Reordering the Reorderable Matrix as an Algorithmic Problem

Erkki Mäkinen and Harri Siirtola[*]

Department of Computer and Information Sciences
P.O. Box 607, FIN-33014 University of Tampere, Finland
{em,hs}@cs.uta.fi

Abstract. The Reorderable Matrix is a visualization method for tabular data. This paper deals with the algorithmic problems related to ordering the rows and columns in a Reorderable Matrix. We establish links between ordering the matrix and the well-known and much studied problem of drawing graphs. First, we show that, as in graph drawing, our problem allows different aesthetic criterions which reduce to known NP-complete problems. Second, we apply and compare two simple heuristics to the problem of reordering the Reorderable Matrix: a two-dimensional sort and a graph drawing algorithm.

1 Introduction

Bertin's Reorderable Matrix [1,2] is a simple visualization method to explore multivariate or multidimensional data. The basic principle is to transform a multidimensional data set into a 2D interactive graphic. The graphical presentation of a data set closely resembles the underlying data table in that it contains rows and columns. These rows and columns can be permuted, allowing different views of the data set. The actual data values are replaced with circles that have a size relative to the actual data value. The smallest value is represented as a 0-sized circle and the largest value as a circle filling the available area. While interacting with the visual presentation, the user has a chance to detect patterns in the presentation and to gain insight into the data. Human vision is well equipped for this kind of pattern recognition.

Figure 1 contains a small Reorderable Matrix [1, p. 32] having binary values only. In this simple example we have sixteen townships $A, B, C, \ldots, P$ for which we know the presence or absence of nine characteristics, $1, 2, 3, \ldots, 9$. The question is whether the same planning decisions should be applied to all these townships?

Processing the Reorderable Matrix in Figure 1 involves bringing together similar rows and columns. This in turn involves dragging rows and columns, one by one, and can take a while. We leave the intermediate steps out and represent the result in Figure 2.

From the arrangement in Figure 2 it is quite easy to see that there are obvious groups in townships. Townships H, K have similar characteristics and could be

[*] HCI Group TAUCHI, http://www.cs.uta.fi/hci/

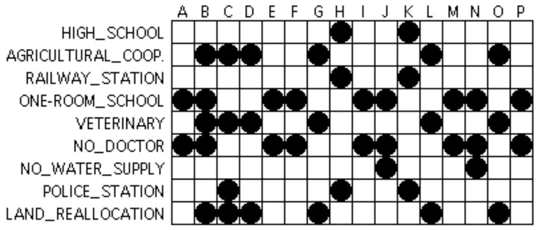


Fig. 1. The townships planning decisions example

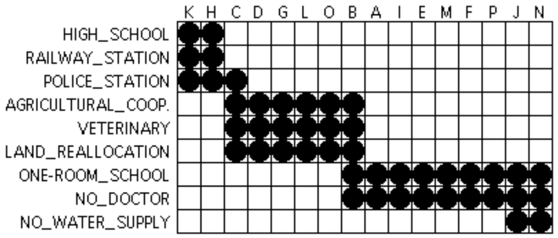


Fig. 2. An arrangement that reveals the correct decisions

labeled column-wise as CITIES and row-wise as URBAN. Similarly, townships N, J, P, M, I, F, E and A could be classified column-wise as VILLAGES and row-wise as RURAL. The remaining set could be called TOWNS, as an intermediate level between CITIES and countryside.

Even for this simple example there is a large number of *arrangements* or possible row and column permutations to be explored. For the townships example the number of possible arrangements is

$$(\#\ of\ rows)! \times (\#\ of\ columns)! > 7.5 \times 10^{18}.$$

Many of the arrangements are isomorphic as patterns – e.g., mirroring the matrix vertically or horizontally does not change the patterns – although the arrangements are not the same. Still, even the number of different patterns is high. In this example the size of the matrix was small, 16×9, when Bertin suggests that the Reorderable Matrix should be usable with sizes up to $x \times y = 10,000$. The largest experiment he writes about was $415 \times 76 = 31,850$. It is obvious that finding the interesting patterns calls for automation.

1.1 Exploring the Reorderable Matrix

The Reorderable Matrix can assist in a knowledge acquisition task in a number of ways. The general principle is to thread the matrix either horizontally or vertically according to, a column or a row, respectively. The chosen column or row should be one that seems to portray a phenomenon that has general influence. After threading, the matrix should be arranged so that similar rows appear together, forming black areas in the matrix. This kind of arrangement is ready to be analyzed.

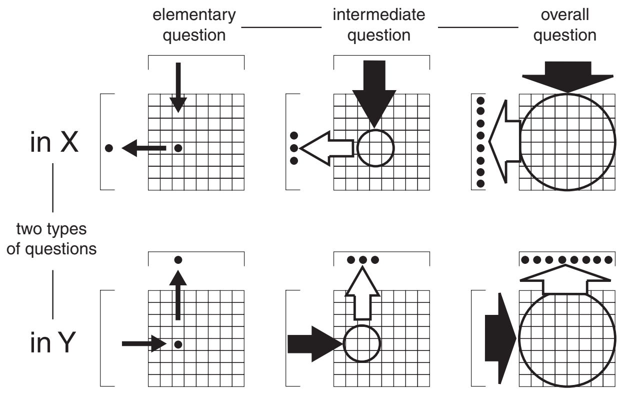


Fig. 3. Three levels of information: elementary, intermediate and overall [1, p. 13]

There are three kind of questions that can be answered with an appropriately arranged matrix: questions about certain objects or characteristics, questions about subsets of objects or characteristics and questions about overall object or characteristic sets. Figure 3 illustrates these three levels of information.

1.2 Motivation for This Work

As far as we know, ordering the Reorderable Matrix has not been regarded before as an algorithmic problem in the literature. Only [9] contains a remark on the possible computational difficulty of the problem.

There exists a great variety of algorithmic problems concerning the Reorderable Matrix. The situation resembles that of graph drawing algorithms: depending on the aesthetic criterions applied, the graph drawing problem may reduce to various known combinatorial and other problems. We can minimize the number of edge crossings, display as much symmetries as possible, or draw the vertices evenly over the drawing area, just to mention a few possibilities (for details and other possibilities, see [4]). One of our main goals here is to show that the same applies to the Reorderable Matrix, i.e., depending on the view chosen, we can model the task of reordering the matrix by using various different combinatorial problems. Notice that both in graph drawing and in ordering the Reorderable Matrix, the different aesthetic criterions are often conflicting, i.e., it is not usually possible to simultaneously fulfill more than one criterion.

Note that we do not give an exact mathematical definition for the problem of reordering the Reorderable Matrix. As an implication, we cannot (mathematically) argue that the exactly formulated problems match to the original reordering problem. However, it is obvious that the exact problems to be discussed later in this paper are closely related to relevant subproblems of reordering the Reorderable Matrix. The situation is analogous to the one of drawing graphs "nicely": we have to always define what we mean by "nicely", and the exact mathematical definitions given do not cover all aspects of nice drawings.

2 Preliminaries

We assume a familiarity with the rudiments of the theory of computational complexity and NP-complete problems as given in [8].

The Reorderable Matrix can be regarded as an $m \times n$ matrix with entries from the set $\{0, 1, \ldots, e\}$. The entry in row i and column j in matrix M is denoted by M_{ij}. The submatrix containing the entries M_{ij}, where $1 \le i \le p$ and $1 \le j \le q$, is said to be the *upper* $(p, q) - submatrix$ of M. Similarly, the entries M_{ij}, where $p \le i \le m$ and $q \le j \le n$, form the *lower* $(p, q) - submatrix$ of M (see Figure 4).

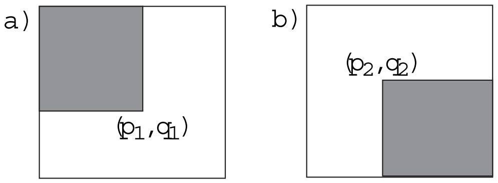


Fig. 4. (a) Upper (p_1, q_1)-submatrix and (b) lower (p_2, q_2)-submatrix

Graph bandwidth minimization [3,7,8] is perhaps the most well-known problem type involving matrices and row and column permutations. In bandwidth minimization our goal is to permute the rows and columns of a given matrix such that the non-zero entries of the matrix form as thin a "band" as possible along the main diagonal (see Figure 5).

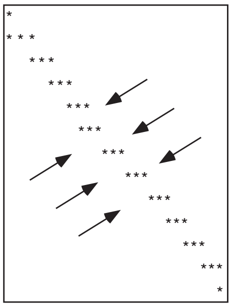


Fig. 5. The purpose of bandwidth minimization (* stands for a non-zero entry)

However, the link between graphs and matrices implies that in these problems row and columns permutations are performed simultaneously, while in reordering the Reorderable Matrix row and column permutations are independent of each other. Despite the differences in formulation, the NP-completeness of various bandwidth problems gives us a hint that most reasonable problems concerning the Reorderable Matrix must also be NP-complete.

Another well-known matrix operation closely resembling the Reorderable Matrix permutation is Gaussian elimination, where rows and columns are multiplied by and subtracted from each other. Contrary to the bandwidth problems, we now handle rows and columns independently. On the other hand, in Gaussian elimination we do not *permute* rows and columns, but perform arithmetic operations between them. For a NP-complete problem formulation related to Gaussian elimination (DIRECTED ELIMINATION ORDERING), see [8,12].

At least the following known NP-complete problems have a close connection with the Reorderable Matrix.

MATRIX DOMINATION [8]
Instance: An $n \times n$ matrix M with entries from $\{0, 1\}$, and a positive integer K.
Question: Is there a subset $C \subseteq \{1, 2, \ldots, n\} \times \{1, 2, \ldots, n\}$ with at most K elements such that $M_{ij} = 1$ for all $(i, j) \in C$, and whenever $M_{ij} = 1$, then there exists an (i', j') in C for which either $i = i'$ or $j = j'$?

RECTILINEAR PICTURE COMPRESSION [8]
Instance: An $n \times n$ matrix M with entries from $\{0, 1\}$, and a positive integer K.
Question: Is there a collection of K or fewer quadruples (a_i, b_i, c_i, d_i), $1 \leq i \leq K$, where $a_i \leq b_i$, $c_i \leq d_i$, such that for every pair (i, j), $1 \leq i, j \leq n$, $M_{ij} = 1$ if and only if there exists k, $1 \leq k \leq K$, such that $a_k \leq i \leq b_k$ and $c_k \leq j \leq d_k$?

3 Reordering Aesthetics

The problem formulations presented in the previous section use square matrices. The Reorderable Matrix is not usually square, but for notational convenience and with no loss of generality, we will only deal with square matrices.

A typical approach in ordering the Reorderable Matrix is to arrange the rows and columns such that there are "black areas" in the top left and bottom right corners implying "white areas" in the top right and bottom left corners. As we assume that the matrix entries are from the set $\{0, 1, \ldots, e\}$, we can define 'white' to be any of the values $0, 1, \ldots, c$, and 'black' to be any of the values $c+1, c+2, \ldots, e$, with an appropriate constant c (the value of which is not fixed here). This aesthetic can be formalized as follows.

PROBLEM1

Instance: An $n \times n$ matrix M with non-negative entries, and an integer K.
Question: Is it possible to perform K or less row permutations and K or less

column permutations such that all the non-white entries appear in the upper (K, n)-submatrix or in the lower $(K+1, n+1)$-submatrix?

Since PROBLEM1 clearly is in NP, its NP-completeness can be proved by defining a polynomial transformation from MATRIX DOMINATION.

Theorem 1. *PROBLEM1 is NP-complete.*

Proof. Consider an instance of MATRIX DOMINATION with an $n \times n$ matrix P and an integer k. The corresponding instance of PROBLEM1 consists of an $(n+k) \times (n+k)$ matrix and an integer $K = k$. The new matrix M has the form shown in Figure 6. The $n \times n$ submatrix is the original matrix P, while the new upper (k, n)-submatrix, the new lower $(k+1, n+1)$-submatrix and the new $k \times k$ matrix in the upper right corner contain zeros only.

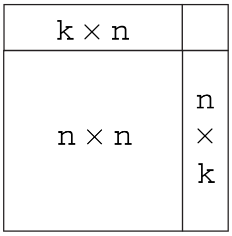


Fig. 6. The matrix in the instance of PROBLEM1

Suppose the instance of MATRIX DOMINATION is related with "yes" answer, i.e. there are K or less non-zero entries dominating P. Arbitrarily order the dominating entries: $P_{i(1)j(1)}, P_{i(2)j(2)}, \ldots, P_{i(\kappa)j(\kappa)}$, where $\kappa \leq k$. For each dominating entry $P_{i(t)j(t)}$, $t = 1, \ldots, \kappa$, permute rows t and $k + i_t$ and columns j_t and $n + t$ in M. Since each non-zero entry of P is dominated by some of the entries $P_{i(t)j(t)}$, $t = 1, \ldots, \kappa$, the permutations done move all the non-white elements of M to the upper (k, n)-submatrix or to the lower $(k+1, n+1)$-submatrix. On the other hand, if such permutations are possible in M, then P must be dominated by $k = K$ or less entries. This completes the proof.

Wilf [15, Section 2.4.], has posed open the completeness status of a problem somewhat similar to PROBLEM1. In Wilf's problem one is asked to find row and column permutations such that the resulting matrix is triangular.

PROBLEM1 is related to a case where we expect that there are two "clusters" of positively correlating factors. In general, there can be any number of such clusters, i.e., any number of "black areas" in the matrix. This, in turn, can be modeled by RECTILINEAR PICTURE COMPRESSION. Again, we consider binary matrices and leave open the definitions of "white" and "black" entries.

PROBLEM2

Instance: An $n \times n$ matrix M with entries from $\{0, 1\}$, and a positive integer K.
Question: Is it possible to perform row and column permutations in M such that

there eventually is a collection of K or fewer quadruples (a_i, b_i, c_i, d_i), $1 \leq i \leq K$, where $a_i \leq b_i$, $c_i \leq d_i$, $1 \leq i \leq K$, such that for every pair (i, j), $1 \leq i, j \leq n$, $M_{ij} = 1$ if and only if there exist a k, $1 \leq k \leq K$, such that $a_k \leq i \leq b_k$ and $c_k \leq j \leq d_k$?

As an example, consider the matrix in Figure 2. Its black areas can be represented by five rectangles in several different ways. One of the solutions with five rectangles is $\{(1, 2, 1, 3), (3, 3, 3, 3), (3, 8, 4, 6), (8, 16, 7, 8), (15, 16, 9, 9)\}$.

Theorem 2. *PROBLEM2 is NP-complete.*

Proof. PROBLEM2 has RECTILINEAR PICTURE COMPRESSION as a subproblem. Since PROBLEM2 clearly is in NP, the theorem follows immediately by restriction [8](pp. 63–66) from the NP-completeness of RECTILINEAR PICTURE COMPRESSION.

Also the following known NP-complete problems could be used as the basis for formulating a reordering aesthetic:

MATRIX COVER [8]
Instance: An $n \times n$ matrix M with non-negative entries, and an integer K.
Question: Is there a function $f : \{1, 2, \ldots, n\} \rightarrow \{+1, -1\}$ such that

$$\sum_{1 \leq i,j \leq n} M_{ij} \cdot f(i) \cdot f(j) \leq K?$$

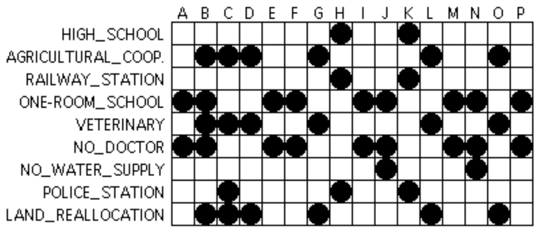
TRIE COMPACTION [5,11]
Instance: A multiset $\{X(1), X(2), \ldots, X(n)\}$ of bit strings of length m and an integer K.
Question: Is it possible to place the strings such that overlapping is possible if in overlapping positions at most one of the strings has a non-null content and such that the total length of the resulting bit string is at most K?

However, we omit the details of these aesthetics here.

4 Methods for Reordering

We have shown that certain subproblems of reordering the Reorderable Matrix are closely related to well-known NP-complete problems. Hence, it is reasonable to consider heuristic approaches to the problem.

We present two methods for reordering the Reorderable Matrix. The former is a simple two dimensional (2D) sort and the latter is a variant of Sugiyama's graph layout algorithm. Both algorithms were implemented and tested with a number of matrices (see web page `http://www.cs.uta.fi/~hs/iv99/`). We present two of these test cases in the following discussion.



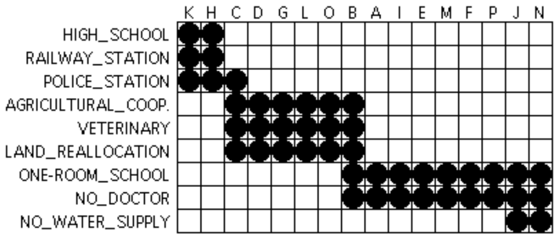
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
HIGH_SCHOOL								●			●					
AGRICULTURAL_COOP.		●	●	●			●					●			●	
RAILWAY_STATION								●			●					
ONE-ROOM_SCHOOL	●	●			●	●			●	●			●	●		●
VETERINARY		●	●	●			●					●			●	
NO_DOCTOR	●	●			●	●			●	●			●	●		●
NO_WATER_SUPPLY										●				●		
POLICE_STATION			●					●			●					
LAND_REALLOCATION		●	●	●			●					●			●	

Fig. 7. The townships matrix

4.1 The Townships Example

The first test case is the townships matrix we presented earlier. It is a simple binary matrix with obvious 'interesting' sets of arrangements. The initial arrangement is displayed again in Figure 7.

In Figure 8 we have one of the 'interesting' arrangements for the townships example. Subsets in the data with similar characteristics can be easily seen.



	K	H	C	D	G	L	O	B	A	I	E	M	F	P	J	N
HIGH_SCHOOL	●	●														
RAILWAY_STATION	●	●														
POLICE_STATION	●	●	●													
AGRICULTURAL_COOP.			●	●	●	●	●	●								
VETERINARY			●	●	●	●	●	●								
LAND_REALLOCATION			●	●	●	●	●	●								
ONE-ROOM_SCHOOL								●	●	●	●	●	●	●	●	●
NO_DOCTOR								●	●	●	●	●	●	●	●	●
NO_WATER_SUPPLY															●	●

Fig. 8. One of the 'optimal' arrangements for the townships example

The set of 'interesting' arrangements in this example is quite large, as Figure 9 illustrates. Most of the rows and columns inside subsets CITIES, TOWNS and VILLAGES could be in any order and the information would still be the same.

4.2 The Hotel Example

The second test case contains monthly data compiled from an imaginary hotel [1, pp. 1–11]. The object set is twelve months and the characteristics are properties that hotel management could use for business planning.

The data contains various numbers describing the clientele, the average price of room and the average length of stay. Most of the numbers are percent figures, except for the average length of stay, the average price of rooms and the information whether or not there was a convention during that month. The average length of stay is given in days and the average price of room is given in the local

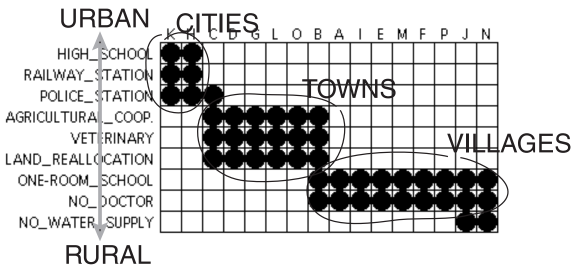


Fig. 9. The interesting subsets for the townships example

J	F	M	A	M	J	J	A	S	O	N	D		
26	21	26	28	20	20	20	20	20	40	15	40	1	% CLIENTELE FEMALE
69	70	77	71	37	36	39	39	55	60	68	72	2	% —〃— LOCAL
7	6	3	6	23	14	19	14	9	6	8	8	3	% —〃— U.S.A.
0	0	0	0	8	6	6	4	2	12	0	0	4	% —〃— SOUTH AMERICA
20	15	14	15	23	27	22	30	27	19	19	17	5	% —〃— EUROPE
1	0	0	8	6	4	6	4	2	1	0	1	6	% —〃— M.EAST, AFRICA
3	10	6	0	3	13	8	9	5	2	5	2	7	% —〃— ASIA
78	80	85	86	85	87	70	76	87	85	87	80	8	% BUSINESSMEN
22	20	15	14	15	13	30	24	13	15	13	20	9	% TOURISTS
70	70	75	74	69	68	74	75	68	68	64	75	10	% DIRECT RESERVATIONS
20	18	19	17	27	27	19	19	26	27	21	15	11	% AGENCY —〃—
10	12	6	9	4	5	7	6	6	5	15	10	12	% AIR CREWS
2	2	4	2	2	1	1	2	2	4	2	5	13	% CLIENTS UNDER 20 YEARS
25	27	37	35	25	25	27	28	24	30	24	30	14	% —〃— 20-35 —〃—
48	49	42	48	54	55	53	51	55	46	55	43	15	% —〃— 35-55 —〃—
25	22	17	15	19	19	19	19	19	20	19	22	16	% —〃— MORE THAN 55 —〃—
163	167	166	174	152	155	145	170	157	174	165	158	17	PRICE OF ROOMS
1.65	1.71	1.65	1.91	1.90	2.00	1.54	1.60	1.73	1.82	1.66	1.44	18	LENGTH OF STAY
67	82	70	83	74	77	56	62	90	92	78	55	19	% OCCUPANCY
			X	X	X			X	X	X	X	20	CONVENTIONS

Fig. 10. The hotel example data table [1, pp. 1–11]

currency. Hotel management would use this kind of data to design marketing, to define price structure, and to plan services offered to the customers.

The data table in Figure 10 is displayed as a Reorderable Matrix in Figure 11. This matrix is not a binary one, but contains numbers from very different domains.

4.3 2D Sort Method

The 2D sort method is a simple heuristics that tries to build black areas to the top left and the bottom right corners of the matrix. This is done by comparing weighted row and column sums and sorting the matrix repeatedly according to these values.

The 2D sort method first arranges the matrix into ascending order according to weighted row sums. The weights are the column positions of each cell. The next phase arranges the matrix again in a similar manner, but now in column-wise direction. These two phases are repeated until no row or column exchanges occur. This termination condition requires that the sort algorithm must be stable.

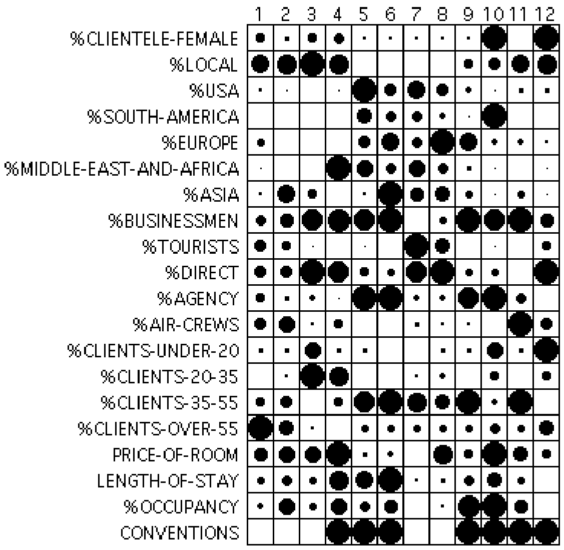


Fig. 11. The hotel data table as a Reorderable Matrix

Generally, when we want to find interesting arrangements, we can have some kind of hypothesis to start with. Usually we want to specify a row or a column that we believe to have overall influence in the data set. In 2D sort this can be accomplished by sorting the matrix according to a single row or a column and then running the 2D sort. This reference row or column will not necessarily stay where we place it – or even remain unchanged during the 2D sort – but it will certainly be one of the driving factors in the sort. This feature can be used to generate different arrangements from the matrix.

4.4 2D Sort: The Townships Example

In Figure 12 we have the townships matrix sorted with the 2D sort. As can be seen, the subsets CITIES, TOWNS and VILLAGES were formed, but townships C, B, J and N caused problems. These townships do not fall completely inside classifications, but have some characteristics that the other similar townships lack. However, for a user it would be a simple task to 'correct' the arrangement to the one preferred by human readers (as in Figure 8).

The 2D sort could be further tuned by experimenting with the weight distribution. Possibly a non-linear distribution of weights could 'draw' the black areas together even better.

4.5 2D Sort: The Hotel Example

Figure 13 shows the result of arranging the hotel example with the 2D sort. This particular operation was initiated by threading the matrix according to the

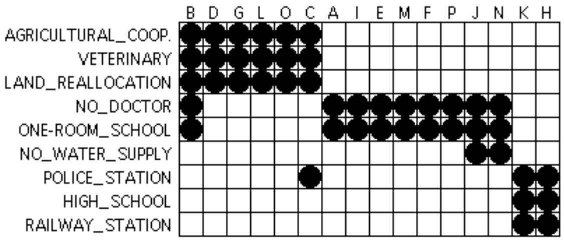


Fig. 12. The townships matrix sorted with the 2D sort

row %BUSINESSMEN and then issuing the sort command. As can be seen, the reference row is not the top row in the resulting arrangement, but it is still part of the black area appearing in the top left corner.

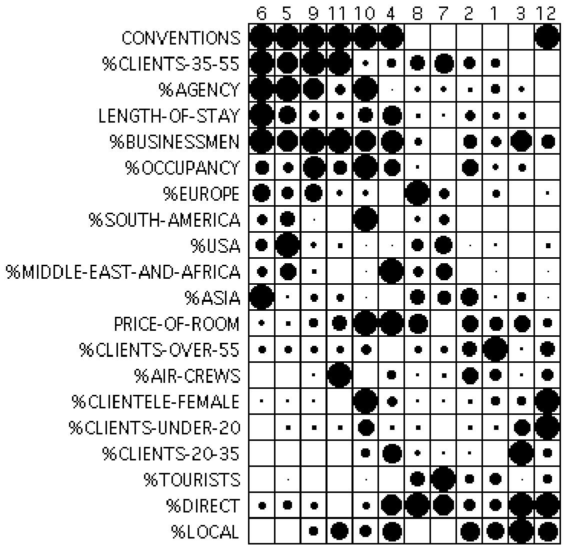


Fig. 13. The hotel matrix sorted with the 2D sort

What can be seen from or how should we interpret the arrangement in Figure 13? At least the following facts are easily found:

- Businessmen are mainly from the age group 35 to 55 years old.
- Businessmen make their room reservations through an agency.
- Businessmen stay at the hotel when there is a convention in town.
- Businessmen usually stay longer than other guests.

Choosing another reference row not appearing in this arrangement's black areas will probably produce new observations from the data set.

4.6 Sugiyama's Algorithm

The problem of drawing bipartite graphs with as few edge crossings as possible is a much studied subproblem of graph drawing. It is assumed that the vertices are drawn in horizontal lines such that separate vertex sets are placed in different horizontal lines and edges are drawn as straight lines between the sets. The drawing problem reduces to the problem of ordering the vertices in the lines such that the number of edge crossings is minimized.

There are actually two separate problems: we can fix the order in one of the horizontal lines and ask the optimal order of vertices in the other line, or we can order the vertices in both vertex sets of the bipartite graph in question, i.e., order the vertices in both of the horizontal lines. Even the former problem is known to be NP-complete [6]. Next, we will discuss the latter, somewhat harder problem. Since there is no possibility of confusion, we call it simply the *drawing problem.*

A well-known heuristic approach to the drawing problem is Sugiyama's algorithm [14] which is based on the *average heuristic* (sometimes called the barycenter heuristic). In the average heuristic we order the vertices according to the averages of their adjacent vertices in the opposite vertex set. By repeating this ordering process in turns in the two vertex sets, we (hopefully) reach orderings of vertices which minimize the number of edge crossings. Figure 14 illustrates a sample bipartite graph whose vertices are ordered according to this heuristic. For further information concerning the drawing problem, consult [4,6,10].

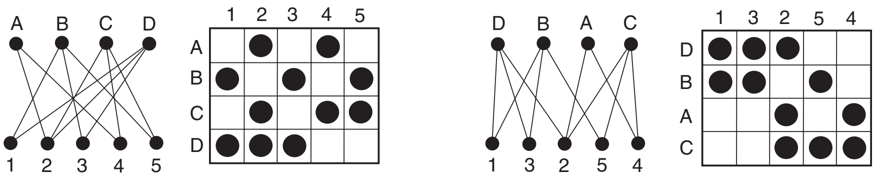


Fig. 14. Applying the average heuristic to a simple bipartite graph

Figure 14 shows how the adjacency matrices of the bipartite graphs are changed when applying the averaging heuristic. It is evident that when applying the heuristic, the corresponding adjacency matrices have the tendency to be reordered so that there are "black areas" in the top left and bottom right corners. This is just what we wanted to establish in connection with PROBLEM1. In what follows we try to make use of the averaging heuristic when ordering the Reorderable Matrix.

A clear difference between the drawing problem and the ordering of the Reorderable Matrix is that while in the former problem we have binary values only, the latter one deals with the set $\{0, 1, \ldots, e\}$. Our first solution to this problem is very simple: we suppose that the possible values of the matrix are divided into two categories: “blacks” and “whites” or 1’s and 0’s. Although in some sense self-evident, this approach has also a “deeper” rationale: the user of the Reorderable Matrix can define different “threshold values” for different objects and their attributes. For some object in an application, a certain value is “black” if the user considers it significant enough.

4.7 Sugiyama’s Algorithm: The Townships Example

Sugiyama’s algorithm will produce different arrangements depending on the number of up and down iterations. With some data sets the arrangement will converge and become stationary, but with some other data sets the arrangement may converge for a moment and then pulsate between a number of states. So, the problem in producing ‘interesting’ arrangements with Sugiyama’s algorithm is with these stationary layouts – you get exactly one arrangement, no matter what the initial setting is.

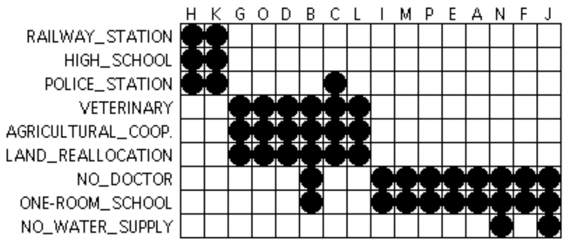


Fig. 15. The townships example drawn with Sugiyama’s algorithm

The townships example is almost a stationary arrangement. The only elements in the matrix that do not converge are the townships that do not fit into classifications. The arrangement in Figure 15 is remarkably close to the optimal human-arranged one: moving three columns will produce the same arrangement as in Figure 8.

4.8 Sugiyama’s Algorithm: The Hotel Example

Instead of dividing the matrix entries into “blacks” and “whites” we can also deal with the original value set $\{0, 1, \ldots, e\}$. In that case we have to replace the normal averages by weighted ones. As above, we deal with the rows and columns of the matrix in turns. Since the entries of an $m \times n$ matrix M can have values $0, 1, \ldots, k$, we count the sums $\sum_{j=1}^{n} j \cdot M_{ij}$ (for rows) and $\sum_{i=1}^{m} i \cdot M_{ij}$ (for columns), and order the rows and columns according to the sums obtained.

In Figure 16 we have chosen on purpose an arrangement that is close to the arrangement presented in Figure 13, the 2D sort version of the hotel example.

The difference in the layout is clear: in this version the black area will occur on the diagonal and close to the middle of the matrix. However, the same kind of observations from the data as made with the 2D sort version can also be made from this arrangement.

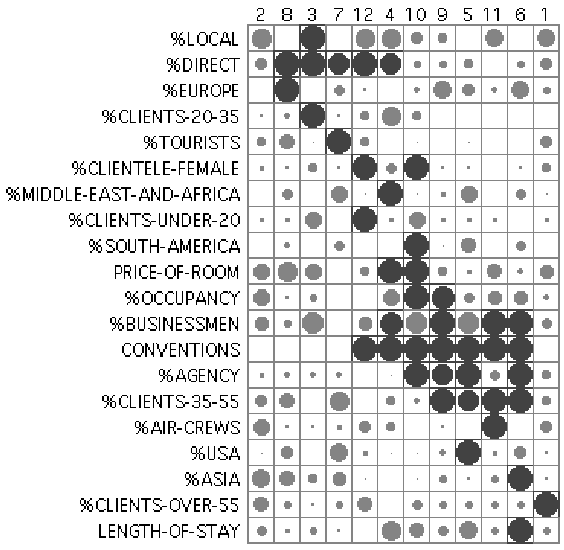


Fig. 16. The hotel example drawn with Sugiyama's algorithm

In the implementation that produced the arrangement in Figure 16, we used normalized values in the range $[0, 1]$. The threshold value for considering cells 'black' was 0.89 and was chosen by experimenting with various settings. In Figure 16 the cells that have a value above the threshold value are black and the cells below the threshold value are grey.

5 Concluding Remarks

We have proved that various problems related to the reordering aesthetics of the Reorderable Matrix are NP-complete. This suggests that it is reasonable to try heuristic reordering approaches.

We have shown that the averaging heuristic for minimizing the number of edge crossings when drawing bipartite graphs is well suited reordering the Reorderable Matrix. We have implemented both a binary version and its generalization which works with all entry values.

We believe that both the 2D sort method and Sugiyama's algorithm are useful in assisting the user while exploring the Reorderable Matrix. However, the usability of these ideas must be verified with user testing. We have already done usability experiments with the basic Reorderable Matrix [13].

Acknowledgements

This work was supported by the Academy of Finland (project 35025) and by the National Technology Agency TEKES (project 20287).

References

1. J. Bertin. *Graphics and Graphic Information Processing.* Walter de Gruyter & Co., Berlin, 1981. (Originally *La graphique et le traitemente graphique de l'information*, 1967, translated in English by William J. Berg and Paul Scott). 453, 455, 460, 461
2. J. Bertin. *Semiology of Graphics – Diagrams Networks Maps.* The University of Wisconsin Press, 1983. (Originally *Sémiologue graphique*, 1967, translated in English by William J. Berg). 453
3. P. Z. Chinn, J. Chvátalová, A. K. Dewdney, and N. E. Gibbs, The bandwidth problem for graphs and matrices – a survey. *J. Graph Theory* **6** (1992), 223–254. 456
4. G. Di Battista, P. Eades, R. Tamassia and I. G. Tollis, Annotated bibliography on graph drawing algorithms. *Comput. Geom. Theory Appl.* **4** (1994) 235–282. 455, 464
5. J. M. Dill, Optimal trie compaction is NP-complete. Cornell University, Dept. of Computer Science, Report **87-814**, March 1987. 459
6. P. Eades and N. Wormald, Edge crossings in drawings of bipartite graphs. *Algorithmica* **10** (1994), 361–374. 464
7. M. R. Garey, R. L. Graham, D. S. Johnson, and D. E. Knuth, Complexity results for bandwidth minimization. *SIAM J. Appl. Math.* **34** (1978), 477–495. 456
8. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. H. Freeman, 1979. 456, 457, 459
9. H. Hinterberger and C. Schmid, Reducing the influence of biased graphical perception with automatic permutation matrices. In *Proceedings of the Seventh Conference on Scientific Use of Statistic-Software, SoftStat93*, Heidelberg. Gustav Fischer Verlag, Stuttgart, 1993, pages 285–291. 455
10. M. Jünger and P. Mutzel, 2-layer straightline crossing minimization: performance of exact and heuristic algorithms. *J. Graph Algorithms and Applications* **1** (1997), 1–25. 464
11. J. Katajainen and E. Mäkinen, A note on the complexity of trie compaction. *EATCS Bull.* **41** (1990), 212–216. 459
12. D. J. Rose and R. E. Tarjan, Algorithmic aspects of vertex elimination of directed graphs. *SIAM J. Appl. Math.* 34 (1978), 176–197. 457
13. H. Siirtola. Interaction with the Reorderable Matrix. In E. Banissi, F. Khosrowshahi, M. Sarfraz, E. Tatham, and A. Ursyn, editors, *Information Visualization IV'99*, pages 272–277. Proceedings International Conference on Information Visualization, IEEE Computer Society, July 1999. `http://www.cs.uta.fi/~hs/iv99/siirtola.pdf`. 466
14. K. Sugiyama, S. Tagawa, and M. Toda, Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.* **SMC-11** (1981), 109–125. 464

15. H. S. Wilf, On crossing numbers, and some unsolved problems. B. Bollobás and A. Thomason (eds), *Combinatorics, Geometry, and Probability: A Tribute to Paul Erdös. Papers from the Conference in Honor of Erdös' 80th Birthday Held at Trinity College*, Cambridge University Press, 1997, pages 557-562. 458