Approximate Seriation in Formal Concept Analysis

François Brucker and Pascal Préa

École Centrale Marseille,
LIF, Laboratoire d'Informatique Fondamentale de Marseille,
CNRS UMR 7279,
Marseille, France,
francois.brucker@lif.univ-mrs.fr, pascal.prea@lif.univ-mrs.fr

Abstract. In this paper we present a method (linear in the size of the formal context) 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.

Keywords: 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. More generally, the seriation problem consists in finding a linear order which underlies a data set, and this problem arises in genetics [2, 6], hypertext browsing [3], philology [4], data visualization [7, 10], musicology [8], ...; the order can be time, altitude, dispersion along a river, influence of an author, or even an unknown reason.

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

In *exact seriation*, there exists a linear order (said *compatible*) such that A is an interval for any formal concept (A, B). The problem is to find one (or all the) compatible orders. Since the intersection of two intervals is also an interval, it is sufficient to check the inf-irreducible elements of the associated concept lattice, *i.e.* the columns of the formal context matrix. In this case, the formal context matrix $\mathcal{M}$ is said to have the *Consecutive One's Property* (*C1P*); that is the lines of $\mathcal{M}$ can be reordered in such a way that on every column of $\mathcal{M}$, the *1*s appear in consecutive order. Note tat the C1P is not a symmetrical property. Indeed, *e.g.* for the classical problem in archeology, the objects organize the sites into a linear order (the time), but the converse is false.

An optimal algorithm to find all the compatible orders was introduced in 1976 [5], based on a special data structure: *PQ-Trees*.

In *approximate seriation*, the data is not accurate (*e.g.* a site may not contain an object that was used when it was built); and the formal context matrix $\mathcal{M}$ does not have the C1P. We suppose (for extra reasons) that there exists a linear order which underlies the data set and the problem is to find one (or several) such order. There are two basic approaches for approximate seriation: *(i)* Minimally modify $\mathcal{M}$ so that it has the C1P (this yields to an NP-Hard problem); *(ii)* Find a maximal submatrix which has the C1P. In the formal concept framework, this approach consists in finding a sub-lattice of the formal concept lattice.

The aim of this paper is, following approach (*ii*), 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 which follows approach *(ii)*. 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 which constructs a consensus between the possibly multiple solutions of the algorithm given in Section 2. In Section 4, we present a possible workflow of our method on an archeological data set. Actually, this workflow makes several runs of the algorithm of Section 3. The inputs of these runs strongly depends on the data and thus has to be decided by the user.

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)}.

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 all 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.

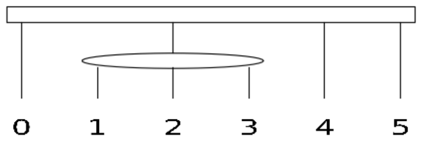


Fig. 1. A PQ-Tree

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

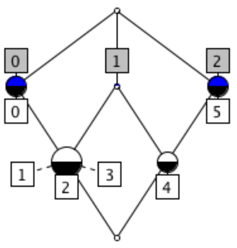


Fig. 2. Example of cross table (left) and its associated lattice (right).

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 existing 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 Figure 3.

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

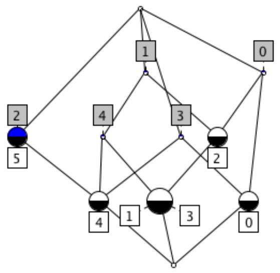


Fig. 3. Extension of the Cross Table 2 (left) and its associated lattice (right).

A first way to solve the approximate seriation problem consists in finding a maximal set of columns M' such that $\mathcal{M}_{|M'}$ has the C1P and exhibit the compatible orders. There exist several maximal sets of attributes having C1P; for instance, the cross-table of Figure 3 admits $\{0, 1, 2, 4\}$ (see Figure 4) or $\{1, 3, 4\}$ (see Figure 6). Remark that the concept lattices of Figures 4 and 6 are sublattices of the one of Figure 3. In addition, for each concept (A, B), A is an interval for any compatible order.

This is made possible by the incremental nature of the Booth and Lueker algorithm, which considers one column of the matrix at each step. 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 Booth and Lueker algorithm [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

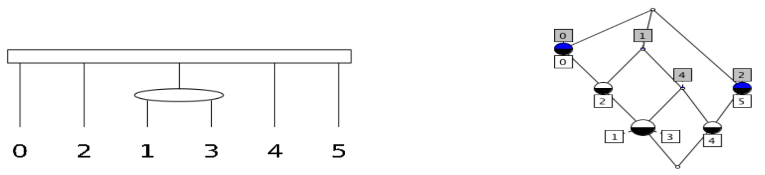


Fig. 4. PQ-Tree built from columns 0, 1, 2 and 4 of Cross-Table of Figure 3 (left) and the associated concept lattice (right).

starts with the PQ-Tree U_n which represents all permutations on $\{1, \ldots n\}$ (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 $O(nm)$. Fo instance, applying this algorithm on the cross table of Figure 2, the algorithm runs as on Figure 5.

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 a column order ζ, Algorithm Maximal-C1P-Construction gives a solution to the approximate seriation problem in linear time. For the formal

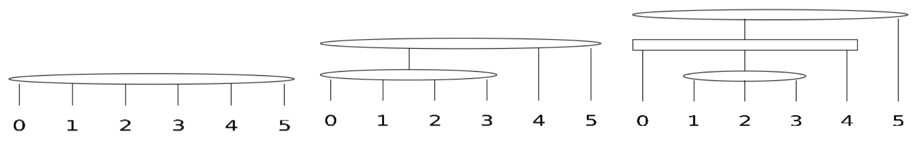
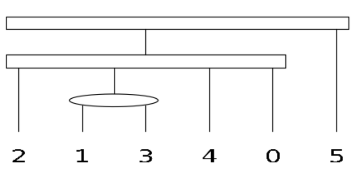


Fig. 5. The intermediate steps of the algorithm of Booth and Lueker, when applied on the cross table of Figure 2: the algorithm starts with the universal tree U_6 (left) and treats the set $\{0, 1, 2, 3\}$ (*i.e.* it forces $\{0, 1, 2, 3\}$ to be an interval); it then gets the PQ-Tree in the middle. Then it treats the set $\{1, 2, 3, 4\}$ and gets the PQ-Tree on the right. By treating the set $\{4, 5\}$, it gets the PQ-Tree of Figure 1.

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($M, \zeta$)
**Input** A $n \times m$ $\{0, 1\}$-matrix $M$.
        A permutation $\zeta$ 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 $\zeta$ Do
        $T' \leftarrow$ UPDATE_TREE($T, c$) ;
        If $T' \neq$ None Then
            $T \leftarrow T'$ ;
            $C \leftarrow C \cup \{c\}$ ;
    return $T, C$ ;
end
```



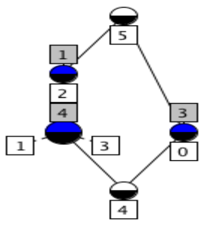


Fig. 6. PQ-Tree built from columns 1, 3 and 4 of Cross-Table of Figure 3 (left) and the associated concept lattice (right).

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 6.

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 greatest obtained set. The problem is that there may exist many such sets. We will see in next Section that it is possible to go over that by using the lattice structure of PQ-Trees.

3 The Lattice Structure of PQ-Trees

We will show here that the PQ-Trees on a set X can be organized as a distributive lattice. This will allow us to build a consensus (by taking the join) of several PQ-Trees given by Algorithm MAXIMAL_C1P_CONSTRUCTION. For instance, the PQ-Tree of Figure 7 is the join of the PQ-Trees of Figures 4 and 6.

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) $\beta_1, \dots \beta_p$, 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 $Int(T) = \{X(\alpha), \alpha \textit{ node of } T\} \cup \bigcup_{\substack{\alpha \textit{ Q-node} \\ \textit{of } T}} \widehat{X(\alpha)}.$

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 $X(\beta_j) \not\subset S$.

Suppose that there exists an ordering $\sigma = (x_1, x_2, \dots x_n)$ of X, compatible with T, such that S is a proper interval $x_i, \dots x_j$ of X. In Case 1, we can suppose that $X(\alpha) = \{x_k, \dots x_l\}$, with $i < k \leq j < l$. By reversing $X(\alpha)$, we get a compatible ordering of X for which S is not an interval. In Case 2, we can suppose that $X(\beta_1) = \{x_k, \dots x_{k'}\}$, $X(\beta_3) = \{x_l, \dots x_{l'}\}$ and $X(\beta_3) = \{x_m, \dots x_{m'}\}$, with $i \leq k \leq k' < l \leq l' \leq j < m \leq m'$. By "exchanging" β_2 and β_3, we get a compatible permutation for which X is not an interval.

In Case 3, let $x \in X(\beta_i)$, $y \in X(\beta_j) \setminus S$ and $x \in X(\beta_k)$. For any compatible ordering of X, $x < y < z$, and thus X 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, if T_1, T_2 and T_3 are PQ-Trees, $Int(T_1) \cup (Int(T_2) \cap Int(T_3)) = (Int(T_1) \cap Int(T_2)) \cup (Int(T_1) \cap Int(T_3))$, *i.e.* $T_1 \wedge (T_2 \vee T_3) = (T_1 \wedge T_2) \vee (T_1 \wedge T_3)$; so $(\mathcal{T}_X, \leq)$ is a distributive lattice. □

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 $\beta_1, \dots \beta_p$. 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 6 is represented on Figure 7. The PQ-Trees of Figure 4 and 6 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 $\{0, \dots, 5\}$. 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 6.

Conversely, the meet of the two PQ-Trees of Figure 4 and 6 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.

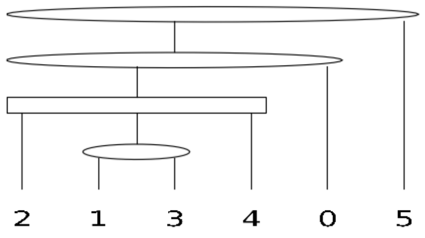


Fig. 7. The join of the PQ-Trees of Figure 4 and 6

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

```
Algorithm APPROXIMATE_SERIATION($M, \kappa$)
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.
          The set $\mathcal{C}$ of columns which have been taken into account.
begin
    $E \leftarrow \emptyset$ ;
    $\mathcal{C} \leftarrow \emptyset$ ;
    For $i \leftarrow 1$ To $Nb\_Trials$ Do
        $\zeta \leftarrow$ random permutation on $\{1, \ldots, m$ ;
        $(T, C) \leftarrow$ MAXIMAL-C1P-CONSTRUCTION($M, \zeta$) ;
        If $Card(C) \geq \kappa$ Then
            $E \leftarrow E \cup \{T\}$ ;
            $\mathcal{C} \leftarrow \mathcal{C} \cup C$ ;
    $//E = \{T_{i_1}, T_{i_2}, \ldots, T_{i_p}\}$
    return $T_{i_1} \vee T_{i_2} \vee \ldots \vee T_{i_p}$, $\mathcal{C}$;
end
```

This algorithm runs in $O(Nb\_Trials \times n \cdot m + p \cdot n^3)$, where p is the number of column sets of size $\geq \kappa$ having the C1P. 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. In next Section, we will apply our algorithm on a real data set and we will see how to choose κ. Moreover, we will see that the use of the algorithm may need other actions of the user.

4 Experimentations

We have experimented our method on a recent archeological data set which is shown in Table 1. This example shows an application where the seriation problem consists in finding trends in the data. In the original paper, the author uses principal component analysis to do that. Our method allows to achieve similar results for binary data, with PQ-Trees replacing principal axes.

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.[1] We found 1563 maximal sets, of size going from 5 to 11. Their distribution is shown in Table 2.

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

At this step, we made a consensus between all the solutions with maximum number of columns, *i.e.* we make APPROXIMATE_SERIATION run with $\kappa = 11$. We obtained the PQ-Tree of Figure 8.

[1] 200 millions is very small when compared to the 31! possible orders of the column set, but actually, we made 20 series of 10 millions of trials. For each of these 20 series, no maximal set M' has been found after trial 50,000. In addition, the maximal sets were the same for the 20 series. It is thus reasonable to suppose that we have found all the maximal sets M'.

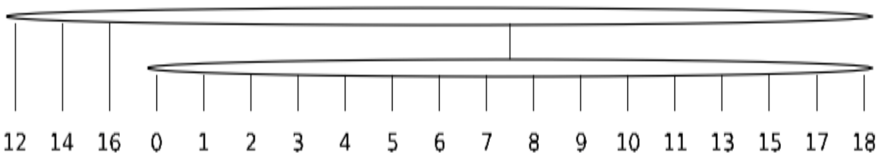


Fig. 8. Consensus PQ-Tree between the twelve maximal PQ-Trees built on 11 columns.

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 determine the maximal sets of columns of the transformed table which have the C1P (in exactly the same way that for the complete table). We get the results of Table 3.

Table 3. Size and number of maximal sets of columns from Table 1 without lines 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 the greatest maximal set of 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 9. Since it is unique, it corresponds to a maximal sub-context (having C1P) of Table 1, whose concept lattice is shown on Figure 10.

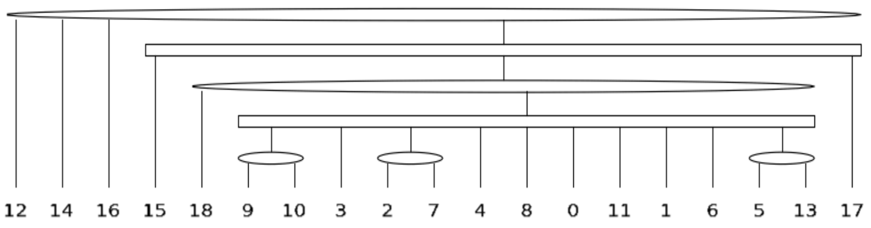


Fig. 9. The PQ-Tree built on 12 columns by putting appart lines 12, 14 and 16

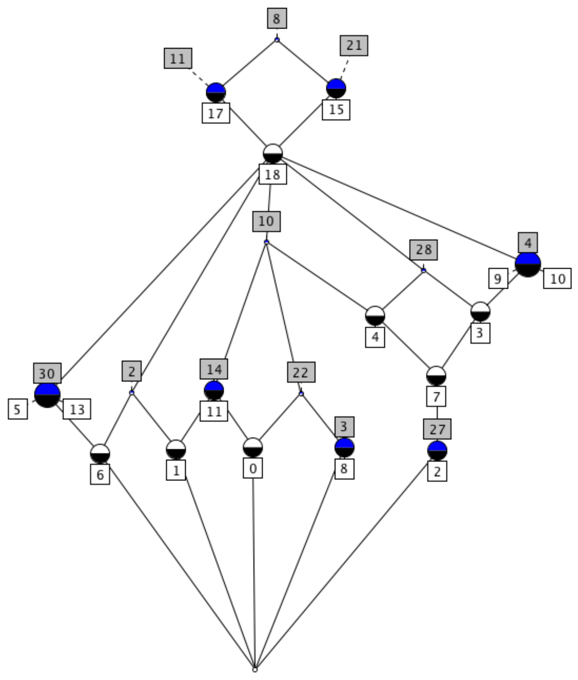


Fig. 10. Concept lattice from the PQ-Tree of Figure 9

This PQ-Tree takes into account only 12 columns, but it is possible to go further that solution. Since 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 and we get the PQ-Tree of Figure 11. This PQ-Tree represents a lot of permutations, but we can see that lines 15 and 17 are appart from the others. So we put them appart and we get the results of Table 4. The consensus of the PQ-Trees corresponding with sets of size 13 is the one of Figure 12. 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 8.

Table 4. Size and number of maximal sets of columns from Table 1 without lines 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

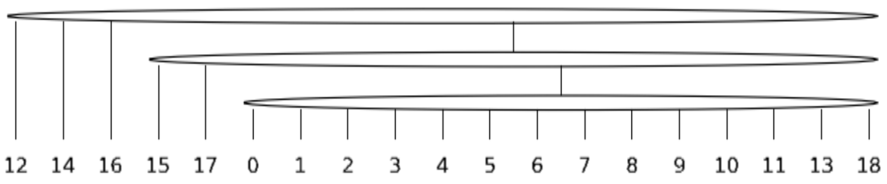


Fig. 11. Consensus PQ-Tree between the maximal PQ-Trees built on more than11 columns with lines 12, 14 and 16 appart.

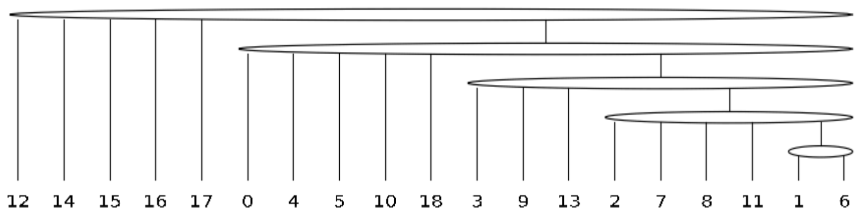


Fig. 12. Consensus PQ-Tree between the PQ-Trees built on 13 columns with lines 12, 14, 15, 16 and 17 appart.

We have seen that one can associate, with each formal context, maximal (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. More precisely, this framework uses some runs (3 for our example) of Approximate_Seriation, which is in $O(Nb\_Trials \times n \cdot m + p \cdot n^3)$. We have used a very high value for Nb_Trials (up to 200 Millions, but 50,000 trials would have yield the same result). Moreover, since only the large column sets having the C1P are interesting for us, a small number of trials would have been sufficient (≈ 1000 in our case).

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)
- The possibility to build a concept lattice from the solution.

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. Alberti G.: 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 (2013).
2. Benzer S.: The Fine Structure of the Gene. Scientific American 206:1, 70–84 (1962).
3. Berry M.W., Hendrickson B., Raghavan P.: Sparse Matrix Reordering Schemes for Browsing Hypertext. In The Mathematics of Numerical Analysis, J. Renegar, M. Shub and S. Smale Eds., 99–123, American Mathematical Society, Lectures in Applied Mathematics (1996).
4. Boneva L.: Seriation with Applications in Philology. In Mathematical Statistics, R. Bartoszyński, J. Koronacki and R. Zieliński Eds., 73–82, PWN Polish Scientific Publishers (1980).
5. Booth K.S., Lueker G.S.: Testing for the Consecutive Ones Property, Interval Graphs and Graph Planarity Using PQ-Tree Algorithm. Journal of Computer and System Sciences 13, 335–379 (1976).
6. Caraux G., Pinloche S.: PermutMatrix: a Graphical Environment to Arrange Gene Expression Profiles in Optimal Linear Order. Bioinformatics 21, 1280–1281 (2005).
7. Chen C.H., Hwu H.G., Jang W.J., Kao C.H., Tien Y.J., Tzeng S., Wu H.M.: Matrix Visualisation and Information Mining. COMPSTAT'2004, Prague (2004).
8. Halperin, D.: Musical Chronology by Seriation. Computer and the Humanities 28, 13–18 (1994).
9. Petrie, W.M.F.: Sequences in Prehistoric Remains. Journal of the Anthropological Institute of Great Britain and Ireland 29, 295–301 (1899).
10. Strehl, A., Ghosh, J.: Relationship-Based Clustering and Visualization for High-Dimensional Data Mining. INFORMS Journal on Computing 15, 208–230 (2003).