A closure algorithm using a recursive decomposition of the set of Moore co-families

Pierre Colomb[1], Alexis Irlande[2], Olivier Raynaud[1], Yoan Renaud[3]

[1] Clermont Université, Université Blaise Pascal, Campus des Cézeaux
63173 Clermont-Ferrand, France
`pierre@colomb.me`, `raynaud@isima.fr`
[2] Universidad Nacional de Colombia
Bogota, Colombia
`irlande@lirmm.fr`
[3] INSA de Lyon, Bâtiment Blaise Pascal
Campus de la Doua, 69621 Villeurbanne, France
`yoan.renaud@insa-lyon.fr`

Abstract. Given a set $U_n = \{1, ..., n\}$, a collection $\mathcal{M}$ of subsets of U_n that is closed under intersection and contains U_n is known as a Moore family. The set of Moore families for a fixed n is in bijection with the set of Moore co-families (union-closed families containing the empty set) denoted itself $\mathbb{M}_n$. This paper follows the work initiated in [8] and [9] about the recursive decomposition of the lattice of Moore co-families. We first show that each Moore co-family can be represented by a decomposition tree and we use this principle to design an original algorithm to close under union any given family. Then we discuss about the time complexity and give some experimental results.

1 Introduction

The concept of collection of sets on a ground set U_n closed under intersection appears with different names depending of the scientific fields. The name 'Moore families' was first used by Birkhoff in [4] referring to E.H. Moore's researches. But, very frequently, such a collection is called 'closure spaces' (or 'closure systems') or 'convexity spaces'. This concept is applied in numerous fields in pure or applied mathematics and computer science. For mathematical researches in algebra and topology we can cite [7, 19, 20]. For researches in order and lattice theory we have to cite [4, 10] for their works on closure operators. Formally a closure operator is an extensive, isotone and idempotent function on 2^{U_n}, and a closure system is then the sets of its fixed points. In particular it is shown that any closure system is a complete lattice. From 1937 Birkhoff in [3] gave a compact representation of 'quasi ordinal spaces' (in other words of collections closed by intersection and union also called distributive lattices). More recently the collection appears again as the main concept in computer science with researches in relational databases ([11]), in data analysis and formal concept analysis ([13, 1, 15]). More precisely, Ganter and Wille define a mathematical framework for classification and Barbut and Monjardet used the Galois lattice for equivalent tasks.

In the same time, in 1985 ([12]) such collections are called 'knowledge spaces' by Doignon et Falmagne. An important fact is that the collection of Moore families on U_n is itself a Moore family or a closure system. Indeed, the system composed of Moore families contains one maximum element (2^{U_n}, all subsets of U_n) and the intersection of two Moore families is a Moore family itself. To get an overall view of the properties of this closure system, see the survey of Caspard and Monjardet [6].

Previously in the introduction, we have defined a Moore family on U_n as a collection of sets containing U_n and closed by intersection. But for reasons of legibility of the results and simplification of expressions, this article deals with the set of families closed by union and containing the empty set called Moore co-families and denoted by $\mathbb{M}_n$. Basically, the set of Moore families is in bijection with this set. For a given Moore family, one only has to complement every set to obtain a Moore co-family. For example, the Moore family $\{\{1\},\{1,2\},\{1,3\},\{1,2,3\}\}$ on U_3 corresponds to the Moore co-family $\{\emptyset,\{2\},\{3\},\{2,3\}\}$ and vice versa.

In [8], Colomb & al counted the exact number of Moore co-families for $n = 7$ and stated a recursive decomposition theorem of the lattice of Moore co-families in [9]. In the same article authors state that the set $\mathbb{M}_n$ is endowed with the quotient partition associated with the operator h (each class of the partition contains all the families which have the same image by h) and prove that each class has a distributive lattice structure. This operator h is the main concept underlying the recursive description of $\mathbb{M}_n$. More recently in [2] authors give a complete characterization of the set $h(\mathcal{M}) \setminus \mathcal{M}$.

In the present article we pursue investigations involving Moore co-families by using the recursive decomposition theorem. In the third section we describe succinctly the theorem and we show that each Moore co-family can be represented by a decomposition tree. In the fourth section we describe an original algorithm to generate a Moore co-family from the set of its join-irreducible elements. Then we discuss about the time complexity of the process. Some experimental results are given in section five.

In the rest of the paper, we denote elements by numbers $(1, 2, 3, \dots)$. Sets are denoted by capital letters $(A, B, C, \dots)$. Families of sets are denoted by calligraphic letters $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots)$. Finally, we denote the sets of families of sets by black board letters $(\mathbb{A}, \mathbb{B}, \mathbb{C}, \dots)$.

2 Definitions and notations

For any integer $n \geq 1$, let U_n denote the set $\{1, \dots, n\}$. Let $\mathcal{M}$ be a family, we note $(\mathcal{M}, \subseteq)$ the corresponding partial order. Two sets M, M' in $\mathcal{M}$ such that neither $M \subseteq M'$ and $M' \subseteq M$ are said *incomparable* in $(\mathcal{M}, \subseteq)$. A family where every pair of sets is incomparable is called an antichain. We say that M is covered by M' in $(\mathcal{M}, \subseteq)$, denoted $M \prec M'$, if $M \subset M'$ and there is not $M" \in \mathcal{M}$ such that $M \subset M"$ and $M" \subset M'$.

Given a family $\mathcal{M}$, a subfamily $\mathcal{I}$ of $\mathcal{M}$ is an *ideal* of $\mathcal{M}$ if it satisfies the following implication for any pair M and M' in $\mathcal{M}$,

$$M \subseteq M' \text{ and } M' \in \mathcal{I} \Rightarrow M \in \mathcal{I}.$$

In other words, an ideal of $\mathcal{M}$ is some antichain of $\mathcal{M}$ and everything below it. We shall use $\mathbb{I}_{\mathcal{M}}$ to denote the sets of ideals on $\mathcal{M}$. Given a set X in a family $\mathcal{M}$, there exists a unique ideal $\mathcal{I}$ of $\mathcal{M}$ with X as a maximum set. Let $\mathcal{I}_{\mathcal{M}}(X)$ denote this ideal. A set $J \in \mathcal{M}$ is called a join-irreducible if J covers only one set. The set of all join-irreducible sets of $\mathcal{M}$ is denoted $\mathcal{J}_{\mathcal{M}}$. Let $\mathcal{M}$ be a Moore co-family and x an element, we denote $\mathcal{M}_x$ (resp. $\mathcal{M}_{\bar{x}}$) the restriction of $\mathcal{M}$ to its sets containing x (resp. to its sets not containing x). The empty set is added to $\mathcal{M}_x$. By extension, we denote by $\mathcal{J}_x$ (resp. $\mathcal{J}_{\bar{x}}$) the set of join-irreducible elements of $\mathcal{M}$ which contain the element x (resp. which do not contain the element x).

3 Recursive decomposition of the Moore co-families lattice

In previous work, Colomb & al. gave a recursive definition of the Moore co-families on U_n, for any $n \geq 1$ (the only Moore co-family on $U_0 = \emptyset$, is $\{\emptyset\}$).

Definition 1. *For any integer n such that $n \geq 1$, we define*

$$\begin{array}{lll} g_{1,n} : \mathbb{M}_{n-1} & \longrightarrow & \mathbb{M}_n \\ \qquad \mathcal{M} & \longmapsto & \{X \in 2^{U_n} \mid \exists M \in \mathcal{M} \setminus \{\emptyset\} \textit{such that } X = M \cup \{n\}\} \cup \{\emptyset\}, \\ g_{2,n} : \mathbb{M}_{n-1} & \longrightarrow & \mathbb{M}_n \\ \qquad \mathcal{M} & \longmapsto & \{X \in 2^{U_n} \mid \exists M \in \mathcal{M} \textit{ such that } X = M \cup \{n\}\} \cup \{\emptyset\}, \\ h_n : \mathbb{M}_{n-1} & \longrightarrow & \mathbb{M}_{n-1} \\ \qquad \mathcal{M} & \longmapsto & \{X \in 2^{U_{n-1}} \mid \forall M \in g_{1,n}(\mathcal{M}) \setminus \{\emptyset\},\ X \cup M \in g_{1,n}(\mathcal{M})\}. \end{array}$$

We may use h, g_1 and g_2 instead of $h_n, g_{1,n}$ and $g_{2,n}$ when no confusion is possible. Function g_2 consists in adding element n to every set of a family $\mathcal{M}$ in $\mathbb{M}_{n-1}$ (thus including the singleton $\{n\}$) plus the empty set. Function g_1 behaves similarly but removes the singleton $\{n\}$. With these functions defined, we can state Theorem 1, giving a description of $\mathbb{M}_n$ with respect to $\mathbb{M}_{n-1}$.

Theorem 1 (Colomb & al. [9]).
For any integer n such that $n \geq 1$,

$$\begin{aligned} \mathbb{M}_n = & \textstyle\bigcup_{\mathcal{M} \in \mathbb{M}_{n-1}} \{g_1(\mathcal{M}) \cup \mathcal{M}' \mid \mathcal{M}' \in \mathcal{I}_{\mathbb{M}_{n-1}}(h(\mathcal{M}))\} \cup \\ & \textstyle\bigcup_{\mathcal{M} \in \mathbb{M}_{n-1}} \{g_2(\mathcal{M}) \cup \mathcal{M}'' \mid \mathcal{M}'' \in \mathcal{I}_{\mathbb{M}_{n-1}}(\mathcal{M})\} \end{aligned}$$

In other words there exists a natural bi-partition of $\mathbb{M}_n$. Families not containing the singleton $\{n\}$ under the form $g_1(\mathcal{M}) \cup \mathcal{M}'$ with $\mathcal{M}'$ a sub-Moore co-family of $h(\mathcal{M})$ and families containing $\{n\}$ under the form $g_2(\mathcal{M}) \cup \mathcal{M}''$ with $\mathcal{M}''$ a sub-Moore co-family of $\mathcal{M}$.

3.1 Decomposition tree of a Moore co-family

Another interpretation is that for any element x in U_n and any Moore co-family $\mathcal{M}$ in $\mathbb{M}_n$, $\mathcal{M}$ can be written $\mathcal{M}_{\bar{x}} \cup g_{i,x}(\mathcal{M}')$ with $\mathcal{M}'$ such that $g_{i,x}(\mathcal{M}')$ corresponds to $\mathcal{M}_x$. By applying recursively this decomposition principle to $\mathcal{M}$ we obtain a decomposition binary tree of $\mathcal{M}$ (each leaf is an empty set which cannot be further decomposed). Basically each leaf corresponds to a set of the initial family $\mathcal{M}$: to obtain the set corresponding to a leaf, one has only to apply iteratively the different functions $g_{1,x}$ or $g_{2,x}$ that can be found in the path from the chosen leaf to the root of the tree. An example of decomposition of Moore co-family is given in Figure 1.

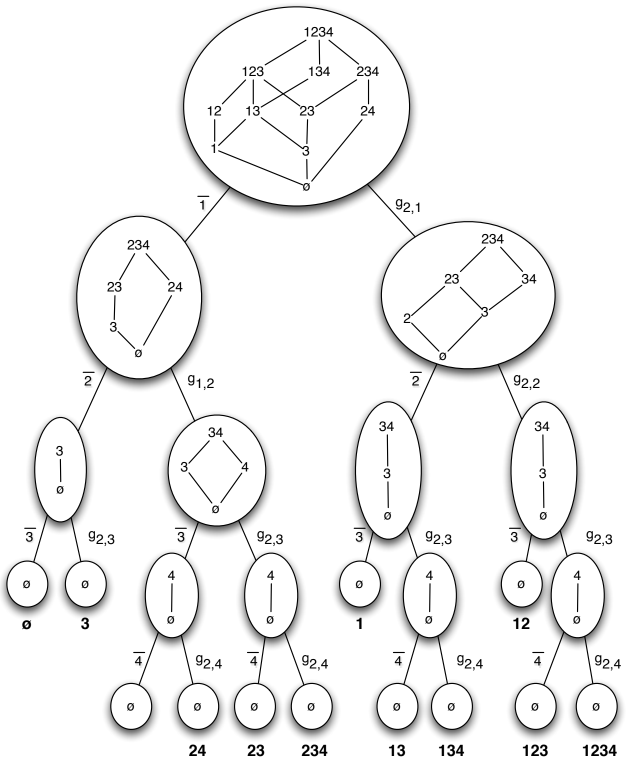

Fig. 1. Decomposition of a Moore co-family

4 A new algorithm to compute a union-closed family

In this section, we present an algorithm to generate a Moore co-family $\mathcal{M}$ from its set of join-irreducible elements denoted $\mathcal{J}_{\mathcal{M}}$. This algorithm is based on the theorem 1.

4.1 Closure under union algorithm

Problem of generation of closed-sets from a representative family has been well-studied these last years. Most of existing algorithms are based on a decomposition strategy [14, 5, 17, 16]. As previously stated, generation of closed-sets and co-closed sets can be treated in the same way (i.e. one only have to complement every sets of input and output families to obtain both closed and co-closed families). The question we ask here is, given a family of sets $\mathcal{J}$, how to compute its associated Moore co-family $\mathcal{M}$, i.e. the smallest Moore co-family that contains all sets of $\mathcal{J}$. We choose to denote $\mathcal{J}$ the given set because the smallest representative family for a Moore co-family is the family of its join-irreducible sets.

We propose an algorithm based on the decomposition of a Moore co-family that only uses families of join-irreducible sets. In other words, for each step of the recursive decomposition, given a local set $\mathcal{J}_{\mathcal{M}}$, we compute families of join-irreducible sets $\mathcal{J}_{\mathcal{M}_{\bar{x}}}$ and $\mathcal{J}_{\mathcal{M}'}$ corresponding to $\mathcal{M}_{\bar{x}}$ and $\mathcal{M}'$ such that $\mathcal{M} = \mathcal{M}_{\bar{x}} \cup g_i(\mathcal{M}')$. Leaves of the decomposition tree obtained after the whole recursive decomposition correspond to the closure of the initial set $\mathcal{M}$. In the following, we describe how to compute $\mathcal{J}_{\mathcal{M}_{\bar{x}}}$ and $\mathcal{J}_{\mathcal{M}'}$.

Proposition 1. *Let $\mathcal{M}$, $\mathcal{M}'$ in $\mathbb{M}_n$, x in U_n and i in $[1,2]$ with $\mathcal{M} = \mathcal{M}_{\bar{x}} \cup g_{i,x}(\mathcal{M}')$. Then*

- $\mathcal{J}_{\mathcal{M}_{\bar{x}}} = \mathcal{J}_{\bar{x}}$.
- $\mathcal{J}_{\mathcal{M}'} \subseteq \{J\backslash\{x\} \mid J \in \mathcal{J}_x\} \ \cup \ \{J_1 \cup J_2 \mid J_1 \in \{J\backslash\{x\} \mid J \in \mathcal{J}_x\},\ J_2 \in \mathcal{J}_{\bar{x}}\}$

Proof.

- $\mathcal{J}_{\mathcal{M}_{\bar{x}}} = \mathcal{J}_{\bar{x}}$
  Any join-irreducible element of $\mathcal{M}$ not containing x is also join-irreducible of $\mathcal{M}_{\bar{x}}$. Indeed, every predecessors of each set in $\mathcal{M}$ not containing x, doesn't contain x itself. Similarly, there is no new join-irreducible element in $\mathcal{J}_{\mathcal{M}_{\bar{x}}}$.
- $\mathcal{J}_{\mathcal{M}'} \subseteq \{J\backslash\{x\} \mid J \in \mathcal{J}_x\} \ \cup \ \{J_1 \cup J_2 \mid J_1 \in \{J\backslash\{x\} \mid J \in \mathcal{J}_x\},\ J_2 \in \mathcal{J}_{\bar{x}}\}$
  Let us show that $g_i(\mathcal{J}_{\mathcal{M}'}) \subseteq \mathcal{J}_x \cup \{J_1 \cup J_2 \mid J_1 \in \mathcal{J}_x,\ J_2 \in \mathcal{J}_{\bar{x}}\}$.
  By contradiction, let J be a set in $g_{i,x}(\mathcal{J}_{\mathcal{M}'})$ such that $J \notin \mathcal{J}_x \cup \{J_1 \cup J_2 \mid J_1 \in \mathcal{J}_x,\ J_2 \in \mathcal{J}_{\bar{x}}\}$. So, we have $x \in J$, $J \notin \mathcal{J}_x$ and $J \notin \mathcal{J}_{\bar{x}}$. Hence, $J \notin \mathcal{J}_{\mathcal{M}}$.
  Let $S = \{s_1, s_2, ..., s_n\}$ (resp. $S' = \{s'_1, s'_2, ..., s'_m\}$) be the family of maximal join-irreducible sets of $\mathcal{M}_{\bar{x}}$ (resp. of $\mathcal{M}_x$) such that $\forall i \in [1,n]$, $s_i \subset J$ (resp. $\forall j \in [1,m]$, $s'_j \subseteq J$).

Since $J \notin \mathcal{J}_x \cup \{J_1 \cup J_2 \mid J_1 \in \{J \backslash \{x\} \mid J \in \mathcal{J}_x\},\ J_2 \in \mathcal{J}_{\bar{x}}\}$ we have $\nexists i \in [1,n]$ and $\nexists j \in [1,m]$ such that $s_i \cup s'_j = J$ or $s'_j = J$. So, $\exists i, j \in [1,n]$ and $\exists k, l \in [1,m]$ such that $s_i \cup s'_k$ is incomparable with $s_j \cup s'_l$ and such that $\forall u \in [1,n]$ and $\forall v \in [1,m]$, either $s_u \cup s'_v \subseteq s_i \cup s'_k$, or $s_u \cup s'_v \subseteq s_j \cup s'_l$. But, with $J = s_i \cup s'_k \cup s_j \cup s'_l$, we can say that J covers $s_i \cup s'_k$ and $s_j \cup s'_l$. Hence, J is not a join-irreducible set of $\mathcal{M}_x$, $J \setminus \{x\}$ is not a join-irreducible set of $\mathcal{M}'$ and we conclude that J does not belongs to $g_{i,x}(\mathcal{J}_{\mathcal{M}'})$. Contradiction.

□

Straightforward from proposition 1 we can design an algorithm to close any given input family (see algorithm 1).

For the first step of algorithm Co-closure, P is initialized to U_n that represents all possible choices of an element for the next decomposition. S is initialized to the empty set and allows to store element that would form a co-closed set. $Verif$ is initialized to $true$. An example of an execution is given in figure 2.

4.2 Discussion about the time complexity of algorithm Co-closure.

Basically, the decomposition tree obtained at the end of the whole process of Algorithm 1 is a binary tree. Internal nodes have zero, one or two children. Only the last case induces the algorithm to compute a direct product which create a new path in the tree corresponding to a new closed set. For this reason we can say that Algorithm 1 computes as many direct products as the number of different closed sets of the final result. By this way, the most important part of the whole time complexity, which corresponds to the computation of direct products, can be dispatched on each internal node with two children. Locally, the algorithm computes one direct product between two families whose the size is variable. Let S be an hypothetic maximal size of a family involved in a product. Then, treatment of an internal node is done with runtime of $\mathcal{O}(n.S^2)$. But one have to know that the size of the family resulting of the product is never superior to the number of leaves of the sub tree rooted to this internal node. That means that the local complexity is polynomially linked with the result.

We give in the last section the experimental results. They are far to be favorable to our approach and one explication could be the following: we said that the size of the direct product computed for each internal node is linked to the number of families which have to be generated further, but the process to compute this product is not linear with the size of the local result of this product. This point give us a huge space to improve our global process. That means that Algorithm 1 makes the link between computing a closure and computing a direct product under constraints to be defined.

Algorithm 1: Co-closure($\mathcal{J}$, P, S, $Verif$)

```
Input: $\mathcal{J}$ a family of sets; $P$, $S$ sets of elements; $Verif$ a Boolean.
begin
    if $\mathcal{J} \neq \{\emptyset\}$ and $\mathcal{J} \neq \{\}$ then
        Choose an element $x \in P$; $\mathcal{J}_{\mathcal{M}_{\bar{x}}} \leftarrow \mathcal{J}_{\bar{x}}$;
        Co-closure($\mathcal{J}_{\mathcal{M}_{\bar{x}}}$, $\bigcup \mathcal{J}_{\mathcal{M}_{\bar{x}}}$, $S$, $Verif$);
        $\mathcal{T} \leftarrow \{J \backslash \{x\} \mid J \in \mathcal{J}_x\}$;
        $\mathcal{J}_{\mathcal{M}'} \leftarrow \mathcal{T} \cup \{T \cup J \mid T \in \mathcal{T},\ J \in \mathcal{J}_{\mathcal{M}_{\bar{x}}}\}$;
        if $\{x\} \notin \mathcal{J}$ then
            $Verif \leftarrow false$;
        else
            $Verif \leftarrow true$;
        Co-closure($\mathcal{J}_{\mathcal{M}'}$, $\bigcup \mathcal{J}_{\mathcal{M}'}$, $S \cup \{x\}$, $Verif$);
    else
        if $Verif = true$ and $\mathcal{J} \neq \{\}$ then
            $return(S)$;
        $S \leftarrow \{\}$;
end
```

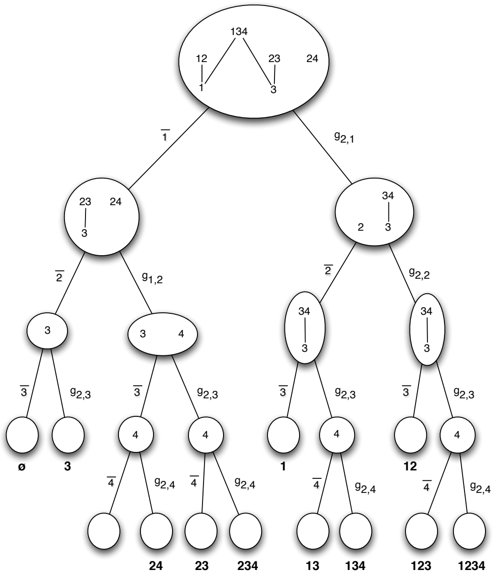

Fig. 2. Decomposition tree corresponding to the execution of the algorithm on the input family $\{\{1\}, \{1,2\}, \{3\}, \{2,3\}, \{1,3,4\}, \{2,4\}\}$

5 Experimental results

5.1 Experimental design

In order to assess performances, the approach described previously has been implemented in C. The executable file was generated with compiler GCC 4.6. The experiments were performed on a Intel Core2 Q9300 processor with 2.2 GHz of speed and 8 Go of memory. Several well-known benchmark real-world data-sets were chosen from [21]. The following table summarizes the data-sets caracteristics.

Data-sets	# Attributes	# Sets	# Attributes per set	# Closed set
Mushroom	119	8124	23	238709
Chess	75	3196	37	930851336
Connect	129	67557	43	1415737994

To compare performances of Co-closure and previous approaches with respect to the size of family, we implemented a version of Norris's algorithm (see [18]) as well as a version of Ganter's algorithm called Next-closure (see [14]). Note that we have used the natural one to one mapping between Moore and co-Moore families to compute closed sets using Co-Closure algorithm.

5.2 Results

We have compared execution time of these different algorithms on parts of benchmark data-sets. In that way, we have executed algorithms on the first m sets of each benchmark data-sets by varying m. On figures 3, 4, and 5 we give numbers of co-closed sets we obtained by execution of the three algorithms for each size of the considered data-sets partition. In the last three rows we give the execution time for each algorithm.

#sets	# closed	Ganter	Norris	Co-Closure
100	10867	20ms	10ms	160ms
200	73033	80ms	80ms	1s56
500	1051399	1s54	2s98	7s3
1000	38558373	2m04	2m48	31m52
1500	183655113	13m24	1h04	?
2000	292107880	26m36	?	?
2500	386235477	46m25	?	?
3196	930851336	2h33	?	?

Fig. 3. Chess

#sets	# closed	Ganter	Norris	Co-Closure
10	75	20ms	20ms	20ms
20	270	20ms	20ms	30ms
50	1208	30ms	20ms	40ms
100	3459	30ms	30ms	110ms
200	7086	30ms	3ms	380ms
500	17781	60ms	5ms	2s
1000	32513	210ms	100ms	6s
1500	48414	460ms	170ms	15s
2000	58982	750ms	250ms	30s
2500	72008	1s23	360ms	45s
3000	80901	1s75	480ms	79s
3500	94350	2s32	650ms	112s
4000	104104	2s99	810ms	137s
4500	122950	3s	1s21	194s
5000	136401	4s	1s47	237s
5500	150948	5s	1s78	304s
6000	156573	6s	1s92	348s
6500	195677	7s	2s80	517s
7000	214950	8s	3s30	593s
7500	230882	10s	3s73	718s
8000	237874	11s	3s94	815s
8124	238709	12s	3s96	818s

Fig. 4. Mushroom

#sets	# closed	Ganter	Norris	Co-Closure
100	13406	410ms	430ms	550ms
200	63360	470ms	440ms	2s92
500	445676	1s	1s48	2m09
1000	1069569	2s	8s	14m26
2000	4732622	22s	56s	?
5000	22543073	3m51	16m50	?
10000	69916189	23m18	3h58	?
20000	242644240	2h50	?	?
67557	1415737994	4j	?	?

Fig. 5. Connect

6 Conclusion

In [8] authors have computed the exact size of $|\mathbb{M}_7|$. From this first challenge has derived a recursive decomposition theorem which has been formalized in [9]. In the present article we first show that each Moore co-family can be represented by a decomposition tree and we use this principle to design an original algorithm to close under union any given family. The process has been implemented and the results on well-known benchmarks are given in the last section. Experimentations state the correctness of the process but show that time complexity and space complexity are not favorable. However, we think the process is interesting for two main reasons. Firstly, the decomposition tree obtained by the process is based on the recursive definition of the set of Moore co-families and corresponds to a new approach. And secondly the computing of each closure is shown to be linked to the computing of a direct product under constraints to be defined. It will be interesting to use some better practices concerning the direct product to improve the process. In that case the time complexity for each generated set will correspond to the time complexity needed for the algorithmic treatment of an internal node with two children. So, any new performance to compute the local direct product should lead to a more efficient algorithm.

References

1. Barbut, M., Monjardet, B.: Ordre et classification. Hachette (1970)
2. Beaudou, L., Colomb, P., Raynaud, O. In: submitted to ISAAC. (2011)
3. Birkhoff, G.: Rings of sets. Duke Mathematical Journal **3** (1937) 443–454
4. Birkhoff, G.: Lattice Theory. Third edn. American Mathematical Society (1967)
5. Bordat, J.P.: Calcul pratique du treillis de galois d'une correspondance. Math. Sci. Hum. **96** (1986) 31–47
6. Caspard, N., Monjardet, B.: The lattices of closure systems, closure operators, and implicational systems on a finite set : a survey. Discrete Applied Mathematics **127** (2003) 241–269
7. Cohn, P.: Universal Algebra. Harper and Row, New York (1965)
8. Colomb, P., Irlande, A., Raynaud, O.: Counting of Moore families on n=7. In: ICFCA, LNAI 5986, Agadir, Marocco. (2010)
9. Colomb, P., Irlande, A., Raynaud, O., Renaud, Y.: About the recursive décomposition of the lattice of moore co-families. In: ICFCA, Nicosia, Cyprius. (2011)
10. Davey, B.A., Priestley, H.A.: Introduction to lattices and orders. Second edn. Cambridge University Press (2002)
11. Demetrovics, J., Libkin, L., Muchnik, I.: Functional dependencies in relational databases: A lattice point of view. Discrete Applied Mathematics **40(2)** (1992) 155–185
12. Doignon, J.P., Falmagne, J.C.: Knowledge Spaces. Springer, Berlin (1999)
13. Duquenne, V.: Latticial structure in data analysis. Theoretical Computer Science **217** (1999) 407–436
14. Ganter, B.: Two basic algorithms in concept analysis., Preprint 831, Technische Hochschule Darmstadt (1984)
15. Ganter, B., Wille, R.: Formal concept analysis, mathematical foundation, Berlin-Heidelberg-NewYork et al.:Springer (1999)

16. Gely, A.: A generic algorithm for generating closed sets of a binary relation. In: ICFCA'05. (2005)
17. Lindig, C.: Fast concept analysis. In: Working with Conceptual Structures - Contributions to ICCS 2000, Shaker Verlag (August 2000) 152–161
18. Norris, E.M.: An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine de Mathématiques Pures et Appliquées **23**(2) (1978) 243–250
19. G.Sierksma: Convexity on union of sets. Compositio Mathematica **volume 42** (1981) 391–400
20. Ven, L.V.D.: Theory of convex structures. North-Hollande, Amsterdam (1993)
21. : Uci machine learning repository.