two-mode network with partitions of clubs and CEOs.

applied the modified branch-and-bound algorithm to identify all of the equally well-fitting partitions. A summary of the results is provided in Table 3.

As was the case for the simulation experiment, the relocation heuristic always obtained the optimal criterion function value within the 10-s time limit. The time required by the branch-and-bound algorithm to confirm global optimality varied markedly as a function of K and L. For $K = L = 2$, global optimality was confirmed in 2.68 s, and only 101.37 s was required for the (3, 3) blockmodel. Computation time increased markedly to 2439.11 s for the (4, 4) blockmodel and was 6845.70 s for the (4, 6) blockmodel. Although this computation time is substantial, it should be noted that the number of feasible partitions for the (4, 6) blockmodel is $SN(20, 4) \times SN(20, 6) \approx 1.948 \times 10^{23}$. Accordingly, it is impressive that global optimality can be confirmed at all.

The (3, 3) blockmodel was selected for further analysis based on several factors: (a) the marked improvement in the criterion function relative to the (2, 3) and (3, 2) blockmodeling solutions, (b) our discovery that the optimal (3, 3)

		3	6	7	9	17	1	5	2	4	8	10	11	12	13	14	15	16	18	19	20
1	OSheriff	1	1	1	1	1	x	1	**1**	0	0	0	**1**	0	0	0	**1**	**1**	0	0	**1**
2	Civil Def	1	1	1	1	1	1	1	x	0	0	0	0	0	0	**1**	0	**1**	0	0	0
3	Coroner	x	1	**0**	1	1	1	1	0	**1**	0	0	**1**	0	0	0	0	0	0	0	0
4	Attorney	1	1	1	1	**0**	1	1	**1**	x	0	0	0	0	0	**1**	0	0	0	0	0
6	Parks n Res	1	x	1	1	**0**	1	1	0	**1**	0	0	**1**	**1**	0	**1**	0	0	0	0	0
7	Game n Fish	1	1	x	1	**0**	1	1	0	**1**	0	**1**	0	0	0	0	0	**1**	0	0	0
9	Army Corps	1	1	1	x	1	1	1	0	0	0	0	**1**	0	0	0	0	0	0	0	0
11	Crable Amb	1	1	1	1	**0**	1	1	0	0	0	0	x	**1**	0	0	0	0	0	0	0
14	Shawney	1	1	1	1	1	1	1	0	0	0	0	0	**1**	**1**	x	0	0	**1**	0	**1**
17	Red Cross	1	1	1	1	x	1	1	**1**	0	0	**1**	0	0	0	**1**	0	0	0	0	0
19	Carb FD	**0**	1	1	1	1	1	1	**1**	0	0	0	0	0	0	0	0	0	0	x	0
5	Highway P	1	1	1	**0**	1	1	x	**0**	1	**0**	**0**	1	1	1	1	**0**	1	**0**	**0**	1
15	Burl Police	1	1	1	1	1	1	1	**0**	1	**0**	1	1	1	1	1	x	1	1	**0**	1
8	Kansas DOT	0	0	0	0	0	1	1	**1**	0	x	**1**	0	0	0	0	0	0	0	0	0
10	Army Reserve	0	0	0	0	0	1	1	0	0	0	x	0	0	0	0	0	0	0	0	0
12	Frank Co. Amb	0	0	**1**	0	**1**	1	1	0	0	0	0	0	x	0	0	0	0	0	0	0
13	Lee Rescue	0	0	0	0	0	1	1	0	0	0	0	0	0	x	0	0	0	0	0	0
16	Lynd Police	0	**1**	**1**	0	0	1	1	**1**	0	0	0	**1**	0	0	0	0	x	0	0	0
18	Topeka FD	0	0	0	0	0	1	1	0	0	0	0	0	0	**1**	**1**	0	0	x	0	0
20	Topeka RBW	0	0	0	0	**1**	1	1	0	0	0	0	0	0	0	0	**1**	**1**	0	0	x

FIGURE 3 The unique globally optimal ($K = 3$, $L = 3$) blockmodel for the Kansas SAR data. The row objects are the sending agencies and the column objects are the receiving agencies. The horizontal lines delineate the row partition and the vertical lines delineate the column partition. The inconsistencies are shown in bold type and values of x are ignored elements corresponding the main diagonal of the original one-mode data.

blockmodel was unique, and (c) the interpretability of the blockmodel. The unique globally optimal (3, 3) blockmodel, which has 57 inconsistencies, is displayed in Figure 3. The image matrix is:

Complete	Complete	Null
Complete	Complete	Complete
Null	Complete	Null

The Highway Patrol is paired with the Burlingame Police in the second cluster of sending agencies. This cluster forms complete blocks with all three of the clusters of receiving agencies, which is indicative of the fact that the Highway Patrol and Burlingame Police each *sent communication to* most of the other agencies. In a similar manner, the Sheriff's Office and the Highway Patrol each *received communication from* all of the sending agencies. Therefore, it is not surprising that the Sheriff's Office and the Highway Patrol formed the second cluster of receiving agencies.

The key feature of the (3, 3) blockmodel that illustrates the value afforded by the two-mode perspective is that the agencies in the first message sending cluster are split across all three of the message-receiving clusters. The first cluster of message senders consists of 11 agencies. Five of these agencies ("Coroner," "Parks n Res," "Game n Fish," "Army Corps," "Red Cross") completely comprise message receiving cluster 1, one of the agencies ("OSherriff") is in message receiving cluster 2, and five of the agencies ("Civil Def," "Attorney," "Crable Amb," "Shawney," "Carb FD") are in message receiving cluster 3.

7. SUMMARY

Two-mode blockmodeling problems have an enormous number of possible solutions because of the need to establish partitions for two distinct sets of objects. For this reason, there has been a reliance on approximate (or heuristic) procedures for two-mode blockmodeling. We have presented a branch-and-bound algorithm for two-mode blockmodeling of binary matrices. A modified version of the algorithm for obtaining all equally well-fitting partitions (up to a maximum limit of 2,000) is also provided. We strongly recommend using the relocation heuristic developed by Doreian et al. (2005) prior to the implementation of the branch-and-bound algorithm to establish a good initial upper bound on the number of inconsistencies.

Like any branch-and-bound procedure for partitioning, our proposed method can gravitate toward total enumeration for problems that are not well structured. Moreover, such algorithms are plagued by a "curse of dimensionality" that makes them prohibitively costly as the number of clusters increases (Du Merle et al., 2000). It is possible that improved bounding procedures might mitigate these problems to some extent; however, they remain inherent for methods of this type. Nevertheless, we are encouraged by the fact that the proposed algorithm has successfully obtained optimal solutions for a number of modestly sized, real-world test problems from the social network literature. We hope that these encouraging results might ultimately lead to inclusion of branch-and-bound methods for blockmodeling in social network software packages such as Ucinet (Borgatti, Everett, & Freeman, 2002) and Pajek (Batagelj & Mrvar, 1998).

REFERENCES

Arabie, P., Boorman, S. A., & Levitt, P. R. (1978). Constructing blockmodels: How and why. *Journal of Mathematical Psychology*, *17*, 21–63.

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

Avella, P., Sassano, A., & Vasil'ev, I. (2007). Computational study of large-scale p-median problems. *Mathematical Programming A*, *109*, 89–114.

Batagelj, V., & Mrvar, A. (1998). Pajek—Program for large network analysis. *Connections*, *21*, 47–57.

Batagelj, V., Mrvar, A., Ferligoj, A., & Doreian, P. (2004). Generalized blockmodeling with Pajek. *Metodoloski Zvezki: Journal of the Statistical Society of Slovenia*, *1*, 455–467.

Borgatti, S. P., & Everett, M. G. (1992). Regular blockmodels of multiway, multimode matrices. *Social Networks*, *14*, 91–120.

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, MA: Analytic Technologies.

Brieger, R. L. (1974). The duality of persons and groups. *Social Forces*, *53*, 181–190.

Brieger, 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. J. (2003). An enhanced branch-and-bound algorithm for a partitioning problem. *British Journal of Mathematical and Statistical Psychology*, *56*, 83–92.

Brusco, M. J. (2006). A repetitive branch-and-bound procedure for minimum within-cluster sums of squares partitioning. *Psychometrika*, *71*, 347–363.

Brusco, M. J., & Cradit, J. D. (2004). Graph coloring, minimum-diameter partitioning, and the analysis of confusion matrices. *Journal of Mathematical Psychology*, *48*, 310–319.

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., & Stahl, S. (2005). *Branch-and-bound applications in combinatorial data analysis*. New York, NY: Springer.

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

Brusco, M., & Steinley, D. (2007). 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. (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 applications to generalized structural balance and other psychological contexts. *Psychological Methods*, *15*, 145–157.

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

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

Diehr, G. (1985). Evaluation of a branch and bound algorithm for clustering. *SIAM Journal for Scientific and Statistical Computing*, *6*, 268–284.

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, UK: Cambridge University Press.

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

Drabek, T. E., Tamminga, H. L., Kilijanek, T. S., & Adams, C. R. (1981). *Managing multi-organizational emergency responses*. Boulder, CO: University of Colorado, Institute of Behavioral Science.

Du Merle, O., Hansen, P., Jaumard, B., & Mladenović, N. (2000). An interior point algorithm for minimum sum-of-squares clustering. *SIAM Journal on Scientific Computing*, *21*, 1485–1505.

Faust, K. (1997). Centrality in affiliation networks. *Social Networks*, *19*, 157–191.

Galaskiewicz, J. (1985). *Social organization of an urban grants economy*. New York, NY: Academic Press.

Glover, F. (1989). Tabu search—Part I. *ORSA Journal on Computing*, *1*, 190–206.

Glover, F. (1990). Tabu search—Part II. *ORSA Journal on Computing*, *2*, 4–32.

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

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

Hansen, P., & Delattre, M. (1978). Complete-link cluster analysis by graph coloring. *Journal of the American Statistical Association*, *73*, 397–403.

Hansen, P., & Jaumard, B. (1997). Cluster analysis and mathematical programming. *Mathematical Programming*, *79*, 191–215.

Hubert, L. (1974). Problems of seriation using a subject by item response matrix. *Psychological Bulletin*, *81*, 976–983.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*, 671–680.

Klein, G., & Aronson, J. E. (1991). Optimal clustering: A model and method. *Naval Research Logistics*, *38*, 447–461.

Koontz, W. L. G., Narendra, P. M., & Fukunaga, K. (1975). A branch and bound clustering algorithm. *IEEE Transactions on Computing*, *C-24*, 908–915.

Latapy, M., Magnien, C., & Del Vecchio, N. (2008). Basic notions for the analysis of large two-mode networks. *Social Networks*, *30*, 31–48.

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

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam & J. Neyman (Eds.), *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281–297). Berkeley: University of California Press.

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.

Palubeckis, G. (1997). A branch-and-bound approach using polyhedral results for a clustering problem. *INFORMS Journal on Computing*, *9*, 30–42.

Sailer, L. D. (1978). Structural equivalence: Meaning and definition, computation and application. *Social Networks*, *1*, 73–90.

Steinhaus, H. (1956). Sur la division des corps matériels en parties. *Bulletin de l'Académie Polonaise des Sciences, Classe III Mathématique, Astronomie, Physique, Chimie, Géologie, et Géographie*, *4*, 801–804.

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

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

White, H. C., Boorman, S. A., & Breiger, R. L. (1976). Social structure from multiple networks. I. Blockmodels of roles and positions. *American Journal of Sociology*, *81*, 730–779.

APPENDIX: DETAILED RESULTS OF THE SIMULATION EXPERIMENT

Cell replicate	Feature levels				Criterion function	Relocation heuristic		B&B CPU time
	N, M	*K, L*	Spread	δ		# of restarts	Attraction rate	
1	12,12	3,3	Even	90%	17	94501	96.38	10.01
1	12,12	3,3	Even	80%	29	89471	73.21	10.01
1	12,12	3,3	Skewed	90%	17	90171	96.59	10.01
1	12,12	3,3	Skewed	80%	29	86023	70.30	10.01
1	12,12	6,3	Even	90%	17	46124	84.38	10.02
1	12,12	6,3	Even	80%	29	44340	57.26	10.12
1	12,12	6,3	Skewed	90%	17	80084	16.03	10.03
1	12,12	6,3	Skewed	80%	28	67321	7.83	10.07
1	12,12	6,6	Even	90%	17	24907	65.15	10.05
1	12,12	6,6	Even	80%	26	27270	14.89	10.89
1	12,12	6,6	Skewed	90%	17	29509	12.35	10.06
1	12,12	6,6	Skewed	80%	27	29347	8.96	11.66
1	18,12	3,3	Even	90%	24	44278	93.53	10.01
1	18,12	3,3	Even	80%	39	39525	89.12	10.01
1	18,12	3,3	Skewed	90%	24	41903	92.07	10.01
1	18,12	3,3	Skewed	80%	39	40756	72.58	10.01
1	18,12	6,3	Even	90%	24	21739	68.59	10.04
1	18,12	6,3	Even	80%	39	19205	71.62	10.54
1	18,12	6,3	Skewed	90%	24	44470	4.39	10.06
1	18,12	6,3	Skewed	80%	38	41633	1.26	10.55
1	18,12	6,6	Even	90%	24	10898	80.50	10.20
1	18,12	6,6	Even	80%	37	11157	53.05	19.90
1	18,12	6,6	Skewed	90%	24	14634	8.94	10.10
1	18,12	6,6	Skewed	80%	37	13505	3.10	17.77
1	18,18	3,3	Even	90%	39	21679	95.58	10.01
1	18,18	3,3	Even	80%	68	20543	75.02	10.06
1	18,18	3,3	Skewed	90%	39	25726	60.80	10.01
1	18,18	3,3	Skewed	80%	68	27087	41.65	10.09
1	18,18	6,3	Even	90%	39	13484	62.58	10.17

(*Continued*)

APPENDIX *(CONTINUED)*

Cell replicate	Feature levels: *N, M*	*K, L*	Spread	δ	Criterion function	Relocation heuristic: # of restarts	Attraction rate	B&B CPU time
1	18,18	6,3	Even	80%	67	11748	52.01	33.20
1	18,18	6,3	Skewed	90%	39	23464	2.19	10.44
1	18,18	6,3	Skewed	80%	64	23827	0.81	110.81
1	18,18	6,6	Even	90%	38	5611	77.74	10.79
1	18,18	6,6	Even	80%	65	6177	14.80	842.92
1	18,18	6,6	Skewed	90%	39	9364	1.22	13.09
1	18,18	6,6	Skewed	80%	62	7017	2.38	849.19
2	12,12	3,3	Even	90%	13	94369	97.75	10.01
2	12,12	3,3	Even	80%	28	86639	78.92	10.01
2	12,12	3,3	Skewed	90%	13	93843	95.53	10.01
2	12,12	3,3	Skewed	80%	28	84304	70.91	10.01
2	12,12	6,3	Even	90%	13	49556	74.89	10.01
2	12,12	6,3	Even	80%	28	49648	45.35	10.10
2	12,12	6,3	Skewed	90%	13	72044	15.46	10.01
2	12,12	6,3	Skewed	80%	25	61806	32.83	10.06
2	12,12	6,6	Even	90%	13	23079	82.59	10.01
2	12,12	6,6	Even	80%	24	27018	27.82	10.37
2	12,12	6,6	Skewed	90%	13	31042	8.79	10.03
2	12,12	6,6	Skewed	80%	27	30003	2.41	11.30
2	18,12	3,3	Even	90%	20	44987	97.44	10.01
2	18,12	3,3	Even	80%	45	37245	86.12	10.03
2	18,12	3,3	Skewed	90%	20	57608	68.46	10.01
2	18,12	3,3	Skewed	80%	45	61908	37.87	10.03
2	18,12	6,3	Even	90%	20	22588	65.51	10.01
2	18,12	6,3	Even	80%	45	18227	67.37	10.46
2	18,12	6,3	Skewed	90%	20	41759	4.92	10.03
2	18,12	6,3	Skewed	80%	44	29705	2.95	13.84
2	18,12	6,6	Even	90%	20	11017	80.73	10.05
2	18,12	6,6	Even	80%	45	13464	8.27	65.03
2	18,12	6,6	Skewed	90%	20	16291	7.12	10.07
2	18,12	6,6	Skewed	80%	42	13638	5.75	54.73
2	18,18	3,3	Even	90%	31	23158	94.49	10.01
2	18,18	3,3	Even	80%	60	20562	89.82	10.06
2	18,18	3,3	Skewed	90%	31	23409	85.96	10.01
2	18,18	3,3	Skewed	80%	60	17914	96.07	10.05
2	18,18	6,3	Even	90%	31	13588	68.17	10.08
2	18,18	6,3	Even	80%	60	11867	37.69	19.17
2	18,18	6,3	Skewed	90%	31	28608	2.66	10.21
2	18,18	6,3	Skewed	80%	60	25901	1.35	37.96
2	18,18	6,6	Even	90%	31	6246	70.93	11.04
2	18,18	6,6	Even	80%	57	5976	34.66	228.67
2	18,18	6,6	Skewed	90%	31	9072	4.06	11.46
2	18,18	6,6	Skewed	80%	60	7509	2.42	1349.34
3	12,12	3,3	Even	90%	14	96717	96.43	10.01
3	12,12	3,3	Even	80%	26	87774	84.56	10.01
3	12,12	3,3	Skewed	90%	14	103163	85.17	10.01
3	12,12	3,3	Skewed	80%	26	102350	66.46	10.01

(*Continued*)

APPENDIX *(CONTINUED)*

Cell replicate	Feature levels				Criterion function	Relocation heuristic		B&B CPU time
	N, M	*K, L*	Spread	δ		# of restarts	Attraction rate	
3	12,12	6,3	Even	90%	14	50119	81.54	10.01
3	12,12	6,3	Even	80%	26	47791	65.36	10.06
3	12,12	6,3	Skewed	90%	12	98603	6.89	10.01
3	12,12	6,3	Skewed	80%	22	89874	4.41	10.05
3	12,12	6,6	Even	90%	14	22657	78.92	10.01
3	12,12	6,6	Even	80%	24	26771	17.14	10.35
3	12,12	6,6	Skewed	90%	14	31918	5.14	10.04
3	12,12	6,6	Skewed	80%	22	27586	8.58	10.26
3	18,12	3,3	Even	90%	23	42660	96.37	10.01
3	18,12	3,3	Even	80%	40	38060	95.63	10.01
3	18,12	3,3	Skewed	90%	23	46995	76.33	10.01
3	18,12	3,3	Skewed	80%	40	45853	63.10	10.01
3	18,12	6,3	Even	90%	23	22231	66.77	10.03
3	18,12	6,3	Even	80%	40	25393	33.12	10.13
3	18,12	6,3	Skewed	90%	23	41051	3.69	10.03
3	18,12	6,3	Skewed	80%	38	39116	1.77	10.49
3	18,12	6,6	Even	90%	19	11456	76.49	10.03
3	18,12	6,6	Even	80%	36	11462	32.52	11.12
3	18,12	6,6	Skewed	90%	23	17135	0.83	10.24
3	18,12	6,6	Skewed	80%	37	15197	1.40	27.96
3	18,18	3,3	Even	90%	32	22126	97.65	10.01
3	18,18	3,3	Even	80%	69	19966	77.05	10.07
3	18,18	3,3	Skewed	90%	32	24595	73.84	10.01
3	18,18	3,3	Skewed	80%	69	18946	77.57	10.12
3	18,18	6,3	Even	90%	32	13342	65.29	10.06
3	18,18	6,3	Even	80%	66	15371	21.16	35.10
3	18,18	6,3	Skewed	90%	32	31261	0.60	10.28
3	18,18	6,3	Skewed	80%	65	27879	0.22	534.04
3	18,18	6,6	Even	90%	32	6099	67.42	10.42
3	18,18	6,6	Even	80%	68	6110	34.58	2797.15
3	18,18	6,6	Skewed	90%	32	8759	2.80	16.80
3	18,18	6,6	Skewed	80%	62	7642	2.47	3526.76