New Approaches in Applied Statistics
Anuška Ferligoj and Andrej Mrvar (Editors)
Metodološki zvezki, 16, Ljubljana: FDV, 2000

Relational Calculator – A Tool for Analyzing Social Networks

Andrej Mrvar[1] and Vladimir Batagelj[2]

Abstract

A relational network consists of units interconnected by binary relation(s). For computational purposes relations are often represented by *binary matrices*. Representation by *directed graphs* is used to visualize networks or to give intuitive meaning to some notions.

Many operations on relations and their properties used for analyzing and describing a network structure can be found in literature. Beside matrices, two additional types of objects are needed to define these operations: *clusterings* and *permutations*. They are connected by many operations and transformations which form a powerful system for expressing and solving network analysis problems.

In this paper we describe a program `RelCalc` (Relational Calculator) which implements a computational system for these three data types. Some examples of application of the program are also presented.

1 Program `RelCalc`

Let $\mathbf{U} = \{X_i\}$ be a *set of units*. A *relational network* on $\mathbf{U}$ is called a structure

$$(\mathbf{U}, R_1, R_2, \ldots, R_k)$$

where $R_s \subseteq \mathbf{U} \times \mathbf{U}$, $s = 1, \ldots, k$.

For computational purposes relations are often represented by *binary matrices* $B(R) = [b_{ij}]$ where

$$b_{ij} = \begin{cases} 1 & X_i \, R \, X_j \\ 0 & \text{otherwise} \end{cases}$$

[1] Faculty of Social Sciences, University of Ljubljana, Kardeljeva pl. 5, Ljubljana, Slovenia.
[2] Department of Mathematics, University of Ljubljana, Jadranska 21, Ljubljana, Slovenia.

and a representation by *directed graphs* is usually used to visualize networks or to give intuitive meaning to some notions. Many operations and properties for analyzing and describing a network structure can be found in literature. Beside networks, two additional types of objects are needed to define these operations: *clusterings* and *permutations*.

A clustering $\mathcal{C} = (C_1, ..., C_m), C_i \subseteq \mathbf{U}$ and $i \neq j \Rightarrow C_i \cap C_j = 0$ can be described by a mapping $C: \mathbf{U} \rightarrow 0..n$; $C(i) = j$ iff unit i belongs to cluster j. A clustering $\mathcal{C}$ is *complete* iff $\cup_{i=1}^{m} C_i = \mathbf{U}$. In the case of incomplete clustering the unclassified units form a cluster C_0 – therefore $C(i) = 0$. A set $A \subseteq \mathbf{U}$ can be represented by a clustering $\mathcal{C} = (C_0, C_1), C_1 = A$.

Permutations $\pi: \mathbf{U} \rightarrow \mathbf{U}$ are often used for reordering units according to some criteria, e.g., reordering rows and columns of matrix of relation so that units that belong to the same cluster are neighbours.

In this paper we describe a program `RelCalc` which implements a computational system for these three data types. In the current implementation `RelCalc` contains the following tools for analyzing relations:

- *Properties of relations*: reflexivity, transitivity, symmetricity, equivalence relation, acyclic relation, ...;
- *Comparisons of relations*: equality of relations, subrelations, distances between relations, ...;
- *Basic operations on relations*: complement, transposition, union, intersection, symmetric sum, difference, multiplication, power, transitive closure;
- *Complex operations on relations*:
  - *factorisation*: partition of units according to strongly connected components (clustering) and factor relation;
  - *degree partition*: partition of units according to input, output or total degree;
  - *depth partition* of an acyclic relation;
  - *core partition*: k-core is a maximal connected subgraph in which every unit has at least k neighbours from the same k-core (Seidman, 1983);
  - *biconnected components*;
  - *transitive skeleton* of an acyclic relation;
  - *concatenation, substitution, and domain alternation.*

- *Additional operations*:
  - creating standard and random relations, clusterings, and permutations;
  - transforming clustering to permutation or equivalence relation;
  - rearranging relational matrix according to a given permutation;
  - transforming clustering to its canonical form;
  - transforming clustering into a set;
  - extracting subrelation determined by a set;
  - shrinking units;
  - transforming permutation to clustering – orbital partition;
  - mirroring of permutation;
  - union and intersection of clusterings;
  - determining the direct neighbours of a given binary clustering with respect to a given relation;
  - comparing clusterings and permutations;
  - removing subdivisions.

`RelCalc` is usually used in the interactive mode. It draws the trace of all user's requests into a log-file. This file can later be rerun to repeat the session and to restore the current state; or we can prepare in such files sequences of commands for frequently used procedures. For example, to compute dissimilarity between clusterings $\mathcal{C}_1$ and $\mathcal{C}_2$ defined as

$$d(\mathcal{C}_1, \mathcal{C}_2) = \mid \mathrm{eqv}(\mathcal{C}_1 \sqcup \mathcal{C}_2) \oplus \mathrm{eqv}(\mathcal{C}_1 \sqcap \mathcal{C}_2) \mid$$

we can use the following log-file

```
% RelCalc 0.1, April 28, 1999
Msg Input first clustering
RDC ?                    %%%  C   1
Msg Input second clustering
RDC ?                    %%%  C   2
Msg Calculation
UnC(1,2)                 %%%  C   3
InC(1,2)                 %%%  C   4
ClR-3                    %%%  R   1
ClR-4                    %%%  R   2
Sum(1,2)                 %%%  R   3
Msg Result
```

The commands `RDC` read the clusterings from input files into local clusterings 1 and 2; `UnC` computes their union, and `InC` their intersection. The commands `ClR` convert the clusterings to the corresponding equivalence relations. `Sum` computes the symmetric sum of both equivalences. To obtain the dissimilarity, we have to display the list of properties of the resulting relation and read its cardinality.

In the following sections we present some typical applications of program `RelCalc`.

2 Set operations and dissimilarities

Using set operations on relations, we can often reveal interesting connections between these relations. Bank Wiring Room (BWR) is, in network analysis, an often used example (Roethlisberger and Dickson, 1939; Borgatti et al., 1991: 204):

> 14 employees work in a single room and include two inspectors (I_1 and I_3), three solderers (S_1, S_2 and S_3), and nine wireman or assemblers (W_1 to W_9). The interaction categories include: games (participation in horseplay); conflict (participation in conflicts about open windows); positive (friendship); negative (antagonistic behaviour); help (helping others with work).

All these relations are given in Appendix. Consider the results of the following operations on these relations:

$$\begin{aligned} \text{negative} \cap \text{positive} &= \emptyset \\ \text{negative} \cap \text{help} &= \{(W_6, W_7)\} \\ \text{positive} \setminus \text{games} &= \{(W_7, S_1), (S_1, W_7)\} \end{aligned}$$

In words: (i) there are no pairs of actors who both like and dislike each other; (ii) there is a pair of actors (W_6 and W_7) who dislike each other yet one (W_6) helps the other (W_7); and (iii) among the actors who like each other only W_7 and S_1 do not play games together.

In Roethlisberger and Dickson (1939) the following partition into cliques was proposed

$$\mathcal{C} = \{\ \{I_1, W_1, W_2, W_3, W_4, S_1\},\ \{I_3\},\ \{W_5\},\ \{W_6, W_7, W_8, W_9, S_4\},\ \{S_2\}\ \}$$

Let $\text{eqv}(\mathcal{C})$ denote the corresponding equivalence relation. If we compute the relation

$$\text{games} \setminus \text{eqv}(\mathcal{C})$$

Table 1: Joint playing games and positive ties in the BWR.

	I_1	I_3	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	S_1	S_2	S_4
I_1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
I_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W_1	0	0	0	0	1	1	0	0	0	0	0	1	0	0
W_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W_3	1	0	1	0	0	1	0	0	0	0	0	1	0	0
W_4	0	0	1	0	1	0	0	0	0	0	0	1	0	0
W_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W_7	0	0	0	0	0	0	0	0	0	1	1	0	0	0
W_8	0	0	0	0	0	0	0	0	1	0	1	0	0	1
W_9	0	0	0	0	0	0	0	0	1	1	0	0	0	1
S_1	0	0	1	0	1	1	0	0	0	0	0	0	0	0
S_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S_4	0	0	0	0	0	0	0	0	0	1	1	0	0	0

we can see that the only unit in relation with others is the unit W_5. It is adjacent to $\{W_1, W_3, W_4, W_7, S_1\}$. All these units except W_7 are in the first class of the partition $\mathcal{C}$. If we move the unit W_5 into the first class, we get a new partition

$$\mathcal{C}' = \{\ \{I_1, W_1, W_2, W_3, W_4, W_5, S_1\},\ \{I_3\},\ \{W_6, W_7, W_8, W_9, S_4\},\ \{S_2\}\ \}$$

for which

$$\text{games} \setminus \text{eqv}(\mathcal{C}') = \{(W_5, W_7), (W_7, W_5)\}$$

We can also measure the *determinacy* of relation S by relation R by introducing the *determinacy* coefficient

$$K(R \Rightarrow S) = \begin{cases} 1 & S = \emptyset \\ \dfrac{|R \cap S|}{|S|} & \text{otherwise} \end{cases}$$

Since $k(\text{eqv}(\mathcal{C}') \Rightarrow \text{games}) = 0.96429$ the games relation can be almost completely explained by the property of belonging to the same clique in $\mathcal{C}'$.

We can define also the *mutual determinacy* coefficient:

$$K(R \Leftrightarrow S) = \begin{cases} 1 & R = S = \emptyset \\ \dfrac{|R \cap S|}{|R \cup S|} & \text{otherwise} \end{cases}$$

Consider again the BWR game playing ties together with the positive ties. There are 56 game playing ties and 26 positive ties. The intersection of the two relations is shown in Table 1.

Table 2: Playing games or positive ties in the BWR.

	I_1	I_3	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	S_1	S_2	S_4
I_1	0	0	1	1	2	1	0	0	0	0	0	0	0	0
I_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W_1	1	0	0	1	2	2	1	0	0	0	0	2	0	0
W_2	1	0	1	0	1	1	0	0	0	0	0	1	0	0
W_3	2	0	2	1	0	2	1	0	0	0	0	2	0	0
W_4	1	0	2	1	2	0	1	0	0	0	0	2	0	0
W_5	0	0	1	0	1	1	0	0	1	0	0	1	0	0
W_6	0	0	0	0	0	0	0	0	1	1	1	0	0	0
W_7	0	0	0	0	0	0	1	1	0	2	2	1	0	1
W_8	0	0	0	0	0	0	0	1	2	0	2	0	0	2
W_9	0	0	0	0	0	0	0	1	2	2	0	0	0	2
S_1	0	0	2	1	2	2	1	0	1	0	0	0	0	0
S_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S_4	0	0	0	0	0	0	0	0	1	2	2	0	0	0

There are 24 ties that are both positive and game playing. If R is game playing and S represents the positive ties then the determinacy coefficient for game playing determining positive ties is $24/26 = 0.92$. On the other hand, the determinacy coefficient for positive ties determining game playing is $24/56 = 0.43$. People who play games together tend to like each other while liking is much less likely to generate playing games together.

Table 2 shows the union of the positive ties and the game playing ties. A 1 indicates a single relation (either positive or game playing) while the 2 indicates both ties are present. Note that the 2s in Table 2 correspond exactly to the 1s in Table 1. The count of the union of ties (with a 2 counted only once) is 58. The coefficient of mutual determinacy is $24/58 = 0.41$.

In contrast, consider negative ties and helping ties. There are 24 helping ties and 38 negative ties. The two relations have only one tie (W_6, W_7) in common. The determinacy coefficient for helping determining negative ties is $1/38 = 0.026$ and the coefficient for negative ties determining helping is $1/24 = 0.042$. The coefficient of mutual determinacy is $1/61 = 0.016$. All of which is consistent with the two relations having a slight overlap.

There are different dissimilarity measures on relations. One example is the *Hamming distance* which is defined as:

$$d_H(R, S) = |R \oplus S|$$

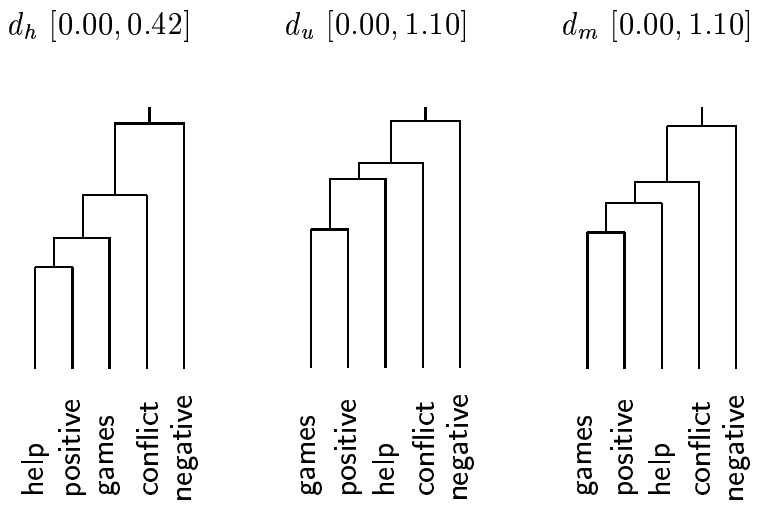

Figure 1: Dendrograms of five BWR relations for three dissimilarity measures.

Normally, it is necessary to normalize this measure. One normalization is:

$$d_h(R,S) = \frac{|R \oplus S|}{|\mathbf{U}|^2}$$

Other distances can also be introduced. For example

$$d_u(R,S) = \frac{|R \oplus S|}{|R \cup S|}$$

Another dissimilarity that could be used is:

$$d_m(R,S) = \begin{cases} 0 & R = S = \emptyset \\ \dfrac{\max(|R \setminus S|, |S \setminus R|)}{\max(|R|, |S|)} & \text{otherwise} \end{cases}$$

We computed three dissimilarity measures (d_h, d_u, and d_m) for five of the BWR relations (playing games, positive affect, negative affect, helping, and conflict over windows). The resulting clusterings (hierarchies) are represented graphically by dendrograms in Figure 1. The range of values of the dissimilarity measures are given with the dendrograms.

Clearly, the partitions differ, showing that both the measures and the relations differ. Using d_h, the helping and positive ties are the least dissimilar, yet for d_u and d_m, the game playing and positive ties are the least dissimilar. This implies that, on the technical side, we need to select dissimilarity measures with care, and on the substantive side, we can explore the nature of the relations among the same.

Table 3: Relations P and N and their products.

$$P = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad N = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

	P	N	PN	NP	N^2
P	P	PN	PN	N^2	N^2
N	NP	N^2	N^2	N^2	N^2
PN	N^2	N^2	N^2	N^2	N^2
NP	NP	N^2	N^2	N^2	N^2
N^2	N^2	N^2	N^2	N^2	N^2

Table 4: d_H, d_u and d_m distances between relations.

	d_H					d_u					d_m				
	N^2	PN	NP	N	P	N^2	PN	NP	N	P	N^2	PN	NP	N	P
N^2	0					0					0				
PN	2	0				0.22	0				0.22	0			
NP	2	2	0			0.22	0.25	0			0.22	0.14	0		
N	3	1	1	0		0.33	0.14	0.14	0		0.33	0.14	0.14	0	
P	5	7	7	8	0	0.56	0.78	0.78	0.89	0	0.56	0.71	0.71	0.83	0

3 Semigroup homomorphisms

Let us, for the next example, take two relations P and N (Bonacich and McConaghy, 1979: 516-520) which are shown in Table 3. These two relations generate the multiplication semigroup consisting of relations P and N and all their possible different products. In this example, the semigroup consists of five elements: P, N, NP, PN, N^2. The corresponding multiplication table is presented on the right side of Table 3. We want to simplify this semigroup by mapping it onto a less complex semigroup using a homomorphism. Some, almost equal, compound relations are identified. Any equation among semigroup elements implies a new set of congruence classes. The error inside congruence classes is measured by the maximum distance between any pair of relations in the class. Three different distance measures, which were defined in previous section, are used: d_H, d_u and d_m.

For this example, distances between relations are given in Table 4. All possible equations and corresponding congruence classes together with maximal distance (according to selected distance) are shown in Table 5.

The most preferred homomorphism is selected considering the error and the number of congruence classes. Let us for example take 3 congruence classes. Accordingly

Table 5: Congruence partitions, equations producing them and maximal distances.

Congruence classes	Equation	$d_{H,max}$	$d_{u,max}$	$d_{m,max}$
1. P,N,NP,PN,N^2	$P = N$	8	0.89	0.83
2. N | P,NP,PN,N^2	$P = NP$ or $P = PN$ or $P = N^2$	7	0.78	0.71
3. P | N,NP,PN,N^2	$N = N^2$	3	0.33	0.33
4. P | N | NP,PN,N^2	$PN = NP$	2	0.25	0.22
5. P | N,PN | NP,N^2	$N = PN$	2	0.22	0.22
6. P | N,NP | PN,N^2	$N = NP$	2	0.22	0.22
7. P | N | NP | PN,N^2	$PN = N^2$	2	0.22	0.22
8. P | N | PN | NP,N^2	$NP = N^2$	2	0.22	0.22
9. P | N | PN | NP | N^2	–	0	0	0

d_H and d_m distances suggest that homomorphisms 4, 5 and 6 are equally preferred, while using d_u distance, homomorphisms 5 and 6 are better than homomorphism 4.

4 Reachability problems

Using relational algebra, changes in states of certain systems can be investigated. The problem of two missionaries and two cannibals is a well known puzzle (Andrasfai, 1991):

> Two missionaries and two cannibals reach the left bank of a river simultaneously and all of them want to cross the river with the aid of a single available boat which carries no more than two persons. On no bank of the river may the cannibals outnumber the missionaries otherwise the former jeopardize the latter.

4.1 Description of the problem

The problem can be expressed in `RelCalc` introducing:

- The set of all feasible states **S** – states where the number of missionaries on both banks of the river is higher or equal to the number of cannibals. Each feasible state can be represented by a pair of numbers: number of missionaries and number of cannibals on the left bank of the river. All feasible states are:

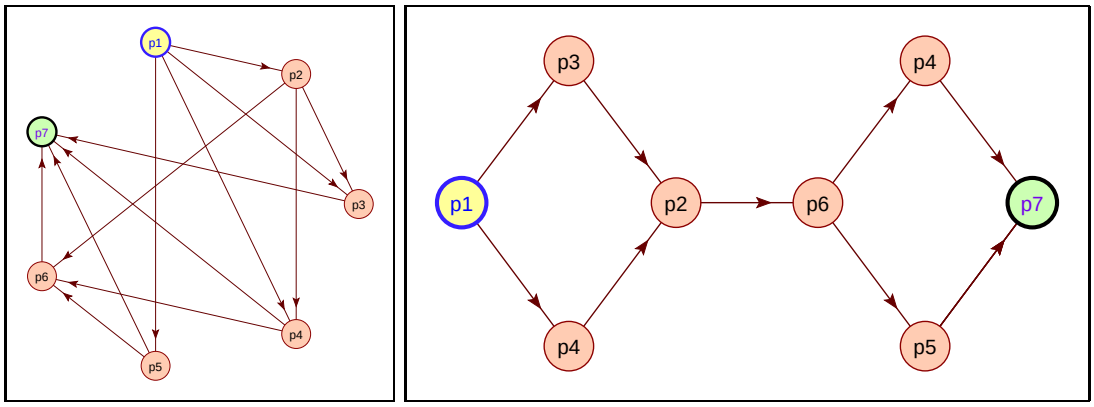

Figure 2: Possible transitions between states (left) and the shortest solutions (right).

$p_1 = (2,2)$, $p_2 = (2,1)$, $p_3 = (2,0)$, $p_4 = (1,1)$, $p_5 = (0,2)$, $p_6 = (0,1)$, $p_7 = (0,0)$.

- Transitions among states – which states can be obtained from a given state by crossing the river only once. Transitions between states are given by the graph on the left side of Figure 2. This graph determines the transition relation $R \subseteq \mathbf{S} \times \mathbf{S}$. In this graph only transitions when the river is crossed from left to right bank are shown. A graph with reversed arc directions describes the transitions when travelling from right to left bank.

State p_1 is the starting state and p_7 the final state. The problem asks: Is it possible to come from state p_1 to state p_7 following transitions shown in the graph on the left side of Figure 2 such that every second transition is in reverse direction (travelling from right to left bank).

4.2 Solution of the problem using `RelCalc`

States p_1 to p_7 can be represented using following clusterings:

$$p_1 = \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \quad p_2 = \begin{bmatrix}0\\1\\0\\0\\0\\0\\0\end{bmatrix} \quad p_3 = \begin{bmatrix}0\\0\\1\\0\\0\\0\\0\end{bmatrix} \quad p_4 = \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \quad p_5 = \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\end{bmatrix} \quad p_6 = \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\end{bmatrix} \quad p_7 = \begin{bmatrix}0\\0\\0\\0\\0\\0\\1\end{bmatrix}$$

and the relation R by its matrix on the left side of Table 6.

To solve the puzzle completely we have to answer the questions:

Table 6: Transition matrix and solvability matrix.

$$B(R) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad B((RR^{-1})^*R) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

1. *Does the problem have any solution?*
   The problem has the solution iff there exists a direction altering trail from p_1 to p_7. This is equivalent to $(p_1, p_7) \in (RR^{-1})^*R$ ($*$ means transitive closure of a relation). The matrix of $(RR^{-1})^*R$ is given on the right side of Table 6. The problem has a solution.

2. *What is the shortest solution?*
   We can answer this question by computing series of clusterings $(RR^{-1})^k R(p_1)$ – these are the states on the right bank of the river after $2k+1$ transitions and look for the first clustering in which the final state p_7 appears. The computation gives $k = 2$ – the lowest number of crossings is 5.

3. *Is the shortest solution unique?*
   There are different solutions of the problem. They can be reconstructed in the following way. Let $P_i = \underbrace{RR^{-1}RR^{-1}\cdots}_{i \text{ terms}}$ and $Q_i = \underbrace{R^{-1}RR^{-1}R\cdots}_{2k+1-i \text{ terms}}$, $i = 1, \ldots, 2k$, then the set $\mathbf{S}_i$ of states, i steps from the start, belonging to the solutions is determined by the expression $\mathbf{S}_i = P_i(p_1) \cap Q_i(p_7)$ Actually there are 4 solutions presented on the right side of Figure 2:

$$p_1 \to p_3 \to p_2 \to p_6 \to p_4 \to p_7 \qquad p_1 \to p_3 \to p_2 \to p_6 \to p_5 \to p_7$$
$$p_1 \to p_4 \to p_2 \to p_6 \to p_4 \to p_7 \qquad p_1 \to p_4 \to p_2 \to p_6 \to p_5 \to p_7$$

It is possible to generalize the problem of two missionaries and two cannibals to more missionaries and cannibals.

5 Conclusion

The packed binary matrix representation of relations used in `RelCalc` is appropriate only for relations of moderate size – some thousands of units. For example, relation

on 3000 units requires approximately 1M of memory.

n	time
50	0.22 s
100	1.48 s
150	4.45 s
250	21.09 s
500	2.76 min
720	8.14 min
1000	21.6 min
3000	9.5 h
10000	14.5 days

In fact the memory requirements are not the main limiting factor. `RelCalc` contains some operations on relations with complexity of order n^3 (transitive closure, multiplication, power) which are producing substantive delays already for some hundred of units. Table presents the execution times of computing transitive closures for $n \leq 720$. Using these times we computed the regression function:

$$time = 0.429281 - 0.008718n + 0.000052776n^2 + 0.00000125n^3$$

and used it for estimation of execution times for $n > 720$.

For the analysis of large networks we are developing a special program package `Pajek`. Using `Pajek` we can reduce a large network into smaller ones which can be further analysed by `RelCalc` or similar programs.

The last versions of `RelCalc` and `Pajek` are freely available at:

`http://vlado.fmf.uni-lj.si/pub/networks/`

References

[1] Aho, A.V., Hopcroft, J.E., and Ullman, J.D. (1976): *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA.

[2] Andrasfai, B. (1991): *Graph Theory: Flows, Matrices*. Akademiai Kiado, Budapest: Technical University of Budapest. 134-139.

[3] Batagelj, V., Mrvar, A. (1998). `Pajek` – A Program for Large Network Analysis. *Connections* **21**, 47-57.

[4] Bonacich, P. and McConaghy, M.J. (1979): The Algebra of Blockmodeling. In Schuessler, K. (Ed.), *Sociological Methodology*. San Fancisco: Jossey-Bass. 489-532.

[5] Borgatti, S.P., Everett, M.G., and Freeman, L.C. (1991): *UCINET IV, 1.00*, Analytic Technologies, Columbia.

[6] Fletcher, J.G. (1980): *A More General Algorithm for Computing Closed Semiring Costs Between Vertices of a Directed Graph*. CACM. 350-351.

[7] Roethlisberger, F. and Dickson, W. (1939): *Management and the Worker*. Cambridge: Cambridge University Press.

[8] Scott, J. (1991): *Social Network Analysis*. London: Sage Publications Ltd.

[9] Seidman, S. (1983): Network Structure and Minimum Degree, *Social Networks*, **5**, 269-287.

Appendix: Bank Wiring Room data

$$\mathbf{U} = \{I_1, I_3, W_1, W_2, W_3, W_4, W_5, W_6, W_7, W_8, W_9, S_1, S_2, S_4\}$$

$$\begin{aligned}\text{help} = \{ & (W_1,W_3), (W_1,W_9), (W_1,S_1), (W_2,W_3), (W_2,W_4),\\ & (W_2,S_1), (W_3,W_2), (W_4,W_1), (W_4,W_3), (W_4,W_6),\\ & (W_5,W_3), (W_6,W_3), (W_6,W_7), (W_6,W_8), (W_6,W_9),\\ & (W_7,S_4), (W_8,W_6), (W_8,W_7), (W_8,W_9), (W_9,S_4),\\ & (S_1,W_7), (S_2,W_6), (S_4,W_4), (S_4,W_8)\ \}\end{aligned}$$

$$\begin{aligned}\text{games} = \{ & (I_1,W_1), (I_1,W_2), (I_1,W_3), (I_1,W_4), (W_1,I_1),\\ & (W_1,W_2), (W_1,W_3), (W_1,W_4), (W_1,W_5), (W_1,S_1),\\ & (W_2,I_1), (W_2,W_1), (W_2,W_3), (W_2,W_4), (W_2,S_1),\\ & (W_3,I_1), (W_3,W_1), (W_3,W_2), (W_3,W_4), (W_3,W_5),\\ & (W_3,S_1), (W_4,I_1), (W_4,W_1), (W_4,W_2), (W_4,W_3),\\ & (W_4,W_5), (W_4,S_1), (W_5,W_1), (W_5,W_3), (W_5,W_4),\\ & (W_5,W_7), (W_5,S_1), (W_6,W_7), (W_6,W_8), (W_6,W_9),\\ & (W_7,W_5), (W_7,W_6), (W_7,W_8), (W_7,W_9), (W_7,S_4),\\ & (W_8,W_6), (W_8,W_7), (W_8,W_9), (W_8,S_4), (W_9,W_6),\\ & (W_9,W_7), (W_9,W_8), (W_9,S_4), (S_1,W_1), (S_1,W_2),\\ & (S_1,W_3), (S_1,W_4), (S_1,W_5), (S_4,W_7), (S_4,W_8),\\ & (S_4,W_9)\ \}\end{aligned}$$

$$\begin{aligned}\text{positive} = \{ & (I_1,W_3), (W_1,W_3), (W_1,W_4), (W_1,S_1), (W_3,I_1),\\ & (W_3,W_1), (W_3,W_4), (W_3,S_1), (W_4,W_1), (W_4,W_3),\\ & (W_4,S_1), (W_7,W_8), (W_7,W_9), (W_7,S_1), (W_8,W_7),\\ & (W_8,W_9), (W_8,S_4), (W_9,W_7), (W_9,W_8), (W_9,S_4),\\ & (S_1,W_1), (S_1,W_3), (S_1,W_4), (S_1,W_7), (S_4,W_8),\\ & (S_4,W_9)\ \}\end{aligned}$$

$$\begin{aligned}\text{negative} = \{ & (I_1,I_3), (I_1,W_2), (I_3,I_1), (I_3,W_5), (I_3,W_6),\\ & (I_3,W_7), (I_3,W_8), (I_3,W_9), (I_3,S_4), (W_2,I_1),\\ & (W_2,W_7), (W_2,W_8), (W_2,W_9), (W_4,W_5), (W_5,I_3),\\ & (W_5,W_4), (W_5,W_6), (W_5,W_7), (W_5,W_8), (W_5,W_9),\\ & (W_5,S_1), (W_5,S_2), (W_6,I_3), (W_6,W_5), (W_6,W_7),\\ & (W_7,I_3), (W_7,W_2), (W_7,W_5), (W_7,W_6), (W_8,I_3),\\ & (W_8,W_2), (W_8,W_5), (W_9,I_3), (W_9,W_2), (W_9,W_5),\\ & (S_1,W_5), (S_2,W_5), (S_4,I_3)\ \}\end{aligned}$$

$$\begin{aligned}\text{conflict} = \{ & (W_4,W_5), (W_4,W_6), (W_4,W_7), (W_4,W_9), (W_5,W_4),\\ & (W_5,W_6), (W_5,S_1), (W_6,W_4), (W_6,W_5), (W_6,W_7),\\ & (W_6,W_8), (W_6,W_9), (W_6,S_1), (W_6,S_4), (W_7,W_4),\\ & (W_7,W_6), (W_7,W_8), (W_7,W_9), (W_7,S_4), (W_8,W_6),\\ & (W_8,W_7), (W_8,W_9), (W_8,S_1), (W_8,S_4), (W_9,W_4),\\ & (W_9,W_6), (W_9,W_7), (W_9,W_8), (W_9,S_1), (S_1,W_5),\\ & (S_1,W_6), (S_1,W_8), (S_1,W_9), (S_1,S_4), (S_4,W_6),\\ & (S_4,W_7), (S_4,W_8), (S_4,S_1)\ \}\end{aligned}$$