Approximate Seriation in Formal Concept Analysis

François Brucker, Pascal Préa

Approximate Seriation in Formal Concept Analysis

François Brucker
Pascal Préa*

École Centrale Marseille
LIF, Laboratoire d'Informatique Fondamentale de Marseille, CNRS UMR 7279
Technopôle de Château-Gombert, 38, rue Joliot Curie, 13451 Marseille Cédex 20, France
{pascal.prea, francois.brucker}@lif.univ-mrs.fr

May 17, 2016 – 15:40

Abstract

We present in this paper an algorithmically efficient (linear in the size of the formal context) method to solve the seriation problem for formal contexts. We show that any maximal solution can be represented by a PQ-Tree. Moreover, the set of PQ-Trees can be seen as a distributive lattice. This lattice yields a consensus method which deals with the multiple solutions.

Key Words

Seriation, PQ-Trees, Consecutive One's Property, Lattice, Consensus.

1 Introduction

The classical problem of *seriation* in Archeology [9] is the following: we are given a set of *objects* (different kinds of necklace, bracelet, dishes...) and a set of *sites* (tombs, houses,...). Each object has been used during an interval of time, and each site contains objects. The problem is to order the sites along time. Similar problems arise in genetics [2][6], hypertext browsing [3], philology [4], data visualization [7][10], musicology[8], ...

Formally speaking, a seriation problem can be represented by a formal context (G, M, I) where G is the set of objects, M the set of sites and I the relation which states if an object g has been found in site m.

In *exact seriation*, on each site, one finds all objects that were used when the site was built/active. Equivalently, the formal context matrix $\mathcal{M}$ has the *Consecutive One's Property* (*C1P*), *i.e.* the lines of $\mathcal{M}$ can be reordered in such

*Corresponding author.

a way that on every column of $\mathcal{M}$, the *1*s appear in consecutive order. An optimal algorithm to find such a reordering was introduced in 1976 [5], based on a special data structure: *PQ-Trees*. In *approximate seriation*, a site may not contain an object that was used when it was built; and the problem of seriation is no longer equivalent to C1P.

The aim of this paper is to present an efficient algorithm for approximate seriation, also based on PQ-Trees. This paper is organized as follows: in Section 2, we present the PQ-Tree structure and a first algorithm. In Section 3, we show that the set of PQ-Trees can be organized as a lattice, which generalizes the semilattice of hierarchies; and we give a second algorithm. In Section 4, we present a possible workflow of our method on an archeological data set.

2 PQ-Trees

Given a finite set X, a *PQ-tree* T on X is a tree that represents a set of permutations on X denoted by S_T. The leaves of T are the elements of X, and the nodes of T are of two types : the *P-nodes* and the *Q-nodes*. We represent P-nodes by ellipses, and Q-nodes by rectangles.

On a P-node, one can apply any permutation of its children (equivalently, its children are not ordered). The children of a Q-node are ordered, and the only permutation we can apply on them is to reverse the order. For instance, the PQ-Tree of Figure 1 represents the set of permutations {(0,1,2,3,4,5), (0,1,3,2,4,5), (0,2,1,3,4,5), (0,2,3,1,4,5), (0,3,1,2,4,5), (0,3,2,1,4,5), (5,4,1,2,3,0), (5,4,1,3,2,0), (5,4,2,1,3,0), (5,4,2,3,1,0), (5,4,3,1,2,0), (5,4,3,2,1)}.

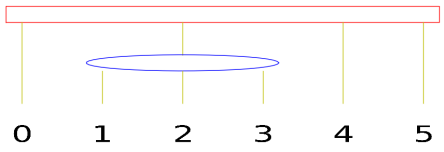

Figure 1: A PQ-Tree

Let $\mathcal{M}$ be a formal context matrix. An order σ on the lines of $\mathcal{M}$ is *compatible* if, when the lines of $\mathcal{M}$ are sorted along σ, on each column of $\mathcal{M}$, the 1's are consecutive. If $\mathcal{M}$ has the Consecutive One's Property (i.e. if there exist compatible orders), the set of compatible orders can be represented by a (unique) PQ-Tree. For instance, the cross-table of Figure 2 has the C1P and its compatible orders are represented by the PQ-Tree of Figure 1. Moreover the the formal concepts are intervals of all the compatible orders.

Given a Formal Context $\mathcal{M}$ satisfying the C1P, the associated PQ-Tree is a condensed representation of the associated concept lattice: if we add to $\mathcal{M}$ columns which are nonempty intersections or non-disjoint unions of already ex-

	0	1	2
0	×		
1	×	×	
2	×	×	
3	×	×	
4		×	×
5			×

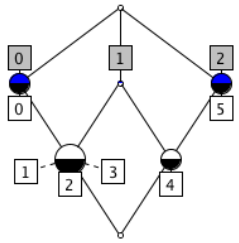

Figure 2: Example of cross table (left) and its associated lattice (right).

isting columns, the associated PQ-Tree remains unchanged. This is why we will use PQ-Trees as a representative of concept lattices to solve seriation problems.

Generally, a formal context does not have the C1P, as that associated with the cross table of Figure 3. We can remark that, although this example was built by adding columns to the example of Figure 2, it is not easy to construct the concept lattice of Figure 2 directly from the one of this example.

	0	1	2	3	4
0	×			×	
1	×	×			×
2	×	×			
3	×	×			×
4		×	×	×	×
5			×		

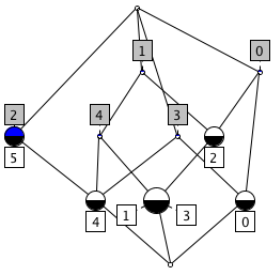

Figure 3: Extension of the Cross Table 2 (left) and its associated lattice (right).

A first way to solve the seriation problem consists in finding a maximal set of columns M' such that $\mathcal{M}_{|M'}$ has the C1P and exhibit the compatible orders. This is made possible by the incremental nature of the Booth and Lueker algorithm. In addition, starting with this maximal set M', it is easy to find the associated concepts: their extensions are the columns and the 2-intersections of columns (the intersection of three intervals is the intersection of two of them).

The algorithm of Booth and Lueker [5] relies on a function UPDATE_TREE(T, A), where T is a PQ-Tree on X and A a subset of X. UPDATE_TREE returns a PQ-Tree T' where $S_{T'}$ is the set of all permutations σ of S_T such that, when X is sorted along σ, A is an interval of X (if there is no such permutations, T' is None); for instance, with the PQ-Tree of Figure 1 and the set $\{1, 3, 4\}$ (column 4 of Figure 3), UPDATE_TREE returns the PQ-Tree of Figure 4. UPDATE_TREE runs in $O(n)$, where n is the size of the column (*i.e.* the number of lines of the

matrix). Given an $n \times m$ $\{0,1\}$-matrix $\mathcal{M}$, the algorithm of Booth and Lueker

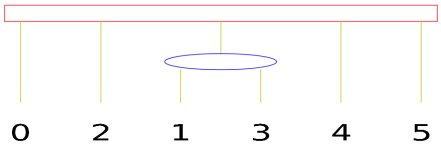

Figure 4: PQ-Tree built from columns 0, 1, 2 and 4 of Cross-Table of Figure 3

starts with the PQ-Tree U_n which represents all permutations on $\{1, \dots n\}$ (U_n has n leaves and one internal node (its root) which is a P-node) and apply UPDATE_TREE for all columns of $\mathcal{M}$. By this way, it determines if $\mathcal{M}$ has the C1P in $O(nm)$. More generally, given a subset S of 2^X, we can apply the algorithm of Booth and Lueker on S and obtain a PQ-Tree $T = \mathcal{BL}(S)$ such that, for any permutation σ represented by T (and only for them), when X is sorted along σ, all the elements of S are intervals.

So, given an order ζ on the columns, Algorithm MAXIMAL-C1P-CONSTRUCTION gives a solution to the approximate seriation problem in linear time. For the formal context of Figure 3, if we consider the columns in increasing order, this algorithm returns the PQ-Tree of Figure 4 and rejects the column 3, which is not compatible with the 3 first columns.

Algorithm MAXIMAL-C1P-CONSTRUCTION(M, ζ)
Input A $n \times m$ $\{0,1\}$-matrix M.
A permutation ζ on the columns of M.
Output A Maximal set C of columns of M such that $M_{|C}$ has C1P;
A PQ-Tree T representing the compatible permutations.
begin
  $T \leftarrow U_n$;
  $C \leftarrow \emptyset$;
  ForAll *columns* c *of* M *taken along* ζ **Do**
    $T' \leftarrow$ UPDATE_TREE(T, c) ;
    If $T' \neq$ **None Then**
      $T \leftarrow T'$;
      $C \leftarrow C \cup \{c\}$;
  return T, C ;
end

If the matrix has not the C1P, there are many solutions depending on the order ζ. For instance, if we consider the columns of the formal context of Figure 3 in reverse order, we keep the columns 4, 3, 1 and obtain the PQ-Tree of Figure 5.

We can see that the maximal sets of compatible columns do not have all the same number of elements. Since MAXIMAL-C1P-CONSTRUCTION is very efficient, it is possible to try many orders on the columns and then take the

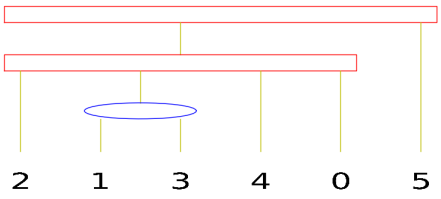

Figure 5: PQ-Tree built from columns 1, 3 and 4 of Cross-Table of Figure 3

greatest obtained set. We will now see that it is possible to go over that by using the lattice structure of PQ-Trees.

3 The Lattice Structure of PQ-Trees

We denote by $\mathcal{T}_X$ the set of all PQ-Trees on a finite set X. Given two elements T_1 and T_2 of $\mathcal{T}_X$, we say that $T_1 \leq T_2$ if $S_{T_1} \subseteq S_{T_2}$. We will show that $(\mathcal{T}_X, \leq)$ is a distributive lattice, which generalizes the semilattice of hierarchies (a hierarchy can be seen as a PQ-Tree with only P-Nodes). This will allow us to define a consensus between the different solutions of MAXIMAL-C1P-CONSTRUCTION.

Given a PQ-Tree T on X, the *Interval Set* of T (denoted by *Int(T)*) is the set of all nonempty subsets S of X such that, for every permutation σ compatible with T, when X is sorted along σ, S is an interval, *i.e.* *Int*(T) is the greatest subset P of $2^X \setminus \{\emptyset\}$ such that $\mathcal{BL}(P) = T$. Equivalently, $S \in$ *Int(T)* $\iff$ UPDATE_TREE$(T, S) = T$.

Let α be a node, we denote by $X(\alpha)$ the set of the leaves under α. If α is a Q-node with sons (in this order) $\beta_1, \ldots \beta_p$, we denote by $\widehat{X(\alpha)}$ the set $\{\bigcup_{k=i}^{j} X(\beta_k), 1 \leq i < j \leq p\}$ (remark that $\widehat{X(\alpha)}$ is a set of sets). We have:

Property 1. *Int*$(T) = \{X(\alpha), \alpha$ *node of* $T\} \cup \bigcup_{\substack{\alpha \text{ Q-node} \\ \text{of } T}} \widehat{X(\alpha)}$.

Proof. Let $I(T) = \{X(\alpha), \alpha \text{ node of } T\} \cup \bigcup_{\substack{\alpha \text{ Q-node} \\ \text{of T}}} \widehat{X(\alpha)}$. Clearly, for every permutation represented by T, all subsets of X in $I(T)$ are intervals. So $I(T) \subset$*Int*(T).

Conversely, let S be a subset of X not in $I(T)$. We are in one of the following cases:

1. $\exists$ node α s.t. $X(\alpha) \cap S \neq \emptyset$, $X(\alpha) \not\subset S$, $S \not\subset X(\alpha)$.
2. $\exists$ P-node α, with sons $\beta_1, \beta_2, ..., \beta_p$, $p > 2$ s.t. $X(\beta_1) \subset S$, $X(\beta_2) \subset S$, $X(\beta_p) \not\subset S$ (actually, if not in case 1, $X(\beta_p) \cap S = \emptyset$).

3. $\exists$ Q-node α, with sons $\beta_1, \beta_2, ..., \beta_p$, $p > 2$ s.t. $\exists i < j < k$ with $X(\beta_i) \subset S$, $X(\beta_k) \subset S$ and $X(\beta_j) \not\subset S$.

In each case, there exists a permutation σ in S_T such that, when X is sorted along σ, S is not an interval. □

Clearly:

Property 2. $T_1 \leq T_2 \iff Int(T_2) \subseteq Int(T_1)$.

Theorem 1. $(\mathcal{T}_X, \leq)$ *is a distributive lattice.*

Proof. By Property 2: $T_1 \wedge T_2 = \mathcal{BL}(Int(T_1) \cup Int(T_2))$ and $T_1 \vee T_2 = \mathcal{BL}(Int(T_1) \cap Int(T_2))$. □

In addition, by Property 1, $T_1 \vee T_2$ and $T_1 \wedge T_2$ can be computed in $O(n^3)$. Remark that, to compute $T_1 \wedge T_2$, we can use the sets $\{X(\beta_i) \cup X(\beta_{i+1}), 1 \leq i < p\}$ instead of $\widehat{X(\alpha)}$ for all Q-nodes α with sons $\beta_1, \ldots \beta_p$. Thus $T_1 \wedge T_2$ can be computed in $O(n^2)$.

The largest element of $(\mathcal{T}_X, \leq)$ is the universal tree $U_{|X|}$ which represents all the permutations on X and the smallest one is None which represents no permutation.

The join of the PQ-Trees of Figure 4 and 5 is represented on Figure 6. The PQ-Trees of Figure 4 and 5 represent respectively 4 and 8 permutations. Their join represents 16 permutations, which is very close to the theoretical minimum of 12, especially when compared to the 720 possible permutations on $\{0, \ldots, 5\}$. In addition, we can see that, for all the permutations represented by this PQ-Tree, the set $\{1, 2, 3, 4\}$ is ordered in $(2, 1, 3, 4), (2, 3, 1, 4), (4, 1, 3, 2)$ or $(4, 3, 1, 2)$, as for the two PQ-Trees of Figure 4 and 5.

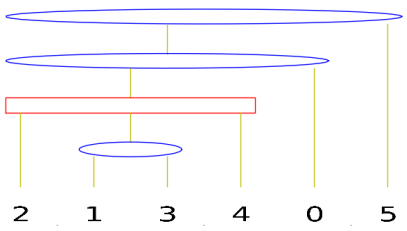

Figure 6: The join of the PQ-Trees of Figure 4 and 5

Conversely, the meet of the two PQ-Trees of Figure 4 and 5 is None, since these two PQ-Trees are compatible with maximal sets of columns. This situation will occur with any two PQ-Trees obtained with Maximal-C1P-Construction: they are built from maximal sets of columns of $\mathcal{M}$ and thus the permutation sets that they represent are already minimal.

We can now improve our algorithm by taking, from the best solutions obtained by MAXIMAL-C1P-CONSTRUCTION a consensus. More precisely:

Algorithm APPROXIMATE_SERIATION(M, κ)
Input A $n \times m$ $\{0,1\}$-matrix M.
A positive integer $\kappa < m$
A positive integer Nb_Trials
Output A PQ-Tree T representing the compatible permutations.
begin
  $E \leftarrow \emptyset$;
  For $i \leftarrow 1$ **To** Nb_Trials **Do**
    $\zeta \leftarrow$ *random permutation on* $\{1, \ldots, m$;
    $(T, C) \leftarrow$ MAXIMAL-C1P-CONSTRUCTION(M, ζ) ;
    If $Card(C) \geq \kappa$ **Then**
      $E \leftarrow E \cup \{T\}$;
  $//E = \{T_{i_1}, T_{i_2}, \ldots, T_{i_p}\}$
  return $T_{i_1} \vee T_{i_2} \vee \ldots \vee T_{i_p}$;
end

The result is a consensus of all "good" PQ-Trees, where "good" means that the PQ-Tree is built on at least κ columns of the matrix. The value of κ must be determined by the user and depends on the data. Our algorithm must be used in an interactive way. This is made possible by the efficiency of the algorithms. We will see that in the next section where we treat real archeological data.

4 Experimentations

We have experimented our method on a recent archeological data set which is shown in Table 1. The first step is to find maximal sets M' of columns such that the table induced by M' has the C1P. To do that, we have made 200 millions trials of MAXIMAL-C1P-CONSTRUCTION with random order of the columns. We found 1563 maximal sets, of size going from 5 to 11. Their distribution is shown in Table 2.

At this step, we made a consensus between all the solutions with maximum number of columns. We obtained the PQ-Tree of Figure 7.

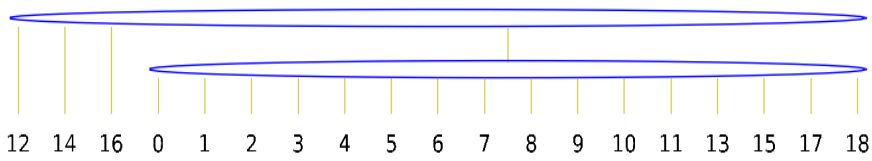

Figure 7: Consensus PQ-Tree between the twelve maximal PQ-Trees built on 11 columns.

Table 1: Cross table from Alberti[1]. The columns are indexed by object types and the lines by the huts of the Punta Milazzese (Aeolian Archipelago, Italy) settlement. We have indicated the presence/absence of objects in the different huts.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	×								×		×	×	×	×	×							×	×	×	×					×	
1	×		×			×	×		×		×	×		×	×		×	×		×	×	×		×	×	×	×				
2	×				×	×	×		×		×	×		×			×	×		×	×	×		×	×	×	×	×	×		
3					×	×	×		×			×	×	×			×			×		×		×	×				×	×	
4	×					×	×		×		×	×	×							×		×			×	×	×		×		
5						×			×			×								×		×		×			×			×	×
6	×		×			×	×		×			×		×		×	×	×		×		×			×	×	×				×
7	×	×			×		×		×	×	×	×	×	×			×	×	×	×	×	×		×	×	×	×		×		
8	×			×			×	×	×	×	×	×		×		×	×	×	×	×	×	×	×	×	×	×	×			×	
9	×	×			×	×	×	×	×			×		×			×			×	×	×			×		×			×	
10	×				×	×	×	×	×	×		×									×	×			×		×				
11	×	×				×	×		×	×	×	×		×	×	×	×	×		×		×		×			×			×	
12	×				×					×		×	×	×			×			×					×						
13	×					×	×	×	×			×		×			×			×	×	×								×	×
14					×									×											×						
15							×		×					×					×			×			×		×			×	
16	×						×			×																				×	
17						×	×		×			×	×				×			×					×						
18	×					×			×	×		×		×					×	×	×	×		×	×					×	

Table 2: Size and number of maximal sets of columns from Table 1 having the C1P.

Number of columns	5	6	7	8	9	10	11
Number of maximal sets	1	28	294	505	514	209	12

This PQ-Tree takes into account 22 columns, but it represents too many permutations (informally speaking, the corresponding consensus is too "soft"). In addition, since it corresponds to a consensus, we cannot build the associated lattice, but all the possible concepts are intervals of the PQ-Tree.

We can remark that all the lines/huts are grouped together except lines 12, 14 and 16. So we put these lines appart from the others (technically, we filled them with 0) and we apply the procedure on the transformed table. We get the results of Table 3.

Table 3: Size and number of maximal sets of columns from Table 1 without columns 12, 14 and 16 having the C1P.

Number of columns	7	8	9	10	11	12	Total
Number of maximal sets	3	142	480	579	144	1	1349

The PQ-Tree built on 12 columns (the columns 2, 3, 4, 8, 10, 11, 14, 21, 22, 27, 28 and 30), and all lines except the lines 12, 14 and 16, is shown on Figure 8. Since it is unique, it corresponds to a maximal sub-context (having C1P) of Table 1, whose concept lattice is shown on Figure 9.

It is possible to go further that solution and reiterate the processus. Since

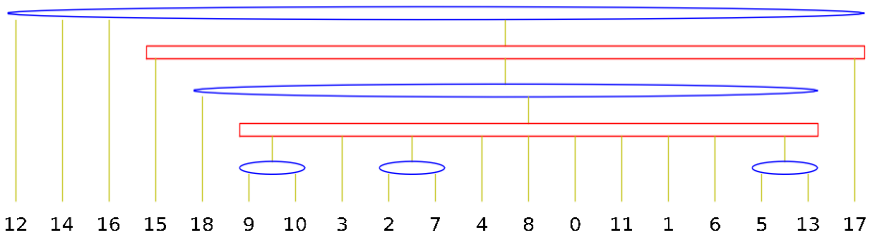

Figure 8: The PQ-Tree built on 12 lines by putting appart columns 12, 14 and 16

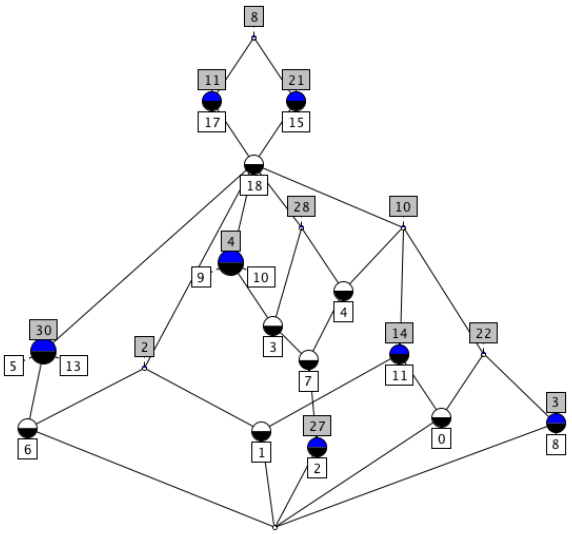

Figure 9: Concept lattice from the PQ-Tree of Figure 8

their is only one maximal column set of size 12 having C1P, we take the join of all PQ-Trees built on column sets of size ≥ 11. This PQ-tree is similar to the one of Figure 7. After putting appart lines 15 and 17, we get the results of Table 4. The consensus of the PQ-Trees corresponding with sets of size 13 is the one of Figure 10. This PQ-Tree takes into account 22 columns (the columns 0, 1, 2, 3, 6, 7, 8, 9, 11, 12, 13, 15, 16, 17, 18, 19, 21, 25, 26, 27, 28 and 29) and represents 2,985,984,000 permutations, which is 167,000 times less that the PQ-Tree of Figure 7.

We have seen that one can associate, with each formal context, maximal

Table 4: Size and number of maximal sets of columns from Table 1 without columns 12, 14, 15, 16 and 17 having the C1P.

Number of columns	8	9	10	11	12	13	Total
Number of maximal sets	1	68	405	377	90	10	951

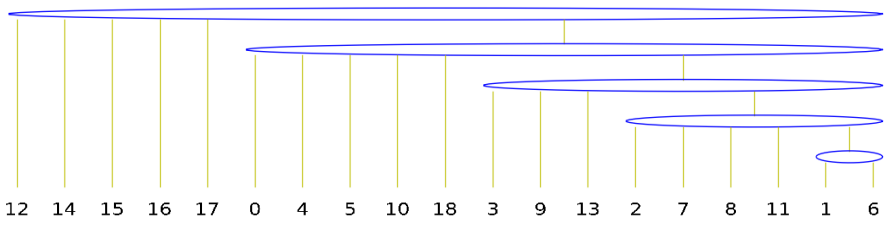

Figure 10: PQ-Tree obtained by putting appart Lines 12, 14, 15, 16 and 17.

(in lines and columns) sub-contexts satisfying the C1P, that we could name *Seriation Formal Concepts*. The intersection of two seriation formal concepts $\mathcal{C}_1$ and $\mathcal{C}_2$ is a seriation formal concept (its PQ-Tree is the meet of the two PQ-Trees associated with $\mathcal{C}_1$ and $\mathcal{C}_2$). So, with any formal context, we can associate the semi-lattice of its seriation formal concepts. At the present time, we are able to determine all the seriation formal concepts containing a given set of lines. We are working on an algorithm which computes all the seriation formal concepts and generates the seriation formal lattice.

5 Conclusion

We have presented in this paper an interactive framework to solve the approximate seriation problem for formal contexts. It is algorithmically efficient and uses the underlying lattice structure of PQ-Trees.

As usual in approximation problems, we are dealing with several criteria which are important to determine the quality of the resulting PQ-Tree:

- The number of columns taken into account (the largest possible).
- The number of removed lines (the smallest possible).
- The number of represented permutations (the smallest possible)

If a formal context admits an exact solution to the seriation problem, then its underlying structure can be represented by a PQ-Tree. If it is not the case, we can build a consensus PQ-Tree which is a solution to the approximate seriation problem but at the present time, we are not able to build an associated concept

lattice. Moreover, from this work appears the new notion of seriation formal concepts and semilattices.

References

[1] G. Alberti (2013), "Making Sense of Contingency Tables in Archeology: the Aid of Correspondence Analysis to Intra-Site Activity Areas Research", *Journal of Data Science* 11, 479–499.

[2] S. Benzer (1962), "The Fine Structure of the Gene", *Scientific American* 206:1, 70–84.

[3] M.W. Berry, B. Hendrickson and P. Raghavan (1996), "Sparse Matrix Reordering Schemes for Browsing Hypertext", in *The Mathematics of Numerical Analysis*, J. Renegar, M. Shub and S. Smale Eds., pp 99–123, American Mathematical Society, Lectures in Applied Mathematics.

[4] L. Boneva (1980), "Seriation with Applications in Philology", in *Mathematical Statistics*, R. Bartoszyński, J. Koronacki and R. Zieliński Eds., pp 73–82, PWN Polish Scientific Publishers.

[5] K.S. Booth and G.S. Lueker (1976), "Testing for the Consecutive Ones Property, Interval Graphs and Graph Planarity Using PQ-Tree Algorithm", *Journal of Computer and System Sciences* 13, 335–379.

[6] G. Caraux and S. Pinloche (2005), "PermutMatrix: a Graphical Environment to Arrange Gene Expression Profiles in Optimal Linear Order", *Bioinformatics* 21, 1280–1281.

[7] C.H. Chen, H.G. Hwu, W.J. Jang, C.H. Kao, Y.J. Tien, S. Tzeng and H.M. Wu (2004), "Matrix Visualisation and Information Mining", *COMPSTAT'2004*, Prague.

[8] D. Halperin (1994), "Musical Chronology by Seriation", *Computer and the Humanities* 28, 13–18.

[9] W.M.F. Petrie (1899), "Sequences in Prehistoric Remains", *Journal of the Anthropological Institute of Great Britain and Ireland* 29, 295–301.

[10] A. Strehl and J. Ghosh (2003), "Relationship-Based Clustering and Visualization for High-Dimensional DataMining", *INFORMS Journal on Computing* 15, 208–230.