Computing on-line the lattice of maximal antichains of posets

Guy-Vincent Jourdan, Jean-Xavier Rampon, Claude Jard

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE

Computing On–Line the Lattice of Maximal Antichains of Posets

Guy–Vincent Jourdan, Jean–Xavier Rampon, Claude Jard

N° 2271

Mai 1994

PROGRAMME 1

Architectures parallèles,

bases de données,

réseaux et systèmes distribués

ISSN 0249-6399

INRIA
RENNES

Computing On–Line the Lattice of Maximal Antichains of Posets

Guy–Vincent Jourdan, Jean–Xavier Rampon, Claude Jard *

Programme 1 — Architectures parallèles, bases de données, réseaux
et systèmes distribués
Projet Pampa

Rapport de recherche n ° 2271 — Mai 1994 — 22 pages

Abstract: This paper is dedicated to the on–line computation of the lattice of maximal antichains, say $\widetilde{MA(P)} = (MA(P), \leq_{MA(P)})$, of a finite poset $\widetilde{P} = (P, \leq_P)$. This on–line computation fulfills what we call the linear extension hypothesis, that is when the ordering of the incoming vertices of $\widetilde{P}$ follows a linear extension of $\widetilde{P}$. Beside its theoretical interest, this abstraction of the lattice of antichains of a poset has structural properties which give it interesting practical behaviours. Particularly this lattice of maximal antichains seems interesting for testing distributed computations, domain where the lattice of antichains is going to be widely used. The on–line algorithm we propose for this computation has a run time complexity in $\mathcal{O}((|P| + \omega^2(P))\omega(P)|MA(P)|)$ which strictly improves the previous off–line algorithms.

(Résumé : tsvp)

*IRISA, Campus de Beaulieu, F-35042 Rennes, FRANCE. email: <name>@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

Constructions "à la volée" du treillis des antichaines maximales d'un ensemble ordonné

Résumé : Nous nous intéressons à la construction au vol du treillis des antichaînes maximales, noté $\widetilde{MA(P)} = (MA(P), \leq_{MA(P)})$, d'un ensemble fini partiellement ordonné $\widetilde{P} = (P, \leq_P)$. Nous nous plaçons dans le cas particulier dit de l'extension linéaire, c'est à dire lorsque l'ordre d'arrivée des éléments de $\widetilde{P}$ respecte une extension linéaire de $\widetilde{P}$. En dehors de considérations purement théoriques rendant ce treillis très intéressant, cette abstraction du treillis des antichaînes d'un ensemble ordonné possède des caractéristiques prometteuses pour d'éventuelles applications. Tout particulièrement ce treillis des antichaînes maximales semble être un bon candidat pour le test des exécutions réparties, domaine où l'utilisation du treillis des antichaînes connait un intérêt grandissant. L'algorithme au vol que nous proposons pour réaliser cette construction a une complexité temporelle en $\mathcal{O}((|P| + \omega^2(P))\omega(P)|MA(P)|)$, ce qui améliore strictement la complexité temporelle des algorithmes off–line jusqu'alors connus.

Computing On–Line the Lattice of Maximal Antichains of Posets

Contents

1 Introduction

This paper is dedicated to the on–line computation of the covering digraph, i.e. the digraph of the transitive reduction, of the lattice of maximal antichains of a finite poset. Beside its theoretical interest, this structure has a practical utility. Indeed, it is an abstraction of the lattice of antichains, which is already used for the verification of distributed computations.

There are usually at least three lattices which are associated to partially ordered sets (*posets* for short): the lattice of the MacNeille completion, the lattice of antichains (or ideal lattice) and the lattice of maximal antichains. These lattices are exploited to characterize properties of the considered poset (linear extensions, minimal interval extensions, dimension, jump number ...). The lattice of maximal antichains has been recently widely studied in the community of ordered sets theory. Working on lattice representation, Markowsky in [15] and [16] shows that any finite lattice is isomorphic to a Galois lattice of a binary relation. Thus he gives a constructive proof that any finite lattice is isomorphic to the lattice of maximal antichains of a height-one order. A similar result has been rediscovered by Behrendt in [3] and implicitly by Wille in [22]. Reuter in [19] focuses on the links between the jump number of a poset and the height of its lattice of maximal antichains. In [11], Habib et al. studied the relation between minimal Ferrers extensions of an incidence structure and the maximal chains of one of its Galois lattice. They show that way that interval order extensions, minimal for inclusion, are in a one-to-one correspondence with the maximal chains of its lattice of maximal antichains. This result is related to those of Bonnet and Pouzet in [4] establishing the one-to-one correspondence between the linear extensions and the maximal chains of the ideal lattice of a given poset. The lattice of maximal antichains, as well as the lattice of the MacNeille completion and the lattice of antichains, can be seen as a concept lattice (see Wille [21]). For example the lattice of maximal antichains of a poset $\widetilde{P}$ is isomorphic to the concept lattice of the context $(P, P, \not>_P)$. Bordat in [5], and Ganter and Reuter in [10] give a general algorithm which can be used to generate all concepts of a context. For the specific problem of the computation of the lattice of maximal antichains, the most efficient algorithm was, up to now, the off–line algorithm presented by Morvan and Nourine in [18], which has a run time complexity of

$\mathcal{O}(|P|\omega^2(P)|MA(P)|)$ (where $|P|$ denotes the number of elements in the poset $\widetilde{P}$, $|MA(P)|$ the number of maximal antichains of $\widetilde{P}$ and $\omega(P)$ the width of $\widetilde{P}$).

The motivation of this work is also to use the on–line computation of the lattice of maximal antichains in the context of distributed programs debugging. Indeed, a distributed execution is usually seen as a set of events partially ordered by the local time on each process and by messages passing between processes. In order to check the correctness of the distributed computation, the events are sent to an observer, which has to reorder them (see Lamport [14], Fidge [9] and Mattern [17]). The ideal lattice is useful to characterize certain classes of properties (see Babaoglu and Raynal [1], Charron-Bost et al. [6], Cooper and Marzullo [7]). In [7], Cooper and Marzullo proposed an algorithm to build on–line this lattice. The algorithm has been improved by Diehl et al. [8] and Jard et al. [13]. The on–line aspect comes from the fact that the verification and thus the computation of ideals must be performed during the execution which is to be observed. The major drawback of this method is the size of this lattice, which can be exponentially larger than the size of the poset. That is why we are interested in building some abstractions of this lattice. The lattice of maximal antichains is one of these abstractions. Note that this lattice may also be exponentially larger than the poset, but it can be practically much smaller than the ideal lattice. Our idea is to characterize and check certain properties on the lattice of maximal antichains rather than on the ideal lattice, using the fact that the interleaving of events is not exhaustive in the lattice of maximal antichains.

We present an efficient on–line algorithm to compute the covering digraph of the lattice of maximal antichains of a poset. This algorithm, based on a general on–line approach of posets, is a continuation of our work of [13]. To simplify the technical aspect, we have intentionally restricted here the on–line condition by forcing the new vertex to be maximal, but a same approach can be used in the general case of "subposet hypothesis" (see section 2.2) and an algorithm with a same time complexity can be designed. Our on–line algorithm has a run time complexity of $\mathcal{O}((|P| + \omega^2(P))\omega(P)|MA(P)|)$, which improves the result of Morvan and Nourine [18].

The paper is organized as follows: in section 2, we first give the definitions and notation, then we shortly describe the on–line paradigm on posets. In sec-

tion 3, we focus on the structural evolution of the lattice of maximal antichain when a new maximal element is added to the poset. We first show how to deduce the new elements in the lattice from the current lattice and the new vertex. Then we characterize the new covering relations. This gives us the principle of the algorithm, given in section 4. In this section, we show how the aforementioned modifications can be computed efficiently.

2 Definitions and Preliminaries

2.1 Definitions

Let $\widetilde{P} = (P, \leq_P)$ be a partially ordered set (*poset* for short). Throughout the paper we consider only finite posets. The covering digraph of $\widetilde{P}$, $cov(\widetilde{P})$, is the digraph of the transitive reduction of the relation $\leq_P$. For $x \in P$, $\downarrow_P^{[} x$ (resp. $\downarrow_P^{]} x$) is the set of all predecessors of x in $\widetilde{P}$, x excluded (resp. included), $\downarrow_P^{im} x$ is the set of all immediate predecessors of x in $\widetilde{P}$; $\uparrow_P^{[} x$, $\uparrow_P^{]} x$ and $\uparrow_P^{im} x$ have similar definitions on successors. We extend these notations to subsets of P: $\forall A \subseteq P$, $\downarrow_P^{[} A = \{\downarrow_P^{[} x, x \in A\}$, $\downarrow_P^{im} A = \{\downarrow_P^{im} x, x \in A\}$ and similarly for $\downarrow_P^{]} A$, $\uparrow_P^{[} A$, $\uparrow_P^{]} A$ and $\uparrow_P^{im} A$. We define $Max_P(A)$ (resp. $Min_P(A)$) as the set of all maximal (resp. minimal) elements of A in $\widetilde{P}$.

A subset A of P is called an *antichain* (resp. a *chain*) of $\widetilde{P}$ iff elements of A are pairwise incomparable (resp. comparable). The set of all antichains of $\widetilde{P}$ is noted $A(P)$. A is an *ideal* of $\widetilde{P}$ iff $\downarrow_P^{]} A = A$. The set of all ideals of $\widetilde{P}$ is denoted $I(P)$. It is well known (see Birkoff [2]) that $I(P)$ ordered by set inclusion is a distributive lattice, which we note $\widetilde{I(P)} = (I(P), \subseteq)$. The mapping $M : I \longrightarrow Max_P(I)$ is an order isomorphism between $\widetilde{I(P)}$ and $\widetilde{A(P)}$, the lattice of antichains of $\widetilde{P}$ (i.e. $\forall Y, Z \in I(P), Y \subseteq Z \Longleftrightarrow M(Y) \leq_{A(P)} M(Z)$). An antichain A is a *maximal antichain* of $\widetilde{P}$ iff every element of P is comparable with some element of A. An ideal I is a *maximal ideal* of $\widetilde{P}$ iff $Max_P(I)$ is a maximal antichain. We note $MI(P)$ the set of all maximal ideals of $\widetilde{P}$. We can order $MI(P)$ by set inclusion to obtain the poset $\widetilde{MI(P)} = (MI(P), \subseteq)$. The poset $\widetilde{MI(P)}$ is a lattice, the *lattice of maximal ideals* of $\widetilde{P}$. There is also an order isomorphism between $\widetilde{MI(P)}$ and $\widetilde{MA(P)}$, the poset of maximal antichains of $\widetilde{P}$ by the mapping M. In the following, we will use the lattice

$\widetilde{MI(P)}$ rather than the lattice $\widetilde{MA(P)}$, since the set approach allows more readable proofs. Figure 1 gives an example of a poset $\widetilde{P}$ and the two lattices $\widetilde{A(P)}$ and $\widetilde{MA(P)}$. In $\widetilde{A(P)}$, antichains which are maximal are drawn in grey.

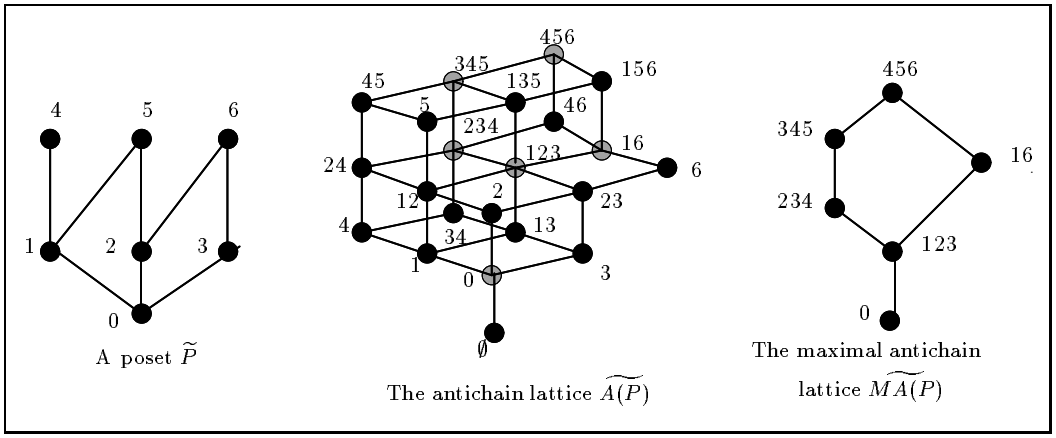

Figure 1: A poset $\widetilde{P}$ and the lattices $\widetilde{A(P)}$ and $\widetilde{MA(P)}$

To conclude with definitions, $\omega(P)$ is the maximal number of elements belonging to an antichain of $\widetilde{P}$. A *linear extension* of $\widetilde{P}$ is a totally ordered set on P including $\widetilde{P}$. A poset $\widetilde{Q} = (Q, \leq_Q)$ is a *subposet* of $\widetilde{P}$ iff $Q \subseteq P$ and $\forall x, y \in Q, x \leq_P y \iff x \leq_Q y$. Moreover, if $Q \neq P$ then $\widetilde{P}$ is a *poset extension* of $\widetilde{Q}$.

2.2 "On–line" approach of posets

An on–line algorithm is supposed to work as follows: at each step, we have an additional and unpredictable knowledge and we want to compute a given function on all the already known knowledge. In our application, the observer has the knowledge of a certain amount of events, which induces a poset $\widetilde{P}$, and he has computed the lattice $\widetilde{MI(P)}$. He then receives a new event x, which transforms $\widetilde{P}$ into $\widetilde{P'}$ and he has to compute $\widetilde{MI(P')}$. Note that any off–line algorithm can be transformed into an on–line one : we just have to system-

atically compute the value at each step on all the already known knowledge. A way to design efficient on–line algorithm is to use as much as possible the values computed at precedings steps to compute the value at a following step.

When dealing with posets, the current knowledge is a poset $\widetilde{P}$ and the additional knowledge is a new vertex $x \notin P$ together with the list of its predecessors in P and the list of its successors in P. In order to preserve the structure of the poset, we define the on–line approach of posets in the following way: at each step, the current poset $\widetilde{P}$ is always a subposet of the current poset $\widetilde{P'}$ obtained at the next step, that is at each step the new poset $\widetilde{P'}$ is a poset extension of the preceding poset $\widetilde{P}$. That is what we call the "subposet hypothesis". This subposet hypothesis is achieved by the time stamps method of Fidge [9] and Mattern [17]. As particular case, we can assume that the new vertex x is maximal in the new poset $\widetilde{P'}$. That is what we call the "linear extension hypothesis". The time stamps method also allows to reach such an hypothesis. The goal is in fact to preserve the structure of the poset during time. Note that as far as graphs are concerned, the usual technique which consists to add a new vertex x and the list of its neighbors in the already known graph G ([12, 20]) ensures that G is a subgraph of the new graph G', without additional assumption. This technique, used on posets, can lead to an inconsistency since the new vertex may induce, by transitivity, new comparabilities between vertices of the already known poset.

In this paper, we have chosen to focus on the linear extension hypothesis, which enables us to describe the principle of our approach. But an algorithm based on the same principle can also be designed under subposet hypothesis. This algorithm is technically much more difficult, but has the same time complexity.

3 $\widetilde{\mathbf{MI}}(\mathbf{P})$ Versus $\widetilde{\mathbf{MI}}(\mathbf{P'})$

In the sequel, we assume that we have a poset $\widetilde{P} = (P, \leq_P)$ and the already computed lattice $\widetilde{MI}(P)$. We also have a new vertex $x \notin P$, together with the list of its immediate predecessors $\downarrow^{im}_{P'} x$. This defines a new poset $\widetilde{P'} = (P', \leq_{P'})$, where $P' = P \cup \{x\}$ and we want to compute the lattice $\widetilde{MI}(P')$, using the knowledge induced by $\widetilde{MI}(P)$. In order to show how $Cov(\widetilde{MI}(P'))$ can be

deduced from $Cov(\widetilde{MI(P)})$, we first give some technical properties. Then we characterize the elements of the set $MI(P')$ using the elements of the set $MI(P)$. Finally we show how the covering relations of $\widetilde{MI}(P')$ can be obtained from the covering relations of $\widetilde{MI}(P)$. This characterization, given in a constructive way, gives the underlying structure of our algorithm.

3.1 Preliminaries

In [19], Reuter gives a characterization of the immediate successor relation in $\widetilde{MI(P)}$ in terms of complete height–one[1] subcovering digraph of $\widetilde{P}$:

Lemma 1 *[19] Let* $I_1, I_2 \in MI(P)$. $I_2 \in \uparrow_P^{im} I_1$ *iff the set* $(I_2 - I_1) \cup (\uparrow_P^{]} Max_P(I_1) - \uparrow_P^{]} Max_P(I_2))$ *induces a complete height-one subposet of* $\widetilde{P}$ *having* $(I_2 - I_1)$ (resp. $(\uparrow_P^{]} Max_P(I_1) - \uparrow_P^{]} Max_P(I_2))$) *as maximal* (resp. *as minimal) elements.*

As a consequence of Lemma 1, we give the following lemma, which enlightens some properties on the elements of P added in covering relations of $\widetilde{MI(P)}$:

Lemma 2 $\forall\ I \in MI(P), \forall\ J_1, J_2 \in \uparrow_{MI(P)}^{im} I$ *such that* $J_1 \neq J_2$,
(i) $J_1 - I$ *is an antichain of* $\widetilde{P}$
(ii) $(J_1 - I) \cap (J_2 - I) = \emptyset$

Reuter also notes in [19] that the completion of an element I of $I(P)$ in an element J of $MI(P)$ (i.e. $I \subseteq J$) is not unique, but that there exists a lowest completion (i.e. there exists $J_l \in MI(P)$ such that $I \subseteq J_l$ and for all $J \in MI(P)$, $I \subseteq J \Longrightarrow J_l \subseteq J$). In the following, we use $\overset{min}{Comp}_{MI(P)}(I)$ to denote such a lowest completion. We propose the following lemma, which shows that $\overset{min}{Comp}_{MI(P)}(I)$ is obtained by adding to I the minimal elements of the set of elements which are incomparable with $Max_P(I)$:

Lemma 3 $\forall\ I \in I(P)$, $\overset{min}{Comp}_P(I) = I \cup Min_P(P - (I \cup \uparrow_P^{[} Max_P(I)))$

[1] A height–one poset has no chains with more than two elements. An height–one poset is complete iff all its minimal elements are comparable with all its maximal elements.

Proof: Let $I' = I \cup Min_{P}(P - (I \cup \uparrow^{[}_{P} Max_{P}(I)))$, then by definition, $I' \in MI(P)$ and $I \subseteq I'$. Let $J \in MI(P)$ be such that $I \subseteq J$ and assume that $I' \not\subseteq J$, i.e. $\exists y \in I' - J$. Then $\exists t \in Max_{P}(J)$ such that $t < y$. But since $I \subseteq J$, either $t \in \uparrow^{]}_{P} Max_{P}(I)$ or $t \in P - (I \cup \uparrow^{[}_{P} Max_{P}(I))$. In both cases, there is a contradiction with the definition of I', thus $I' \subseteq J$. ■

With Lemma 3, it is then straightforward to prove that the lowest completion is order preserving:

Lemma 4 $\forall I, J \in I(P), I \subseteq J \Longrightarrow \overset{min}{Comp}_{MI(P)}(I) \subseteq \overset{min}{Comp}_{MI(P)}(J)$.

3.2 The Set Relations

The set $MI(P)$ can be seen as a kind of subset of $MI(P')$. Indeed, each element of $MI(P)$ will generate either one or two elements of $MI(P')$. This can be obtained in splitting $MI(P)$ into three subsets, characterized by $\downarrow^{im}_{P'} x$, namely $A_1 = \{I \in MI(P), \downarrow^{im}_{P'} x \not\subseteq I\}$, $A_2 = \{I \in MI(P), \downarrow^{im}_{P'} x \subseteq I \text{ and } \downarrow^{im}_{P'} x \cap Max_{P}(I) \neq \emptyset\}$ and $A_3 = \{I \in MI(P), \downarrow^{im}_{P'} x \subseteq (I - Max_{P}(I))\}$. Elements of A_1 will not be changed in $MI(P')$. Elements of A_2 will also belong to $MI(P')$, but some of them will generate new elements. Elements of A_3 will be modified. First note the following property :

Property 1 *The non empty A_i's define a partition of $MI(P)$.*

Remark 1

(*i*) $\forall\ i \in \{1, \cdots, 3\}, \uparrow^{[}_{MI(P)} A_i \subseteq \bigcup_{j=i}^{3} A_j$

(*ii*) *The set $A_2 \cup A_3$ has an infimum which is $\overset{min}{Comp}_{MI(P)}(\downarrow^{im}_{P'} x)$.*

Before characterizing $MI(P')$, we have to point out some particular elements of A_2, which define the set $A'_2 = \{I \in A_2,\ I \cup \{x\} \in MI(P')\}$. These are elements of A_2 generating two elements in $MI(P')$. They are characterized as follow:

Property 2 $\forall\ I \in A_2, (I \in A'_2) \Longleftrightarrow$
$(\forall\ y \in \uparrow^{im}_{P}(\downarrow^{im}_{P'} x \cap Max_{P}(I)),\ y \in \uparrow^{[}_{P}(Max_{P}(I) - \downarrow^{im}_{P'} x))$

To compute some covering relations in $IM(P')$, we will need the following technical lemma:

Lemma 5 $\forall I \in A_2 - A_2', \forall J \in A_3, J \in \uparrow_P^{im} I \Longrightarrow$
$\downarrow_{P'}^{im} x \cap Max_P(I) = Max_P(I) - Max_P(J)$

Proof: By definition of A_3, $\downarrow_{P'}^{im} x \cap Max_P(I) \subseteq Max_P(I) - Max_P(J)$. Let $y \in (Max_P(I) - Max_P(J)) - (\downarrow_{P'}^{im} x \cap Max_P(I))$ and let $z \in \uparrow_P^{im}(\downarrow_{P'}^{im} x \cap Max_P(I))$, then either $z \in \uparrow_P^{[}(Max_P(I) \cap Max_P(J))$, or $z \in \uparrow_P^{]}(Max_P(J) - Max_P(I))$, and thus with Lemma 1 $z \in \uparrow_P^{[} y$. So by Property 2, $I \in A_2'$, contradiction. ■

To characterize $IM(P')$, we define the sets $B_1 = A_1$, $B_2 = A_2$, $B_3 = \{I \cup x,\ I \in A_2'\}$ and $B_4 = \{I \cup \{x\},\ I \in A_3\}$. Since the A_i's are pairwise disjoint, so are the B_i's. Moreover, it is easy to check that $MI(P') = \bigcup B_i$. Thus we get:

Property 3 *The non empty B_i's define a partition of $MI(P')$*

Remark 2 $\forall\, i \in \{1, \cdots, 4\}, \uparrow_{MI(P')}^{[} B_i \subseteq \bigcup_{j=i}^{4} B_j$

Figure 2 shows the partition of $MI(P)$ induced by the A_i's and the partition of $MI(P')$ induced by the B_i's.

3.3 The Covering Relations

We now have to order the B_i's by set inclusion to get $\widetilde{MI(P')}$. Once again, we use the knowledge of $\widetilde{MI(P)}$. Indeed, a part of the covering relations can be exactly deduced from $\widetilde{MI(P)}$. For the remaining covering relations, the use of the lowest completion provides a necessary condition. This will be enough for our algorithm.

Theorem 1 *Let $I \in MI(P')$, we have:*

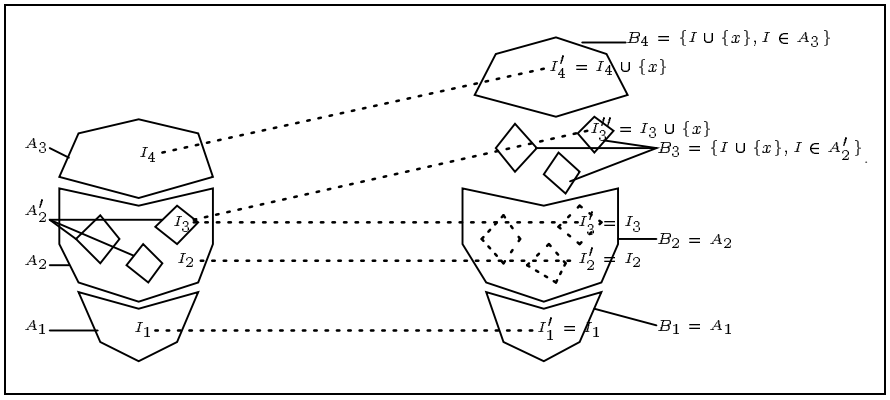

Figure 2: Partition induced by $\downarrow^{im}_{P'} x$ on $\widetilde{MI(P)}$ and $\widetilde{MI(P')}$

(i) for $I \in B_1, \uparrow^{im}_{MI(P')} I = \uparrow^{im}_{MI(P)} I$

(ii) for $I \in B_2, \uparrow^{im}_{MI(P')} I = (\uparrow^{im}_{MI(P)} I \cap A_2) \cup \{I \cup \{x\}$ if $I \in A'_2$

$= (\uparrow^{im}_{MI(P)} I \cap A_2) \cup \{(J \cup \{x\}), J \in (\uparrow^{im}_{MI(P)} I \cap A_3)\}$ otherwise

$(iii)$for $I \in B_3, \uparrow^{im}_{MI(P')} I \subseteq \{\overset{min}{Comp}_{MI(P')} (J \cup \{x\}), J \in \uparrow^{im}_{MI(P)} (I - \{x\})\}$

$(iv)$for $I \in B_4, \uparrow^{im}_{MI(P')} I = \{J \cup \{x\}, J \in \uparrow^{im}_{MI(P)} (I - \{x\})\}$

Proof: (i) It is an immediate consequence of $\uparrow^{im}_{IM(P)} A_1 \subseteq A_1 \cup A_2$.

(ii) let $I \in B_2$, then $I \in A_2$, and thus $\uparrow^{im}_{MI(P')} I \cap B_2 = \uparrow^{im}_{MI(P)} I \cap A_2$. If $I \in A'_2$, then $I \cup \{x\} \in MI(P')$ and so Lemma 1 ensures that $I \cup \{x\} \in \uparrow^{im}_{MI(P')} I$. We conclude this case since Lemma 2 (ii) ensures that $I \cup \{x\}$ is the only immediate successor of I outside B_2. If $I \in A_2 - A'_2$, let $J \in \uparrow^{im}_{MI(P)} I \cap A_3$, then Lemma 5 and then Lemma 1 ensure that $J \cup \{x\} \in \uparrow^{im}_{MI(P')} I \cap B_4$. In order to conclude, consider $J \in \uparrow^{im}_{MI(P)} I \cap (B_3 \cup B_4)$. First of all we can notice that $J \in B_3$ is impossible, since $J - \{x\} \in A'_2$ and so $I \subset J - \{x\} \subset J$. So $J \in B_4$, thus $J - \{x\} \in A_3$ and Lemma 1 ensures that $J \in \uparrow^{im}_{MI(P')} I \Longrightarrow J - \{x\} \in \uparrow^{im}_{MI(P)} I$, since $\{x\} \subseteq J - I$.

(*iii*) Let $K \in \uparrow^{im}_{MI(P')} I$, then $I - \{x\} \subset K - \{x\}$ and $\{I - \{x\}, K - \{x\}\} \subseteq A_2 \cup A_3$. So $\exists J \in \uparrow^{im}_{MI(P)} I - \{x\}$ such that $I - \{x\} \subset J \subseteq K - \{x\}$ and then by Lemma 4 we get $\overset{min}{Comp}_{MI(P')}(I) \subseteq \overset{min}{Comp}_{MI(P')}(J \cup \{x\}) \subseteq \overset{min}{Comp}_{MI(P')}(K)$. Since $\overset{min}{Comp}_{MI(P')}(I) = I$, $J \cup \{x\} \subseteq \overset{min}{Comp}_{MI(P')}(J \cup \{x\})$ and $\overset{min}{Comp}_{MI(P')}(K) = K$, we have $I \subset \overset{min}{Comp}_{MI(P')}(J \cup \{x\}) \subseteq K$ which concludes the proof.
(*iv*) It follows from Remark 2. ■

Theorem 1 is illustrated in figure 3. For sake of readability, we use $\widetilde{MA}(P)$ instead of $\widetilde{MI}(P)$.

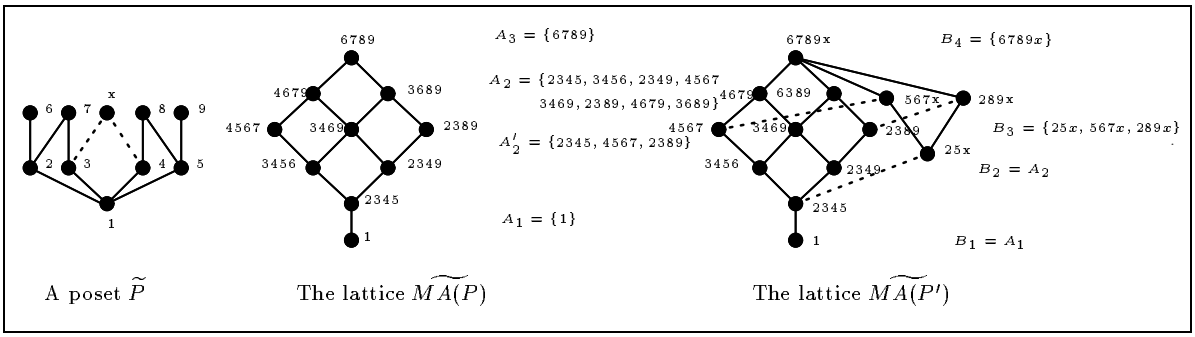

Figure 3: Deducing $\widetilde{MA}(P')$ from $\widetilde{MA}(P)$, when $P' = P \cup \{x\}$

With Property 3 and Theorem 1, we have the tools needed for the on–line computation of $Cov(\widetilde{MI}(P'))$. Since these results are constructive, we only have to focus on efficiency.

4 The algorithm

According to the previous section, the algorithm has to find the sets A_1, A_2 and A_3 into $MI(P)$. Then it duplicates elements of A'_2 and it modifies the covering relations w.r.t. the rules given in Theorem 1.

4.1 Data Structures

Assumptions on data are the following: each vertex y of P is labeled with respect to a linear extension on $\widetilde{P}$ and there is a list $\downarrow_P^{im} y$ which contains the immediate predecessors of y in $\widetilde{P}$ and which is sorted by increasing labels. Hence the new incoming element x has the highest label $(|P|+1)$ and is received with the list $\downarrow_{P'}^{im} x$.

Two lists, sorted by increasing labels, are associated with every element I of $MI(P)$: $Max(I)$, the list of its maximal elements and $label(I) = \{y,\ y \in (J-I),\ J \in \uparrow_{MI(P)}^{im} I\}$, the list of elements of P which are added to I in a covering relation. Moreover, any element y of $label(I)$ has a direct access to its corresponding element in P and a direct access, denoted by $edge(y)$, to its corresponding edge in $\widetilde{MI(P)}$ (which is unique by Lemma 2). The set of all elements of $label(I)$ related to the same edge e between I and J is exactly the set $J-I$. This set is stored in a list $list(e)$ sorted by increasing labels. We access to J via $tail(e)$.

Note that all lists $\downarrow_P^{im} y$ for $y \in P'$, $label(I)$ and $Max(I)$ for $I \in MI(P)$ and $list(e)$ for an edge e of $Cov(\widetilde{MI}(P))$ contain always a set of pairwise incomparable elements in $\widetilde{P}$. Thus they all have a size bounded by $\omega(P)$.

4.2 Computation of $\mathbf{C\overset{min}{o}mp_{MI(P')}\ (I \cup \{x\})}$

We will have to compute $\overset{min}{Comp}_{MI(P')}\ (I \cup \{x\})$ for some $I \in A_2 \cup A_3$. When $I \in A_2 \cup A_3$, with Lemma 3 we can state that $\overset{min}{Comp}_{MI(P')}\ (I \cup \{x\}) = I \cup \{x\} \cup L_I$, where $L_I = Min_{P'}(\uparrow_{P'}^{[}(Max_{P'}(I) \cap \downarrow_{P'}^{im} x) - \uparrow_{P'}^{[}(Max_{P'}(I) - \downarrow_{P'}^{im} x))$
$= \{y \in \uparrow_{P'}^{im}(Max_{P'}(I) \cap \downarrow_{P'}^{im} x), \downarrow_{P'}^{[} y \subseteq I \text{ and } \downarrow_{P'}^{im} y \cap Max_{P'}(I) \subseteq \downarrow_{P'}^{im} x \cap Max_{P'}(I)\}$.

Moreover, since $I \cup \{x\} \cup L_I \in B_3 \cup B_4$, then $I \cup L_I \in A_2 \cup A_3$. So in order to compute $\overset{min}{Comp}_{MI(P')}\ (I \cup \{x\})$, we only have to find $I \cup L_I$ in $\widetilde{MI(P)}$, using the algorithm **Complet(I)**, which works has follows: there exists a path between I and $I \cup L_I$ in $\widetilde{MI(P)}$. In order to move along such a path, at each step we choose an immediate successor which adds only elements of L_I to the current ideal. There exists a naive version of that algorithm which runs in $\mathcal{O}(\omega^3(P))$. In order to improve this time complexity, we first assume that $\forall y \in \downarrow_{P'}^{im} x, \forall z \in \uparrow_P^{im} y$, z has been marked. Clearly, all elements of L_i are marked. This allows us to select at each step a set of "candidates" (the

list *Current*). We put the candidates which belong to L_I into a list *lost* and the other candidates into a list *notlost*. That way, we have only one test per candidate, and we reach a time complexity of $\mathcal{O}(\omega^2(P))$.

Lemma 6 *For $I \in A_2 \cup A_3$, the algorithm* **complet(I)** *returns the ideal $I \cup L_I$ in $\mathcal{O}(\omega^2(P))$.*

ALGORITHM "Complet"

input : $I \in A_2 \cup A_3, \downarrow^{im}_{P'} x$

output : The element $I \cup L_I$ in $\widetilde{MI(P)}$

precondition : $\forall y \in \downarrow^{im}_{P'} x, \forall z \in \uparrow^{im}_{P} y$, z is marked

Var *Current*, *lost*, *notlost*: sorted list of poset vertices **init** to empty
Res: element of $\widetilde{MI(P)}$

Begin
Res := I; *Current* := marked elements of *label*(I); /* set of candidates */
Forall $y \in Current$
If $\downarrow^{im}_{P'} y \cap Max(I) \subseteq \downarrow^{im}_{P'} x \cap Max(I)\}$ **Then** $lost := lost \cup y$ /* $y \in L_I$ */
Else $notlost := notlost \cup y$; /* $y \notin L_I$ */
*(1)***While** $lost \neq \emptyset$ **Do**
let $y \in label(Res) \cap lost$; $lost := lost - list(edge(y))$;
(2) $Res := tail(edge(y))$;
Current := marked elements of $label(Res) - (lost \cup notlost)$; /* set of candidates */
Forall $y \in Current$
If $\downarrow^{im}_{P'} y \cap Max(Res) \subseteq \downarrow^{im}_{P'} x \cap Max(Res)\}$ **Then** $lost := lost \cup y$ /* $y \in L_I$ */
Else $notlost := notlost \cup y$; /* $y \notin L_I$ */
EndWhile;
End;

Proof: We show that at step *(1)*, $Res \subseteq I \cup L_I$. At first step *(1)*, ${Res_1}^2 = I$. Assume that after i steps *(1)*, $Res_i \subseteq I \cup L_I$. Thus $\forall k \leq i$, $Max(I) - \downarrow^{im}_{P'} x = Max(Res_k) - \downarrow^{im}_{P'} x$. If $lost_i \neq \emptyset$, we choose $y \in label(Res_i) \cap lost_i$ (such a

[2] In the following, we note Res_k and $lost_k$ the value of *Res* and *lost* at the k^{ith} step *(1)*

y always exists, since $lost_i \neq \emptyset$ and there is a path between I and $I \cup L_I$ in $\widetilde{MI(P)}$). We have $\downarrow_P^{im} y \cap Max(Res_i) \subseteq \downarrow_P^{im} x \cap Max(Res_i)$ because $y \in lost_i$. Moreover by Lemma 1, $\forall z \in list(edge(y)), \downarrow_P^{im} z \cap Max(Res_i) = \downarrow_P^{im} y \cap Max(Res_i)$, so $z \in lost_i$ and thus after step *(2)*, $Res_{i+1} \subseteq I \cup L_I$.

When the algorithm terminates (it obviously terminates since each vertex may be put into *lost* only once), $\forall y \in label(Res), \downarrow_{P'}^{im} y \cap Max(Res) \not\subseteq \downarrow_{P'}^{im} x \cap Max(Res)\}$, thus $Res \in A_2' \cup A_3$, i.e. $Res \cup \{x\} \in B_3 \cup B_4$. Since $Res \subseteq I \cup L_I$, we get $I \cup \{x\} \subseteq Res \cup \{x\} \subseteq I \cup \{x\} \cup L_I$, and thus $Res \cup \{x\} = I \cup \{x\} \cup L_I$ by minimality of $I \cup \{x\} \cup L_I$.

For the time complexity, thanks to the data structure (all lists are sorted and have a size bounded by $\omega(P)$) each step can be done independently in $\mathcal{O}(\omega(P))$. During the whole computation, each element of $Current$ goes either in *lost* or in *notlost*. In *lost*, we put elements of L_I, which is an antichain by definition. Moreover, *notlost* is also an antichain, else $\exists y, z \in notlost$ such that $y \leq_P z$, and when $z \in label(Res), y \in Res$ and thus $Max(I) - \downarrow_{P'}^{im} x \neq Max(Res) - \downarrow_{P'}^{im} x$. So there is no more than $\omega(P)$ elements into *lost* and *notlost*, i.e. no more than $2\omega(P)$ elements into $Current$, during the all computation. ■

4.3 Computation of $\mathbf{Cov}(\widetilde{\mathbf{MI}}(\mathbf{P}'))$

We give now the algorithm **"COV"**, which takes as input x and $\downarrow_{P'}^{im} x$ (sorted) and deduces $\widetilde{MI(P')}$ from $\widetilde{MI(P)}$. This algorithm is based on Theorem 1. The detection of A_1, A_2 and A_3, as well as the transformation of A_3 into B_4, is not detailed. For the generation of B_3 and the covering relations in $B_2 \cup B_3 \cup B_4$, we use a list L which will contain all elements of A_2'. For each element in L, we generate the corresponding element in B_3 and compute a superset of the covering relations according to Theorem 1. We then delete transitive relations using Lemma 1. The algorithm uses three procedures, namely $duplicate(I : ideal)$, $remove\_edge(IJ : edge)$ and $add\_edge(IJ : edge)$. The procedure $duplicate(I : ideal)$ creates a new ideal $I' = I \cup \{x\}$, if it is not already done, then puts $(Max(I) - \downarrow_{P'}^{im} x) \cup \{x\}$ into $Max(I')$, adds x at the end of $label(I)$, creates edge II' with x in $list(II')$. Procedure $add\_edge(IJ : edge)$

creates an edge IJ between I and J, adds $Max(J) - Max(I)$ to $label(I)$ and to $list(IJ)$ according to the ordering. Procedure $remove\_edge(IJ : edge)$ removes this edge and removes $list(IJ)$ from $label(I)$. Thanks to data structure, these procedures may be performed in no more than $\mathcal{O}(\omega(P))$.

Theorem 2
We can deduce $\widetilde{MI(P')}$ *from* $\widetilde{MI(P')}$ *in* $\mathcal{O}(|MI(P)|\omega(P) + (|MI(P')| - |MI(P)|)\omega^3(P) + \omega^2(P))$.

ALGORITHM "COV"

input : $x, \downarrow^{im}_{P'} x, \widetilde{MI(P)}$

output : $\widetilde{MI(P')}$

Var L : list of elements of $\widetilde{MI(P)}$ **init** to empty
I, I', J, J' : elements of $\widetilde{MI(P)}$

Begin
$\forall y \in \downarrow^{im}_{P'} x, \forall z \in \uparrow^{im}_{P} y$, mark z ; /* precondition of procedure $Complet()$ */
Compute A_1, A_2 and A_3; Let inf be the infimum of $A_2 \cup A_3$;
/* elements of A_3 */
$\forall I \in A_3$, add x at the end of $Max(I)$; /* I becomes $I \cup \{x\}$ in $IM(P')$ */
$\forall$ edge $IJ \in \widetilde{MI(P)}$ such that $I \in A_2$ and $J \in A_3$, add x at the end of $list(IJ)$ and x at the end of $label(I)$;
/* elements of A_2 */
$inf := complet(inf)$; **If** $inf \in A_2$ **Then** L:=inf;
While $L \neq$ empty **Do**
Let $I \in L$; $L := L - I$; $I' := duplicate(I)$ /* $I \in A'_2$ */
Forall $J \in \uparrow^{im}_{MI(P)} I$ /* Computation of $\uparrow^{im}_{MI(P')} I'$ according to Theorem 1 */
If $J \in A_2$ **Then** $J := complet(J)$
Else remove_edge(IJ); /* $I \in A_3$, so $J \notin \uparrow^{im}_{MI(P')} I$ */
If $J \in A_2$ **Then begin**
$J' := duplicate(J)$;
If $J \notin L$ **Then** $L := L \cup J$;
EndIf;
Else $J' := J$; /* $J \in A_3$, so the possible edge is $I'J$ */
If $J' \in \uparrow^{im}_{MI(P')} I'$ **then** add_edge($I'J'$); /* Theorem 1 ensures that we find a superset of $\uparrow^{im}_{MI(P')} I'$ */
EndForall;
EndWhile;
$\forall y \in \downarrow^{im}_{P'} x, \forall z \in \uparrow^{im}_{P} y$, unmark z ; /* restore */
End;

Proof: The correctness of the algorithm **"COV"** is a consequence of Theorem 1. This theorem ensures that we add only correct covering relations. And we add all the new covering relations because clearly, if inf is the infimum of

$A_2 \cup A_3$, $complet(inf)$ is the infimum of $B_3 \cup B_4$ (and thus each element of B_3 will be reached).

For the time complexity, marking and unmarking $\uparrow_P^{im}(\downarrow_{P'}^{im} x)$ is done in $\mathcal{O}(\omega^2(P))$. The computation of the A_i's can be done in $\mathcal{O}(|MI(P)|\omega(P))$, in listing in a first breadth-first search the ideals $I \in MI(P)$ such that $\downarrow_{P'}^{im} x \subseteq Max(I)$ (we obtain a subset of A_2 which contains the infimum of $A_2 \cup A_3$). Then a second breadth-first search from the already found subset of A_2, listing the ideals $I \in MI(P)$ such that $\downarrow_{P'}^{im} x \cap Max(I) \neq \emptyset$ gives us exactly A_2 and permits to find the infimum of $A_2 \cup A_3$. So this part of the algorithm, as well as the computation of the elements of A_3, may be performed in $\mathcal{O}(|MI(P)|\omega(P))$.

For the computation on elements of A_2, it takes $\mathcal{O}(\omega^2(P))$ to find $complet(inf)$. The body of the "Forall" loop can be performed in $\mathcal{O}(\omega^2(P))$ (for the test $J' \in \uparrow_{MI(P')}^{im} I'$, we use Lemma 1). Since there is no more than $\omega(P)$ immediate successors of I, the body of the "While" loop may be performed in $\mathcal{O}(\omega^3(P))$. Now the tricky point is that we put only elements of A_2' into the list L, and only once. Since $|A_2'| \leq |MI(P')| - |MI(P)|$, we reach the announced time complexity. ■

Corollary 1 *Let $\widetilde{P}$ be a poset. We can compute on-line the covering graph of the lattice of maximal antichains of $\widetilde{P}$ in $\mathcal{O}(|MI(P)|(|P|\omega(P) + \omega^3(P)))$.*

Proof: Clearly, with theorem 2, we compute on-line the covering digraph of $\widetilde{MI(P)}$ in $\mathcal{O}(\sum_{i=1}^{|P|}(|MI(P_{i-1})|\omega(P_{i-1}) + (|MI(P_i)| - |MI(P_{i-1})|)\omega^3(P_{i-1}) + \omega^2(P_{i-1}))) = \mathcal{O}(|MI(P)|(|P|\omega(P) + \omega^3(P)) + |P|\omega^2(P))$. Let C be the maximal number of elements in a chain of $\widetilde{P}$. We conclude since for all poset $\widetilde{P}$ we have $C * \omega(P) \geq |P|$ and $|MI(P)| \geq C$. ■

If a poset $\widetilde{Q}$ is given by its covering relations, since we can simulate our on–line assumptions in no more than $\mathcal{O}(|Q|\ \omega(Q))$, the algorithm may also be executed off–line with the same complexity. This improves the result given in [18].

5 Conclusion

We have given an algorithm to compute on–line the maximal antichain lattice of a poset $\widetilde{P}$, within a time complexity of $\mathcal{O}(|MI(P)|(|P|\omega(P)+\omega^3(P)))$. This algorithm is more efficient than the best off-line algorithm known so far.

There are some possibilities to get a more efficient off-line algorithm using our method. Especially, it may be possible to improve our complexity result by finding a particular computable linear extension which simplifies the construction.

Moreover, it is possible to weaken our constraints on $\widetilde{P}$ and $\widetilde{P'}$ by suppressing the maximality condition on x ("linear extension hypothesis"). Hence, one would remain with the only assumption that $\widetilde{P}$ is a subposet of $\widetilde{P'}$, where $P' = P \cup \{x\}$ ("subposet hypothesis"). Using a similar approach, we reach an algorithm of the same time complexity.

Acknowledgements : We are grateful to C. Bareau, B. Ganter and D. Kratsch for the fruitful discussions we had. This work has received a financial support from the French national project C^3 on concurrency, the French-Israeli research cooperation and the research center of the French army (Celar).

References

[1] O. Babaoglu, M. Raynal, *Sequence–Based Global Predicates for Distributed Computations: Definitions and Detection Algorithms*, IRISA research report N^o 729, May 1993.

[2] G. Birkhoff, *Rings of Sets*, Duke Math. J-3, 1937, 311–316.

[3] G. Behrendt, *Maximal Antichains in Partially Ordered Sets*, ARS Combinatoria 25C, 149–157, 1988.

[4] R. Bonnet, M. Pouzet, *Extensions et stratifications d'ensembles dispersés*, C.R.A.S. Paris, t. 268, Série A, p. 1512–1515, 1969.

[5] J.P. Bordat, *Calcul pratique du treillis de Galois d'une correspondance*, Math. Sci. Hum. 96 (1896), pp 31–47.

[6] B. Charron–Bost, C. Delporte–Gallet, H. Fauconnier, *Local and Temporal Predicates in Distributed Systems*, research report N^o 92–36, LITP, Paris 7, 1992.

[7] R. Cooper and K. Marzullo, *Consistent Detection of Global Predicates* In Proc. ACM/ONR *Workshop on Parallel and Distributed Debugging*, pp 163–173, Santa Cruz, California, May 1991.

[8] C. Diehl, C. Jard, J.X. Rampon, *Reachability Analysis on Distributed Executions*, TAPSOFT'93: Theory and Practice of Software Development, in Lecture Notes in Computer Science N^o 668, Springer–Verlag 1993, pp. 629–643.

[9] C. Fidge, *Timestamps in Message Passing Systems that Preserve the Partial Ordering*, In Proc. 11^{th} Australian Computer Science Conference, 55–66, February 1988.

[10] B. Ganter, K. Reuter, *Finding All Closed Sets: A General Approach*, Order 8 : 283–290, 1991.

[11] M. Habib, M. Morvan, M. Pouzet, J.X. Rampon, *Incidence Structures, Coding and Lattice of Maximal Antichains*, R.R. N^o 92-079, LIRMM Montpellier, 1992.

[12] S. Irani, *Coloring Inductive Graphs On–Line*, I.E.E.E. 31nd Symposium on Foundations of Computer Science, pp. 470–479.

[13] C. Jard, G.V. Jourdan, J.X. Rampon, *Some "On–Line" Computations of the Ideal Lattice of Posets*, IRISA research report N^o 773, October 1993.

[14] L. Lamport, *Time, Clocks and the Ordering of Events in a Distributed System*, Communications of the ACM, 21(7): 558–565, July 1978.

[15] G. Markowsky, *The Factorization and Representation of Lattices*, Transactions of the American Mathematical Society *Volume* 203, 1975, 185–200.

[16] G. Markowsky, *Primes, Irreductibles and Extremal Lattices* Order 9 : 265–290, 1992.

[17] F. Mattern, *Virtual Time and Global States of Distributed Systems*, In Cosnard, Quinton, Raynal and Robert, editors, Proc. Int. Workshop on Parallel and Distributed Algorithms, Bonas France October 1988, North–Holland, 1989.

[18] M. Morvan, L. Nourine, *Generating Minimal Interval Extensions* R.R. N° 92-015, LIRMM Montpellier, 1992.

[19] K. Reuter, *The Jump Number and the Lattice of Maximal Antichains* Discrete Mathematics 88 (1991), 289–307.

[20] J. Westbrook, D. C. K. Yan *Greedy Algorithms for the On–Line Steiner Tree and Generalized Steiner Problems*, WADS'93: Algorithms and Data Structures, Lecture Notes in Computer Science N^o 709, Springer–Verlag 1993 (August 1993), pp. 622–633.

[21] R. Wille, *Restructuring lattice theory: an approach based on hierarchies of concepts* I. Rival ed., Ordered Sets (Reidel, Dordrecht, 1982), 445–470.

[22] R. Wille, *Finite Distributive Lattices as Concept Lattices* Atti Inc. Logica Mathematica (Siena) 2 (1985) 635–648.

Unité de recherche INRIA Lorraine, Technôpole de Nancy-Brabois, Campus scientifique,
615 rue de Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399