1–35
Preliminary version of a paper published by Kluwer Academic Publishers, Boston

Optimal narrowing of a block of sortings in optimal time

NOËLLE BLEUZEN-GUERNALEC noelle.bleuzen-guernalec@lim.univ-mrs.fr
ALAIN COLMERAUER alain.colmerauer@lim.univ-mrs.fr
Laboratoire d'Informatique de Marseille, ESA 6077, CNRS et Universités de Provence et de la Méditerranée

Editor:

Abstract. Let **D** be a totally ordered set. Call an n-block, a Cartesian product of n closed and possibly empty intervals of **D**. Let *sort* be the set of all $2n$-tuples of elements of **D** of the form $(x_1, \ldots, x_{2n})$, where $(x_{n+1}, \ldots, x_{2n})$ is the n-tuple obtained by sorting the elements of the n-tuple $(x_1, \ldots, x_n)$ in non-decreasing order.

We present and justify an algorithm of complexity $\mathcal{O}(n \log n)$ which, given a $2n$-block a, computes a $2n$-block which, by inclusion, is the smallest block containing the set $sort \cap a$. We show that this complexity is optimal.

Keywords: sorting, constraint, narrowing, interval, block, box

1. Introduction

Given a $2n$-tuple $(x_1, \ldots, x_{2n})$ of elements of a totally ordered set, the sortedness constraint

$$(x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n}) \in sort$$

expresses that the n-tuple $(x_{n+1}, \ldots, x_{2n})$ is equal to the n-tuple obtained by sorting the elements of the n-tuple $(x_1, \ldots, x_n)$ in non-decreasing order. W.J. Older, F. Swinkels and M. van Emden introduced this constraint for stating and solving problems of the job-shop scheduling type [4]. More recently Jianyang Zhou [5] has solved well known difficult job-shop scheduling problems by introducing a sortedness constraint with $3n$ variables, the n extra variables being used for making explicit a permutation linking the x_i's.

We are interested in the *optimal narrowing* of a *basic* sortedness constraint, which is a constraint of the form:

$$(x_1, \ldots, x_{2n}) \in sort \wedge x_1 \in a_1 \wedge \cdots \wedge x_{2n} \in a_{2n}, \tag{1}$$

where the a_i's are closed and possibly empty intervals of the domain of the x_i's. By "optimal narrowing" we mean: replacing the intervals a_i by the intervals $b_i \subseteq a_i$ which, by inclusion, are the smallest intervals which do not modify the set of solutions of the constraint, that is to say such that

$$sort \cap a_1 \times \cdots \times a_{2n} = sort \cap b_1 \times \cdots \times b_{2n}. \tag{2}$$

Notice that, in order to know whether the basic constraint has at least one solution, it suffices to check, after narrowing, that none of the new intervals is empty.

As BNR-Prolog has taught us, this narrowing applies to any kind of basic constraint and allows progress in the solving of a conjunction of basic constraints by narrowing, one after another, each basic constraint, until the intervals attached to the different variables reach a fixed point, see [3, 1].

If we let $a = a_1 \times \cdots \times a_{2n}$, the optimal narrowing of the constraint (1) consists in computing the $2n$-block b which, by inclusion, is the smallest block which obeys $sort \cap a = sort \cap b$. It can be shown that b is also the smallest block which contains the set $sort \cap a$. The purpose of this paper is to present and justify an algorithm, of complexity $\mathcal{O}(n \log n)$, for computing such a $2n$-block b. A simple remark shows that this complexity is optimal. The paper is organized in 11 sections:

Section 2 introduces the terminology and the core of the problem: computing the endpoints of the $2n$ projections of a set of $2n$-tuples, related to *sort*, in $\mathcal{O}(n \log n)$ elementary instructions. Sections 3,4,5,6,7 establish four theorems for obtaining the least elements of the first n projections and the greatest elements of the last n projections. Section 8 presents a dual principle for obtaining the other endpoints and describes in detail the narrowing algorithm. Section 9 justifies the algorithm by using the four preceding theorems and section 10 details the algorithm, using an example. We conclude by additional information about the algorithm.

2. Statement of the problem

2.1. Global relation

Let $\mathbf{D}$ be an arbitrary set. We denote by $\mathbf{D}^\star$ the set of all finite sequences of elements of $\mathbf{D}$, including the empty sequence ε. All finite sequences of length $n > 0$ are written as n-tuples $(d_1, \ldots, d_n)$, with the d_i's taken in $\mathbf{D}$. We denote by $\mathbf{D}^n$ the set of sequences of $\mathbf{D}^\star$ which are of length n. If $d \in \mathbf{D}^\star$ then $set\,(d)$ denotes the set of elements of $\mathbf{D}$ occurring in d. The concatenation operation between the elements of $\mathbf{D}^\star$ is written with a dot. In particular, we have $d \cdot \varepsilon = \varepsilon \cdot d = d$, for all d in $\mathbf{D}^\star$.

While an *n-ary relation on* $\mathbf{D}$ is a subset of $\mathbf{D}^n$, a *global relation on* $\mathbf{D}$ is a subset r of $\mathbf{D}^\star$. For any integer $i > 0$, the *ith projection* of r is the (possibly empty) subset of $\mathbf{D}$:

$$\pi_i(r) = \{e \in \mathbf{D} \,|\, \text{there exists } (d_1, \ldots, d_n) \in r \text{ with } n \geq i \text{ and } e = d_i\}.$$

2.2. Intervals and blocks

Generally speaking, if $(\mathbf{D}, \preceq)$ is a totally ordered set with at least two elements, we call an *interval of* $(\mathbf{D}, \preceq)$ any (possibly empty) subset of $\mathbf{D}$, which is of the form

$$\{d \in \mathbf{D} \,|\, e \preceq d \text{ and } d \preceq e'\},$$

where e and e' are any elements of $\mathbf{D}$. We write it as $[e, e']$. The fact that we do not impose $e \preceq e'$ and that the set $\mathbf{D}$ has more than one element, ensures that the empty set is also an interval. For the ordered set $(\mathbf{N}, \leq)$ of natural integers, we write $e..e'$ instead of $[e, e']$.

A *block of dimension* n, or *n-block*, on $(\mathbf{D}, \preceq)$ is a subset a of $\mathbf{D}^n$ of the form $a_1 \times \cdots \times a_n$, with $n \geq 1$, where each a_i is an interval of $(\mathbf{D}, \preceq)$. Since the a_i's can be empty, the empty set $\emptyset$ is a block of dimension n for any n. Notice that the n-tuple $(a_1, \ldots, a_n) = (\pi_1(a), \ldots, \pi_n(a))$ is such that $a = a_1 \times \cdots \times a_n$ and, if a is not empty, the intervals a_i's such that $a = a_1 \times \cdots \times a_n$ are unique.

Let a be a subset of $\mathbf{D}$. We denote by $\underline{a}$ and $\overline{a}$ the least element and the greatest element of a in $(\mathbf{D}, \preceq)$, when they exist. We keep the notations $\mathit{min}\, a$ and $\mathit{max}\, a$ for the ordered set $(\mathbf{N}, \leq)$. We denote by $\mathit{apx}(a)$ the least interval of $(\mathbf{D}, \preceq)$ which contains a, if it exists. The notion of least is taken with respect to the inclusion relation. Clearly, if $\underline{a}$ and $\overline{a}$ exist, $\mathit{apx}(a)$ exists and is equal to $[\underline{a}, \overline{a}]$. We extend the $\mathit{apx}(a)$ notation as follows:

Definition 1. If r is a subset of $\mathbf{D}^n$, we denote by $\mathit{apx}(r)$ the least n-block on $(\mathbf{D}, \preceq)$ which contains r, if it exists.

Here also the notion of least is taken with respect to the inclusion relation. If s is a subset of $\mathbf{D}^n$, it is obvious that:

PROPERTY 1 *Either the sets* $\mathit{apx}(s)$ *and* $\mathit{apx}(\pi_1(s)) \times \cdots \times \mathit{apx}(\pi_n(s))$ *exist and are equal, or neither of the sets exist.*

Finally, we say that an n-tuple $(d_1, \ldots, d_n)$ is *increasing for* $\preceq$, if $d_i \preceq d_{i+1}$, for each $i \in 1..(n-1)$.

2.3. The problem

Throughout this paper, $(\mathbf{D}, \preceq)$ is any totally ordered set. In particular $(\mathbf{D}, \preceq)$ can be the ordered set $(\mathbf{R}, \leq)$ of reals, the ordered set $(\mathbf{Z}, \leq)$ of integers, the ordered

set $(\mathbf{N}, \leq)$ of natural integers or any subset of $\mathbf{R}$ which is ordered by $\leq$. As usual we write $\prec$ for the relation $\preceq$ without pairs of equal elements. We denote by *sort* the global relation on $\mathbf{D}$ defined by:

$$d \in \mathit{sort} \iff \begin{pmatrix} d \text{ is of the form } (d_1, \ldots, d_{2n}),\ n \geq 1, \\ (d_{n+1}, \ldots, d_{2n}) \text{ is increasing for } \preceq \text{ and} \\ \text{there exists a one to one mapping } f \\ \text{from } 1..n \text{ to } (n+1)..(2n) \text{ such that} \\ (d_1, \ldots, d_n) = (d_{f(1)}, \ldots, d_{f(n)}) \end{pmatrix}$$

For example, in $(\mathbf{Z}, \leq)$, we have

$$(5, 12, 10, 5, 12, 3,\ 3, 5, 5, 10, 12, 12) \in \mathit{sort}.$$

Thus the problem is, given a $2n$-block a on $(\mathbf{D}, \preceq)$, to show that the set $\mathit{apx}(\mathit{sort} \cap a)$ always exists and to compute it by an algorithm of optimal complexity $\mathcal{O}(f(n))$.

As usually, by an algorithm of complexity $\mathcal{O}(f(n))$, we understand that there exists a strictly positive real c, independent on n, such that for any n, the number of executed elementary instructions is at most $cf(n)$. For an algorithm of *optimal* complexity $\mathcal{O}(f(n))$, we request, in addition, that there exists another strictly positive real c', independent on n, such that for any n, any other algorithm, which computes $\mathit{apx}(\mathit{sort} \cap a)$, executes a number of elementary instructions greater than $c' f(n)$ for at least one value of a.

The elementary instructions are those of a virtual machine which has an infinite number of cells, each one being able to contain any element of $\mathbf{D}$ or $\mathbf{N}$. Without going into the details, these elementary operations include: moving the content of a cell into another cell, comparison of two elements of $\mathbf{D}$ and comparison of two elements of $\mathbf{N}$.

If for a we take a $2n$-block of the form $[e_1, e_1] \times \cdots \times [e_n, e_n] \times [d_1, d_2] \times \cdots \times [d_1, d_2]$, with $d_1 = \underline{\{e_1, \ldots, e_n\}}$ and $d_2 = \overline{\{e_1, \ldots, e_n\}}$, the computation of $\mathit{apx}(\mathit{sort} \cap a)$ amounts to sorting the n-tuple $(e_1, \ldots, e_n)$. Such a sorting needs at least $c' n \log n$ comparisons of the d_i's to handle the worst case. (See section 5.3.1 in Donald E. Knuth [2].) It follows that an algorithm for computing $\mathit{apx}(\mathit{sort} \cap a)$, with complexity $\mathcal{O}(n \log n)$, will have an optimal complexity.

For the computation of $\mathit{apx}(\mathit{sort} \cap a)$, we will use Property 1. If we let $s = \mathit{sort} \cap a$, this computation amounts then to the computation of the $2n$ intervals $\mathit{apx}(\pi_i(s))$, which itself amounts to the computation of their endpoints $\underline{\pi_i(s)}$ and $\overline{\pi_i(s)}$.

EXAMPLE: If we take $n = 11$ and the $2n$-block on $(\mathbf{Z}, \leq)$

$a =$
$[10, 24] \times [18, 26] \times [18, 22] \times [18, 22] \times [32, 38] \times [32, 42] \times [16, 28] \times [10, 28] \times$
$[6, 46] \times [0, 46] \times [0, 44] \times$
$[0, 4] \times [8, 16] \times [8, 14] \times [18, 27] \times [18, 25] \times [14, 24] \times [26, 32] \times [32, 42] \times$
$[32, 42] \times [42, 48] \times [40, 48],$

the algorithm must compute the 22-block

$$apx\,(sort \cap a) =$$
$$[10,14]\times[18,26]\times[18,22]\times[18,22]\times[32,38]\times[32,42]\times[16,28]\times[10,14]\times$$
$$[32,46]\times[0,46]\times[0,44]\times$$
$$[0,4]\times[10,14]\times[10,14]\times[18,22]\times[18,22]\times[18,24]\times[26,28]\times[32,38]\times$$
$$[32,42]\times[42,46]\times[42,46].$$

The transformation $a \Longrightarrow apx\,(sort \cap a)$ is illustrated in figure 1. The intervals are

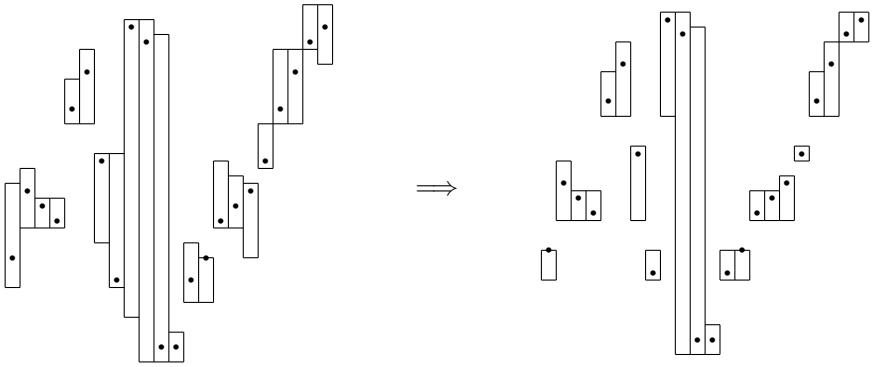

Figure 1. Transformation $a \Longrightarrow apx\,(sort \cap a)$, with a of dimension $11 + 11$.

drawn as rectangles and the dots inside each of them represent a randomly chosen element of $sort \cap a$, here the 22-tuple

$$(14, 23, 21, 19, 34, 39, 27, 11, 45, 43, 2)\cdot(2, 11, 14, 19, 21, 23, 27, 34, 39, 43, 45).$$

□

3. Normalized block

We show here how to transform a block a on $(\mathbf{D}, \preceq)$ into an empty or normalized block b such that $sort \cap a = sort \cap b$. We also present the two essential properties of normalized blocks.

Definition 2. A *normalized* $2n$-block is a non-empty $2n$-block on $(\mathbf{D}, \preceq)$ of the form $a_1 \times \cdots \times a_{2n}$, with the n-tuples $(\underline{a_{n+1}}, \ldots, \underline{a_{2n}})$ and $(\overline{a_{n+1}}, \ldots, \overline{a_{2n}})$ increasing for $\preceq$.

3.1. Normalization

If $(d_1, \ldots, d_n)$ is an element of $\mathbf{D}^n$, the n-tuples $(e_1, \ldots, e_n) = \uparrow (d_1, \ldots, d_n)$ and $(e'_1, \ldots, e'_n) = \downarrow (d_1, \ldots, d_n)$ are defined as follows:

$$e_1 = d_1 \quad \text{and} \quad e_{i+i} = \overline{\{e_i, d_{i+1}\}}, \text{ for each } i \text{ in } 1..(n-1),$$
$$e'_n = d_n \quad \text{and} \quad e'_{i-1} = \underline{\{e'_i, d_{i-1}\}}, \text{ for each } i \text{ in } 2..n.$$

From the definition of the global relation *sort* it follows immediately:

PROPERTY 2 *Let* $a = a_1 \times \cdots \times a_{2n}$ *be a non-empty* $2n$*-block on* $(\mathbf{D}, \preceq)$ *and let*

$$b = a_1 \times \cdots \times a_n \times [e_1, e'_1] \times \cdots \times [e_n, e'_n], \quad \textit{with} \quad \begin{array}{rcl}(e_1, \ldots, e_n) & = & \uparrow(\underline{a_{n+1}}, \ldots, \underline{a_{2n}}),\\ (e'_1, \ldots, e'_n) & = & \downarrow(\overline{a_{n+1}}, \ldots, \overline{a_{2n}}).\end{array}$$

Then $sort \cap a = sort \cap b$ *and block* b *is either empty or normalized.*

EXAMPLE: If we take the $2n$-block in $(\mathbf{R}, \leq)$

$a =$
$[0,13] \times [6,10] \times [10,11] \times [4,16] \times [4,6] \ \times \ [1,3] \times [5,10] \times [6,9] \times [11,17] \times [10,15],$

with $n = 5$, we have

$$\begin{array}{rcl}\uparrow(1,5,6,11,10) & = & (1,5,6,11,11),\\ \downarrow(3,10,9,17,15) & = & (3,9,9,15,15)\end{array}$$

and thus the block

$$b \ = \ [0,13] \times [6,10] \times [10,11] \times [4,16] \times [4,6] \ \times \ [1,3] \times [5,9] \times [6,9] \times [11,15] \times [11,15]$$

obeys $sort \cap a = sort \cap b$ and is normalized, since it is not empty. If we now take the $2n$-block

$a =$
$[3,3] \times [1,3] \times [1,9] \times [2,7] \times [5,6] \times [6,6] \times [5,6] \times [5,10] \times [3,11] \times [9,11] \times [9,9] \times$
$[1,4] \times [1,2] \times [2,4] \times [4,7] \times [7,8] \times [5,7] \times [4,6] \times [5,8] \times [8,10] \times [7,11] \times [9,11],$

we have

$$\begin{array}{rcl}\uparrow(1,1,2,4,7,5,4,5,8,7,9) & = & (1,1,2,4,7,7,7,7,8,8,9),\\ \downarrow(4,2,4,7,8,7,6,8,10,11,11) & = & (2,2,4,6,6,6,6,8,10,11,11)\end{array}$$

and thus the block

$b =$
$[3,3] \times [1,3] \times [1,9] \times [2,7] \times [5,6] \times [6,6] \times [5,6] \times [5,10] \times [3,11] \times [9,11] \times [9,9] \times$
$[1,2] \times [1,2] \times [2,4] \times [4,6] \times [7,6] \times [7,6] \times [7,6] \times [7,8] \times [8,10] \times [8,11] \times [9,11]$

verifies $sort \cap a = sort \cap b$. Due to the occurrence of the empty interval $[7,6]$, we have $b = \emptyset$, thus $sort \cap a = \emptyset$ □

COMPLEXITY 1 Clearly, the normalization of a $2n$-block a can be performed in $\mathcal{O}(n)$ elementary instructions.

3.2. Set of relevant indices

Let a be a $2n$-block on $(\mathbf{D}, \preceq)$. The set of *relevant indices of a for i in $1..n$* is

$$rel\,(a,i) = \{j \in (n+1)..(2n) \,|\, a_i \cap a_j \neq \emptyset\}$$

and the set of *relevant indices of a for j in $(n+1)..(2n)$* is

$$rel\,(a,j) = \{i \in 1..n \,|\, a_i \cap a_j \neq \emptyset\}$$

By noticing that, for non-empty intervals, we have $a_i \cap a_j \neq \emptyset$ if and only if we have $\underline{a_i} \preceq \overline{a_j}$ and $\underline{a_j} \preceq \overline{a_i}$, it can be verified that:

PROPERTY 3 *If $a = a_1 \times \cdots \times a_{2n}$ is a normalized $2n$-block and i an element of $1..n$, the set $rel\,(a,i)$ is an interval of $(\mathbf{N}, \leq)$ and, if this interval is not empty, we have*

$$\begin{aligned} min\,(rel\,(a,i)) &= min\,\{j \in (n+1)..(2n) \,|\, \underline{a_i} \preceq \overline{a_j}\}, \\ max\,(rel\,(a,i)) &= max\,\{j \in (n+1)..(2n) \,|\, \underline{a_j} \preceq \overline{a_i}\}. \end{aligned}$$

3.3. First projections of sort $\cap$ $\boldsymbol{a}$

Here is the second essential property of a normalized block:

PROPERTY 4 *If $a = a_1 \times \cdots \times a_{2n}$ is a normalized $2n$-block and if Φ denotes the set of bijections f from $1..n$ to $(n+1)..(2n)$ such that $a_1 \times \cdots \times a_n \cap a_{f(1)} \times \cdots \times a_{f(n)} \neq \emptyset$ then, for any $i \in 1..n$, we have*

$$\pi_i(sort \cap a) = \bigcup_{f \in \Phi} a_i \cap a_{f(i)}. \tag{3}$$

Proof: Let e be an element of $\mathbf{D}$. By definition of *sort*, if e belongs to the left member of (3), it also belongs to the right member of (3). Let us suppose that e belongs to the right member of (3) and let us show that it belongs to the left member of (3). Then there exists a bijection f from $1..n$ to $(n+1)..(2n)$ such that the set $b = a_1 \times \cdots \times a_n \cap a_{f(1)} \times \cdots \times a_{f(n)}$ is not empty and $e \in a_i \cap a_{f(i)}$. There exists an n-tuple $(d_1, \ldots, d_n) \in b$ such that $e = d_i$. Let d be the $2n$-tuple $(d_1, \ldots, d_{2n})$ obtained by letting $(d_{f(1)}, \ldots, d_{f(n)}) = (d_1, \ldots, d_n)$. Clearly $d \in a$. Let (j, j') be a pair of indices, taken in $(n+1)..(2n)$, such that $j < j'$ and $d_{j'} \prec d_j$. Since a is normalized, the $2n$-tuple, obtained from d by swapping d_j and $d_{j'}$, still belongs to a. By a finite number of such operations, it is then possible to put in increasing $\preceq$ order the last n terms of the $2n$-tuple d, eventually obtaining an element of $sort \cap a$. Since we still have $e = d_i$, the element e belongs indeed to the left member of (3). ■

4. Block of type I

Property 4 will allow us to compute the least elements of the first n projections of $sort \cap a$ in $\mathcal{O}(n \log n)$ elementary instructions, provided that the $2n$-block a is in a special form, called type I. The transformation of a normalized block into a block of type I is not straightforward. As we will see later, the block first must be transformed into a block of type II, then into a block of type III, and finally decomposed into several blocks of type I.

Definition 3. A $2n$-block *of type I* is a normalized block $a_1 \times \cdots \times a_{2n}$, such that $a_1 \times \cdots \times a_n \cap a_{n+1} \times \cdots \times a_{2n} \neq \emptyset$ and $a_1 \times \cdots \times a_{n-1} \cap a_{n+2} \times \cdots \times a_{2n} \neq \emptyset$.

4.1. Computing the least elements of the first n projections of $sort \cap a$

THEOREM 1 *If $a = a_1 \times \cdots \times a_{2n}$ is a $2n$-block of type I, then, for each $i \in 1..n$, the element $\underline{\pi_i(sort \cap a)}$ exists and is equal to $\underline{a_i \cap a_j}$, with $j = min\,(rel\,(a, i))$.*

Proof: Let $a = a_1 \times \cdots \times a_{2n}$ be a $2n$-block of type I. Pick element i in $1..n$. Since $a_i \cap a_{i+n} \neq \emptyset$, the index $j = min\,(rel\,(a, i))$ and thus the element $\underline{a_i \cap a_j}$ exist.

Let us show that $\underline{a_i \cap a_j} \in \pi_i(sort \cap a)$. The block a being of type I, the mapping from $1..n$ to $(n+1)..(2n)$ defined by

$$i' \mapsto \begin{cases} i' + n, & \text{if } i' \notin (j-n)..i, \\ i' + n + 1, & \text{if } i' \in (j-n)..(i-1), \\ j, & \text{if } i' = i \end{cases}$$

belongs to the set Φ of one to one mappings f from $1..n$ to $(n+1)..(2n)$, such that $a_1 \times \cdots \times a_n \cap a_{f(1)} \times \cdots \times a_{f(n)} \neq \emptyset$. By Property 4, we have $a_i \cap a_j \subseteq \pi_i(sort \cap a)$, thus $\underline{a_i \cap a_j} \in \pi_i(sort \cap a)$.

Let us show that $e \in \pi_i(sort \cap a)$ implies $\underline{a_i \cap a_j} \preceq e$. If $e \in \pi_i(sort \cap a)$, by definition of $sort$, there exists $f \in \Phi$ such that $e \in a_i \cap a_{f(i)}$, thus there exists $j' \in (n+1)..(2n)$ such that $a_i \cap a_{j'} \neq \emptyset$ and $\underline{a_i \cap a_{j'}} \preceq e$. The block a being normalized, we have $\underline{a_i \cap a_j} \preceq e$. ■

EXAMPLE: The $2n$-block on $(\mathbf{R}, \leq)$

$$a \;=\; [0,13] \times [10,11] \times [4,16] \times [1,3] \times [11,11] \times [11,15]$$

is of type I. According to Theorem 1, we get

$$\begin{array}{lcll} \underline{\pi_1(sort \cap a)} & = & \underline{[0,13] \cap [1,3]} & = \; 1, \\ \underline{\pi_2(sort \cap a)} & = & \underline{[10,11] \cap [11,11]} & = \; 11, \\ \underline{\pi_3(sort \cap a)} & = & \underline{[4,16] \cap [11,11]} & = \; 11. \end{array}$$

□

COMPLEXITY 2 The computation of all $\underline{\pi_i(sort \cap a)}$, with $i \in 1..n$, can be performed in $\mathcal{O}(n \log n)$ elementary instructions. According to Property 3, to each $i \in 1..n$, we must associate the index

$$j_i = min\,\{j \in (n+1)..(2n) \mid \underline{a_i} \preceq \overline{a_j}\}.$$

We first sort the n-tuple of pairs $((1, a_1), \ldots, (n, a_n))$ in increasing order (for $\preceq$) of the $\underline{a_i}$'s. We obtain in $\mathcal{O}(n \log n)$ elementary instructions an n-tuple of the form $((h_1, a_{h_1}), \ldots, (h_n, a_{h_n}))$. The computations of the j_i's can then be performed in $\mathcal{O}(n)$ elementary instructions by scanning once and simultaneously the n-tuples $((h_1, a_{h_1}), \ldots, (h_n, a_{h_n}))$ and $((n+1, a_{n+1}), \ldots, (2n, a_{2n}))$ from left to right.

5. Block of type II

Given a $2n$-normalized block a, we study here how to decide if the set $sort \cap a$ is not empty and, if not, how to compute the greatest elements of its last n projections. For simplicity, we suppose that a is in a special form, called of type II, to which it is easy to move.

Definition 4. A $2n$-block *of type II* is a normalized block $a_1 \times \cdots \times a_{2n}$ with the n-tuple $(\overline{a_1}, \ldots, \overline{a_n})$ increasing for $\preceq$.

5.1. Preliminary results

PROPERTY 5 *Let $a = a_1 \times \cdots \times a_{2n}$ be a $2n$-block and φ a bijection from $1..n$ into $(n+1)..(2n)$. The three following properties are equivalent:*

$$\varphi(i) = min\,(rel\,(a, i) - \varphi(1..(i-1))), \quad \textit{for all } i \in 1..n, \tag{4}$$

$$\varphi^{-1}(j) = min\,(rel\,(a, j) - \varphi^{-1}((n+1)..(j-1))), \quad \textit{for all } j \in (n+1)..(2n), \tag{5}$$

$$\begin{array}{l} a_i \cap a_{\varphi(i)} \neq \emptyset, \quad \textit{for all } i \in 1..n, \textit{ and} \\ a_i \cap a_{\varphi(i')} \neq \emptyset, \quad i < i', \quad \varphi(i) > \varphi(i'), \quad \textit{for no pair } (i, i') \textit{ of elements in } 1..n. \end{array} \tag{6}$$

Here $\varphi(1..(i-1))$ denotes the set $\{\varphi(1), \ldots, \varphi(i-1)\}$ and is empty when $i = 1$. The same holds for $\varphi^{-1}((n+1)..(j-1))$.

Proof: By taking into account that φ is a bijection, property (4) can be rewritten as follows:

$$\begin{array}{l} \text{for all } i \in 1..n, \\ \varphi(i) \in rel\,(a, i) \text{ and, for all } j \in rel\,(a, i) - \varphi(1..(i-1)), \ \varphi(i) \leq j, \end{array}$$

that is, by letting $i' = \varphi^{-1}(j)$,

for all $i \in 1..n$,
$\varphi(i) \in rel\,(a,i)$ and, for all $i' \in i..n$, $\varphi(i') \in rel\,(a,i) \Longrightarrow \varphi(i) \leq \varphi(i')$.

which is nothing else than property (6). Similarly property (5) can be written as:

for all $j \in (n+1)..(2n)$,
$\varphi^{-1}(j) \in rel\,(a,j)$ and, for all $i' \in rel\,(a,j) - \varphi^{-1}((n+1)..(j-1))$, $\varphi^{-1}(j) \leq i'$,

that is, by letting $i = \varphi^{-1}(j)$,

for all $i \in 1..n$,
$i \in rel\,(a,\varphi(i))$ and, for all $i' \in rel\,(a,\varphi(i))$, $\varphi(i') \in \varphi(i)..2n \Longrightarrow i \leq i'$,

which is nothing else than property (6). ■

LEMMA 1 *If $a = a_1 \times \cdots \times a_{2n}$ is a $2n$-block of type II and if it is possible to define a one to one mapping φ from $1..n$ to $(n+1)..(2n)$ such that, for each $i \in 1..n$,*

$$\varphi(i) = min\,(rel\,(a,i) - \varphi(1..(i-1))), \tag{7}$$

then the $2n$-tuple $e = (e_1, \ldots, e_{2n})$, with $e_{\varphi(i)} = e_i$ and $e_i = \overline{a_i \cap a_{\varphi(i)}}$, for each $i \in 1..n$, belongs to $sort \cap a$.

Proof: Let us suppose that the property is not true and let us show that we are led to a contradiction. By construction, we have $e \in a$. Thus there exists a pair (i,i') of elements of $1..n$ such that $\varphi(i) < \varphi(i')$ and $e_{i'} \prec e_i$. Since a is normalized, we have $\overline{a_{\varphi(i)}} \preceq \overline{a_{\varphi(i')}}$ and thus $\overline{a_{i'}} \prec \overline{a_i}$. Since a is of type II, we have $i > i'$. According to Property 3, equality (7) can be replaced by

$$a_i \cap a_{\varphi(i)} \neq \emptyset,\ \varphi(i) = min\,\{j \in (n+1)..(2n) \mid \underline{a_i} \preceq \overline{a_j} \text{ and } j \notin \varphi(1..(i-1))\}.$$

We have $i > i'$ and $\varphi(i) < \varphi(i')$. It follows that $\overline{a_{i'}} \prec \underline{a_{\varphi(i)}}$. Since a is normalized and $\varphi(i) < \varphi(i')$ we have also $\underline{a_{\varphi(i)}} \preceq \underline{a_{\varphi(i')}}$. By transitivity, we obtain $\overline{a_{i'}} \prec \underline{a_{\varphi(i')}}$, which contradicts the fact that $a_{i'} \cap a_{\varphi(i')} \neq \emptyset$. ■

LEMMA 2 *Let $a = a_1 \times \cdots \times a_{2n}$ be a $2n$-block of type II such that the set $sort \cap a$ has at least one element $(d_1, \ldots, d_{2n})$. It is then possible to define a one to one mapping φ from $1..n$ to $(n+1)..(2n)$, such that,*

$$\varphi(i) = min\,(rel\,(a,i) - \varphi(1..(i-1))), \quad \textit{for each } i \in 1..n, \tag{8}$$

$$d_{\varphi(i)} \preceq \overline{a_i \cap a_{\varphi(i)}}, \quad \textit{for each } i \in 1..n. \tag{9}$$

Proof: By Property 5, the condition (8) can be written as the conjunction of two properties:

$$a_i \cap a_{\varphi(i)} \neq \emptyset, \quad \text{for each element } i \text{ of } 1..n, \tag{10}$$

$$a_i \cap a_{\varphi(i')} \neq \emptyset,\ i < i',\ \varphi(i) > \varphi(i'),\ \text{for no pair } (i,i') \text{ of elements in } 1..n. \tag{11}$$

Since $(d_1,\ldots,d_{2n}) \in sort \cap a$, there exists a one to one mapping φ from $1..n$ to $(n+1)..(2n)$ which has properties (10) and (9). Let us modify this mapping, as many times as possible, by the following transformation: choose in $1..n$ a pair (i,i') of indices such that

$$a_i \cap a_{\varphi(i')} \neq \emptyset \ \text{ and } \ i < i' \ \text{ and } \varphi(i) > \varphi(i') \tag{12}$$

and replace φ by the one to one mapping φ' from $1..n$ to $(n+1)..(2n)$ which differs from φ only by the fact that $(\varphi'(i),\varphi'(i')) = (\varphi(i'),\varphi(i))$. Since after each transformation, the integer $\Sigma_{i=1}^{n} i \times \varphi(i)$ strictly increases, but remains less or equal to n^3, this process terminates and produces a final mapping φ which has property (11). It remains to be shown that this final mapping has properties (10) and (9). As the initial mapping φ has these properties, it is sufficient to show that these two properties are preserved after each transformation of φ.

Thus let φ be a one to one mapping from $1..n$ to $(n+1)..(2n)$ which has properties (10) and (9), and let (i,i') be a pair of elements of $1..n$ fulfilling (12). We must show that

$$\varphi(i') \in rel\,(a,i), \quad \varphi(i) \in rel\,(a,i'), \quad d_{\varphi(i')} \preceq \overline{a_i \cap a_{\varphi(i')}}, \quad d_{\varphi(i)} \preceq \overline{a_{i'} \cap a_{\varphi(i)}}.$$

By property (12) we already have $\varphi(i') \in rel\,(a,i)$. Properties (10) and (12), the fact that a is of type II and Property 3 alltogether imply that

$$min\,(rel\,(a,i')) \leq \varphi(i') < \varphi(i) \leq max\,(rel\,(a,i)) \leq max\,(rel\,(a,i')).$$

The set $rel\,(a,i')$ being an interval, thus $\varphi(i) \in rel\,(a,i')$. From the fact that $(d_1,\ldots,d_{2n}) \in sort$, from properties (9) and (12) and from the fact that a is of type II, we conclude that

$$d_{\varphi(i')} \preceq d_{\varphi(i)} \preceq \overline{a_i} \preceq \overline{a_{i'}}.$$

From this and property (9) we deduce that $d_{\varphi(i')} \preceq \overline{a_i \cap a_{\varphi(i')}}$ and $d_{\varphi(i)} \preceq \overline{a_{i'} \cap a_{\varphi(i)}}$. ■

5.2. Computing the greatest elements of the last $\boldsymbol{n}$ *projections of* $sort \cap \boldsymbol{a}$

Lemmas 1 and 2 exhibit a particular bijection φ, which is constructed by successively associating the least still available relevant index to each element i of $1..n$. The theorem which follows shows the importance of this mapping φ, which, by construction, is unique if it exists.

THEOREM 2 *Let $a = a_1 \times \cdots \times a_{2n}$ be a block of type II and, if is possible to define it, let φ be a one to one mapping from $1..n$ to $(n+1)..(2n)$ such that, for each $i \in 1..n$,*

$$\varphi(i) = min\,(rel\,(a,i) - \varphi(1..(i-1))). \tag{13}$$

Then
(1) there exists at most one such mapping φ,
(2) if the set $sort \cap a$ is not empty, φ exists,
(3) if φ exists, the set $sort \cap a$ is not empty and, for each $j \in (n+1)..(2n)$, the element $\overline{\pi_j(sort \cap a)}$ exists and is equal to $\overline{a_j \cap a_{\varphi^{-1}(j)}}$.

Proof: Since $\varphi(1) = min\,(rel\,(a,1))$ and since, for each $i \in 2..(n-1)$, the value of $\varphi(i)$ depends only on $\varphi(1), \ldots, \varphi(i-1)$, there exists at most one mapping φ.

If the set $sort \cap a$ is not empty, it has at least one element $(d_1, \ldots, d_{2n})$, thus by Lemma 2, the mapping φ exists.

If the mapping φ exists, then according to Lemma 1, the $2n$-tuple $(e_1, \ldots, e_{2n})$, with $e_i = e_{\varphi(i)} = \overline{a_i \cap a_{\varphi(i)}}$, for each $i \in 1..n$, belongs to $sort \cap a$. Moreover, according to Lemma 2, if $(d_1, \ldots, d_{2n})$ is an element of $sort \cap a$, for each $i \in 1..n$, we have $d_{\varphi(i)} \preceq \overline{a_i \cap a_{\varphi(i)}}$, that is $d_{\varphi(i)} \preceq e_i$. Thus the element $\overline{\pi_{\varphi(i)}(sort \cap a)}$ exists and is equal to $\overline{a_i \cap a_{\varphi(i)}}$. By letting $j = \varphi(i)$, it follows that, for each $j \in (n+1)..(2n)$, the element $\overline{\pi_j(sort \cap a)}$ exists and is equal to $\overline{a_j \cap a_{\varphi^{-1}(j)}}$. ■

EXAMPLE: For $n = 5$ let us take the $2n$-block on $(\mathbf{Z}, \leq)$

$$a = [4,6] \times [6,10] \times [10,11] \times [7,13] \times [4,16] \times [1,3] \times [5,9] \times [6,9] \times [11,15] \times [11,15].$$

This block is of type II. We construct from left to right the longest sequence $\varphi(1), \varphi(2), \ldots, \varphi(m)$ satisfying, for each $i \in 1..m$,

$$\varphi(i) = min\,(rel\,(a,i) - \varphi(1..(i-1))).$$

In particular

$$\varphi(1) = min\,(rel\,(a,1)) = 7.$$

Continuing, we get

$$(\varphi(1), \ldots, \varphi(4)) = (7,8,9,10)$$

Thus $m = 4$. Since $m < n$, the mapping φ is not defined on $1..n$, therefore $sort \cap a = \emptyset$. However, if we take the $2n$-block

$$a = [4,6] \times [6,10] \times [10,11] \times [0,13] \times [4,16] \times [1,3] \times [5,9] \times [6,9] \times [11,15] \times [11,15],$$

the same construction gives

$$(\varphi(1), \ldots, \varphi(5)) = (7,8,9,6,10).$$

The sequence being of length n, we have constructed the bijection φ which satisfies condition (13). Hence the set $r = sort \cap a$ is not empty. We then have

$\varphi^{-1} = \{6 \mapsto 4,\ 7 \mapsto 1,\ 8 \mapsto 2,\ 9 \mapsto 3,\ 10 \mapsto 5\}$

and

$(\overline{\pi_6(r)}, \ldots, \overline{\pi_{10}(r)}) = (3, 6, 9, 11, 15).$

□

COMPLEXITY 3 It is possible to decide if the mapping φ exists and, if so, to compute φ, in $\mathcal{O}(n \log n)$ elementary instructions. In fact we must construct the longest sequence $\varphi(1), \varphi(2), \ldots, \varphi(m)$ which, for each $i \in 1..m$, satisfies condition (13). As in the proof of Lemma 1, condition (13) can be replaced by

$$a_i \cap a_{\varphi(i)} \neq \emptyset,\ \ \varphi(i) = \mathit{min}\,\{j \in (n{+}1)..(2n) \mid \underline{a_i} \preceq \overline{a_j} \text{ and } j \notin \varphi(1..(i{-}1))\}.$$

Given the set $c = (n{+}1)..(2n)$ and knowing that the sequence $(\overline{a_{n+1}}, \overline{a_{n+2}}, \ldots, \overline{a_{2n}})$ is increasing for $\preceq$, we must successively give the values $1, 2, \ldots, n$ to i and each time, in $\mathcal{O}(\log n)$ elementary instructions,

- decide if there exists an element j of c such that $\overline{a_i} \preceq \underline{a_j}$ and, if it is the case, compute the least element $\varphi(i)$ of c,
- test if the set $a_i \cap a_{\varphi(i)}$ is not empty,
- remove $\varphi(i)$ from c.

This is possible by representing the initial set c by a balanced binary tree of n nodes. Each node p is labeled by an element $\mathit{label}\,(p)$ of c. The relation $<$ is encoded by requiring that, for each pair (p, q) of nodes, if q is accessible from the left daughter of p, then $\mathit{label}\,(q) < \mathit{label}\,(p)$ and, if q is accessible from the right daughter of p, then $\mathit{label}\,(p) < \mathit{label}\,(q)$.

By reasonably extending the set of elementary operations of our virtual machine, we know that the construction of such a tree can be performed in $\mathcal{O}(n)$ elementary instructions, and that searching for a node or deletion of a node in such a tree can be performed in $\mathcal{O}(\log n)$ elementary instructions. Of course the emptiness test of $a_i \cap a_{\varphi(i)}$ can be performed in a constant number of elementary operations.

6. Block of type III

The mapping φ is the basic tool for computing the greatest elements of the last n projections of $\mathit{sort} \cap a$, for a $2n$-block a of type II. It will also be used for transforming a block of type II into a block of type III, which can then be decomposed into several $2m$-blocks b of type I. Recall that we know how to compute the least elements of the first m projections of $\mathit{sort} \cap b$, when b is of type I. Before presenting the decomposition itself, we will establish two results: one about the sorting of the elements of $(n+1)..(2n)$ in an ad hoc order, the second about using this sorting for partitioning the set $1..2n$ of indices into subsets, which are said to be autonomous.

Definition 5. A $2n$-block *of type III* is a normalized $2n$-block of the form $a_1 \times \cdots \times a_{2n}$, for which $a_1 \times \cdots \times a_n \cap a_{n+1} \times \cdots \times a_{2n} \neq \emptyset$ and for which there exists no pair (j, j') of elements of $(n+1)..(2n)$ such that
$j < j'$ and $\overline{a_{j'-n}} \prec \overline{a_{j-n}}$ and $a_{j'-n} \cap a_j \neq \emptyset$.

6.1. Transforming a block of type II into a block of type III

PROPERTY 6 *Let $a = a_1 \times \cdots \times a_{2n}$ be a block of type II for which there exists a bijective mapping φ from $1..n$ to $(n+1)..(2n)$ such that, for each $i \in 1..n$, we have*

$$\varphi(i) = \mathit{min}\,(\mathit{rel}\,(a, i) - \varphi(1..(i-1))).$$

The block

$$a_{\varphi^{-1}(n+1)} \times \cdots \times a_{\varphi^{-1}(2n)} \times [\underline{a_{n+1}}, \overline{a_{n+1} \cap a_{\varphi^{-1}(n+1)}}] \times \cdots \times [\underline{a_{2n}}, \overline{a_{2n} \cap a_{\varphi^{-1}(2n)}}] \qquad (14)$$

is of type III.

Proof: From Lemma 1 it follows that block (14) is normalized. From the definition of φ it follows then that it is of type III. ■

EXAMPLE: Let us consider the second block in the example of section 5.2. This block a is of type II, with

$$a \;=\; [4,6] \times [6,10] \times [10,11] \times [0,13] \times [4,16] \;\times\; [1,3] \times [5,9] \times [6,9] \times [11,15] \times [11,15],$$

$$\varphi^{-1} = \{6 \mapsto 4,\ 7 \mapsto 1,\ 8 \mapsto 2,\ 9 \mapsto 3,\ 10 \mapsto 5\},$$

$$(\overline{\pi_6(r)}, \ldots, \overline{\pi_{10}(r)}) = (\overline{a_6 \cap a_{\varphi^{-1}(6)}}, \ldots, \overline{a_{10} \cap a_{\varphi^{-1}(10)}}) = (3, 6, 9, 11, 15).$$

Then the block

$$b = [0,13] \times [4,6] \times [6,10] \times [10,11] \times [4,16] \;\times\; [1,3] \times [5,6] \times [6,9] \times [11,11] \times [11,15]$$

is of type III. □

6.2. Sorting the last n indices in increasing order of $\leq_a$

Let $a = a_1 \times \cdots \times a_{2n}$ be a normalized $2n$-block. The binary relations $\triangleright_a$ and $\leq_a$ between the elements of $(n+1)..(2n)$ are defined by the following equivalences:

$$i \triangleright_a j \iff a_{i-n} \cap a_j \neq \emptyset$$

$$i \leq_a j \iff \begin{pmatrix} \text{either } i \leq j \text{ and there exist } k_1, \ldots, k_h \text{ with } i = k_1,\ h \geq 1,\ k_h = j, \\ \qquad\qquad k_i < k_{i+1},\ k_i \triangleright_a k_{i+1}, \text{ for each } i \in 1..(h-1), \\ \text{or } i > j \text{ and we do not have } j \leq_a i. \end{pmatrix}$$

We first establish two simple properties:

PROPERTY 7 *Let a be a $2n$-block of type III and i, j, k be elements of $(n+1)..(2n)$ such that $i < j < k$. If $j \triangleright_a i$ and $i \triangleright_a k$, then $j \triangleright_a k$.*

Proof: We notice first that, since a is of type III, we have $i \in rel\,(a, i-n)$ and $j \in rel\,(a, j-n)$. Therefore none of these relevant sets of indices is empty.

If $j \triangleright_a i$ then $a_{j-n} \cap a_i \neq \emptyset$. Since $i < j$ and a is of type III, we deduce that $\overline{a_{i-n}} \preceq \overline{a_{j-n}}$ and, according to Property 3, that $max\,(rel\,(a, i-n)) \leq max\,(rel\,(a, j-n))$. If additionally $i \triangleright_a k$, we obtain $k \in rel\,(a, i-n)$, hence

$$min\,(rel\,(a, j-n)) \leq j < k \leq max\,(rel\,(a, i-n)) \leq max\,(rel\,(a, j-n)$$

It follows that $k \in rel\,(a, j-n)$, thus $j \triangleright_a k$. ■

PROPERTY 8 *Let a be a normalized $2n$-block and i, j, k be elements of $(n+1)..(2n)$ such that $i < j < k$. If $i \leq_a k$, then $i \leq_a j$.*

Proof: Since $i < j < k$ and $i \leq_a k$, there exists an m-tuple $(k_1, \ldots, k_m)$ of integers, which is increasing for $<$ and such that $i = k_1$, $k = k_m$ and $k_{h+1} \in rel\,(a, k_h - n)$ for each $h \in 1..(m-1)$. So, there exists $h \in 1..(m-1)$ such that $j \in k_h..k_{h+1}$. Since $k_h \in rel\,(a, k_h - n)$, $k_{h+1} \in rel\,(a, k_h - n)$ and $rel\,(a, k_h - n)$ is an interval, we have $j \in rel\,(a, k_h - n)$, thus $i \leq_a j$. ■

Using this last property we can prove the following result:

PROPERTY 9 *If a is a normalized $2n$-block, the relation $\leq_a$ is a total order on the set $(n+1)..(2n)$.*

Proof: It is obvious that if i, j are elements of $(n+1)..(2n)$, then $i \leq_a j$ or $j \leq_a i$. It is also evident that the relation $\leq_a$ is antisymmetric and reflexive. It remains to show that it is transitive. Let i, j, k be elements of $(n+1)..(2n)$ such that $i \leq_a j$ and $j \leq_a k$. Six possible cases arise:
If $i \leq j$ and $k < i$, we cannot have $k \leq_a i$ since otherwise, by the first case in the definition of $\leq_a$, we would have $k \leq_a j$, which would contradict $j \leq_a k$. Thus $i \leq_a k$.
If $i \leq j$ and $k \in i..j$, by Property 8 we have $i \leq_a k$.
If $i \leq j$ and $j < k$, by the first case in the definition of $\leq_a$, we have $i \leq_a k$.
If $j < i$ and $k < j$, we cannot have $k \leq_a i$ since otherwise, by Property 8, we would have $k \leq_a j$, which would contradict $j \leq_a k$. Thus $i \leq_a k$.
If $j < i$ and $k \in j..i$, we cannot have $k \leq_a i$ since otherwise, by the first case in the definition of $\leq_a$, we would have $j \leq_a i$, which would contradict $i \leq_a j$. Thus $i \leq_a k$.
If $j < i$ and $i < k$, we have $j \leq_a k$ and, by Property 8, $j \leq_a i$, which contradicts $i \leq_a j$. This case cannot happen. ■

We can now formulate the main result of this subsection.

THEOREM 3 *Let a be a $2n$-block of type III. While it is possible, let us apply one of the transformations, described below, to the 3-tuple $(\varepsilon, (n+1, \ldots, 2n), \varepsilon)$. After $2n$ transformations we otain a 3-tuple of the form $(\varepsilon, \varepsilon, w)$, where w is the n-tuple*

$(n{+}1, \ldots, 2n)$ *sorted in increasing order for* $\leq_a$.
Transformations:

1. $(u,\, (k){\cdot}v,\, w) \longrightarrow (u{\cdot}(k),\, v,\, w)$, *if* $u = \varepsilon$ *or* $u \neq \varepsilon$ *and* $\grave{u} \triangleright_a k$
2. $(u{\cdot}(k),\, v,\, w) \longrightarrow (u,\, v,\, (k){\cdot}w,)$, *if* $v = \varepsilon$ *or* $v \neq \varepsilon$ *and* $k \not\triangleright_a \acute{v}$.

where k *is an integer,* u *and* v *are (possibly empty) sequences of integers,* $\grave{u}$ *denotes the last term of* u *and* $\acute{v}$ *the first term of* v*, provided that* u *and* v *are not empty.*

Proof: Let (u', v', w') be a current 3-tuple obtained by transformations of the initial 3-tuple. If both sequences u',v' are empty, no transformation is applicable to the 3-tuple and, if at least one of the sequences u', v' is not empty, exactly one transformation is applicable. Since transformation 1 moves one element from v' to u' and transformation 2 moves one element from u' to w', we conclude that, in $2n$ transformations, we reach deterministically a final configuration of the form $(\varepsilon, \varepsilon, w')$, where w' is the sequence $(n{+}1, n{+}2, \ldots, 2n)$ rearranged in a different order. To show that the sequence w' of the final 3-tuple is increasing with respect to $\leq_a$, we will establish a stronger property, namely that the current 3-tuple (u', v', w') obeys (1)–(4), i.e.,

(1) $u'{\cdot}v'$ is increasing for $<$,
(2) $i > j$, for each $i \in \mathit{set}\,(v')$ and $j \in \mathit{set}\,(w')$,
(3) $i \leq_a j$, for each $i \in \mathit{set}\,(v')$ and $j \in \mathit{set}\,(w')$,
(4) $u'{\cdot}w'$ is increasing for $\leq_a$.

The properties (1),(2),(3) and (4) hold for the initial 3-tuple. It is thus sufficient to show that they are preserved after each transformation. The only not obvious case is to show that property (3) is preserved when the 3-tuple (u', v', w') is of the form

$(u{\cdot}(k),\, v,\, w)$, with $v \neq \varepsilon$ and $k \not\triangleright_a \acute{v}$,

when transformation 2 is applied and the 3-tuple

$(u,\, v,\, (k){\cdot}w)$

is obtained. Suppose by way of contradiction that in this case property (3) is not preserved. Then there exists $l \in \mathit{set}\,(v)$ such that $k \leq_a l$. Since the sequence $u{\cdot}(k){\cdot}v$ is increasing for $<$, according to Property 8, we have $k \leq_a \acute{v}$. Since for each $i \in \mathit{set}\,(v)$ and each $j \in \mathit{set}\,(w)$ we have $i \leq_a j$, there exists no $k' \in (n+1)..(2n)$ such that $k < k' < \acute{v}$ and $k \leq_a k' \leq_a \acute{v}$. We conclude that $k \triangleright_a \acute{v}$, getting a contradiction. Thus property (3) is also preserved in this case. ■

Example: The $2n$-block on $(\mathbf{Z}, \leq)$

$$a \;=\; [0,13]\times[4,6]\times[6,10]\times[10,11]\times[4,16]\times[1,3]\times[5,6]\times[6,9]\times[11,11]\times[11,15],$$

with $n = 5$, is of type III. Sorting the 5-tuple $(6, 7, 8, 9, 10)$ in increasing order for $\leq_a$ is performed by the $2n$ transformations

$(\varepsilon, (6,7,8,9,10), \varepsilon)$
$\longrightarrow\ ((6), (7,8,9,10), \varepsilon)$
$\longrightarrow\ ((6,7), (8,9,10), \varepsilon)$
$\longrightarrow\ ((6,7,8), (9,10), \varepsilon)$
$\longrightarrow\ ((6,7), (9,10), (8))$
$\longrightarrow\ ((6), (9,10), (7,8))$
$\longrightarrow\ ((6,9), (10), (7,8))$
$\longrightarrow\ ((6,9,10), \varepsilon, (7,8))$
$\longrightarrow\ ((6,9), \varepsilon, (10,7,8))$
$\longrightarrow\ ((6), \varepsilon, (9,10,7,8))$
$\longrightarrow\ (\varepsilon, \varepsilon, (6,9,10,7,8))$

and produces the 5-tuple $(6,9,10,7,8)$. □

COMPLEXITY 4 Due to the nature of the transformations of Theorem 3 and the fact that $2n$ transformations are applied, the sorting of the n-tuple $(n{+}1, \ldots, 2n)$, in increasing order for $\leq_a$, can be performed in $\mathcal{O}(n)$ elementary instructions.

6.3. Autonomous subsets of indices

Let c be a set of integers and n a positive integer. We say that c is *n-balanced* if c is a subset of $1..2n$ and the sets $c \cap 1..n$ and $c \cap (n{+}1)..(2n)$ have the same number of elements. If a is a $2n$-block on $(\mathbf{D}, \preceq)$, we say that c is *a-autonomous* if c is an n-balanced subset of $1..2n$ and if, for each one to one mapping f from $1..n$ to $(n{+}1)..(2n)$, we have the implication

$$a_1 \times \cdots \times a_n \ \cap\ a_{f(1)} \times \cdots \times\ a_{f(n)} \neq \emptyset \ \implies\ f(c \cap 1..n) = c \cap (n{+}1)..(2n),$$

with of course $a_i = \pi_i(a)$.

Notice that if c and d are a-autonomous, then the subsets $c \cap d$, $c \cup d$ and $c - d$ are autonomous too.

PROPERTY 10 *Let $a = a_1 \times \cdots \times a_{2n}$ be a non-empty $2n$-block on $(\mathbf{D}, \preceq)$ and $P = \{c_1, \ldots, c_k\}$ be a partition of $1..2n$ into k n-balanced classes. A sufficient condition for the classes of P to be a-autonomous is that, for each $i \in 1..n$ and $j \in (n{+}1)..(2n)$, we have the implication*

$$\mathit{rank}\,(i, P) > \mathit{rank}\,(j, P) \ \implies\ a_i \cap a_j = \emptyset, \tag{15}$$

where $\mathrm{rank}\,(i, P)$ denotes the index h such that $i \in c_h$.

Proof: Let us suppose that at least one class in P is not a-autonomous and let us find the indices $i \in 1..n$ and $j \in (n{+}1)..(2n)$ which violate (15). Let h be the greatest index such that the class c_h in P is not a-autonomous. Then there exists a one to one mapping f from $1..n$ to $(n{+}1)..(2n)$ such that $a_1 \times \cdots \times a_n \cap a_{f(1)} \times \cdots \times a_{f(n)} \neq \emptyset$ and $f(c_h \cap 1..n) \neq c_h \cap (n{+}1)..(2n)$. Since c_h is n-balanced, there exists then an index $i \in c_h$ such that $f(i) \notin c_h$. The class of $f(i)$ in P is thus not a-autonomous and is

distinct from c_h. Since h is the greatest index such that c_h is a non a-autonomous class of P, we have $h > rank(f(i), P)$. By noticing that $rank(i, P) = h$ and by letting $j = f(i)$, we have $i \in 1..n$ and $j \in (n+1)..(2n)$ with $rank(i, P) > rank(j, P)$ and $a_i \cap a_j \neq \emptyset$. ■

We can now state the main result of this subsection.

LEMMA 3 *Let a be a $2n$-block of type III, let w be the $2n$-tuple $(n+1, \ldots, 2n)$ sorted in increasing order for $\leq_a$ and let $w_1, \ldots, w_k$ be the n_i-tuples, which are increasing for $<$, have maximal lengths and are such that $w = (w_1 \cdot - - - \cdot w_k)$. The set $P = \{c_1, \ldots, c_k\}$ with $c_i = set(w_i) \cup \{j-n \mid j \in set(w_i)\}$ is a partition of $1..2n$ into a-autonomous subsets.*

Proof: Pick i, j in $1..k$, with $j < i$. According to Property 10, it is sufficient to show that for no $i' \in set(w_i)$ and no $j' \in set(w_j)$ we have $i' \triangleright_a j'$. If $i' < j'$, since $j' \leq_a i'$ we do not have $i' \triangleright_a j'$. If $i' > j'$, let $\grave{w}_j$ be the last term of w_j and w'_{j+1} be the first term of w_{j+1}. Then we have $i' < \grave{w}_j$. Otherwise from $w'_{j+1} < \grave{w}_j < i'$ and $w'_{j+1} \leq_a i'$, by Property 8, we would infer $w'_{j+1} \leq_a \grave{w}_j$, which is false. Thus $i' \in (j'+1)..(\grave{w}_j - 1)$. Let j'' be the term which precedes i' and k be the term which succeeds i' in w_j. If $i' \triangleright_a j'$, by Property 3, we would have $i' \triangleright_a j''$ and, since $j'' \triangleright_a k$, by Property 7, we would obtain $i' \triangleright_a k$, contrary to $k \leq_a i'$ and $i' < k$. Hence we do not have $i' \triangleright_a j'$. ■

EXAMPLE: For the $2n$-block

$$a = [0,13] \times [4,6] \times [6,10] \times [10,11] \times [4,16] \times [1,3] \times [5,6] \times [6,9] \times [11,11] \times [11,15],$$

of type III of the preceding example, we have $w = (6, 9, 10, 7, 8)$. As a partition of $1..2n$ into autonomous subsets, we get

$$P = \{\{6, 9, 10, 1, 4, 5\}, \{7, 8, 2, 3\}\}.$$

□

7. Decomposition of a block of type III into blocks of type I

We can now establish a theorem for decomposing a block of type III into blocks of type I. The statement of the theorem will make use of a constraint formalism which we introduce first. The same formalism will be used, in the next section, for describing the algorithm which computes $apx(sort \cap a)$.

7.1. Basic constraint and composed constraint

Let **V** be an infinite set of *variables*. By a *basic constraint*, we mean a formula p of the form

$$(x_1, \ldots, x_m) \in r \wedge x_1 \in a_1 \wedge \cdots \wedge x_m \in a_m \tag{16}$$

where the x_i's are distinct elements of $\mathbf{V}$, the a_i's are intervals of $(\mathbf{D}, \preceq)$ and r is a global relation on $\mathbf{D}$, that is to say a subset of $\mathbf{D}^\star$. We denote by $box(p)$ the m-block $a_1 \times \cdots \times a_m$. By a *composed constraint*, we mean a formula of the form

$$p_1 \wedge \cdots \wedge p_k, \tag{17}$$

where the p_i's are basic constraints sharing no variable. We do not distinguish between two composed constraints which can be rendered equal by changing the associations or commuting the operands of the $\wedge$ connector. We denote by $var(q)$ the set of elements of $\mathbf{V}$ which occur in the composed constraint q. If $x \in var(q)$, we denote by $dom(x, q)$ the interval b of $\mathbf{D}$ such that the formula $x \in b$ occurs in q.

Of course, a *solution* σ of a basic constraint of the form (16) is a mapping from $\mathbf{V}$ to $\mathbf{D}$ such that $(\sigma(x_1), \ldots, \sigma(x_m))$ belongs to r and such that each $\sigma(x_i)$ belongs to the corresponding a_i. A *solution* of a composed constraint of the form (17) is a mapping σ from $\mathbf{V}$ to $\mathbf{D}$ which is a solution of each basic constraint p_i. We denote by $sol(q)$ the set of solutions of q. Two composed constraints q_1 and q_2 are *equivalent* iff $sol(q_1) = sol(q_2)$. In this case we write $q_1 \equiv q_2$.

We notice that the solutions of a basic constraint of the form (16) are the mappings σ from $\mathbf{V}$ to $\mathbf{D}$ such that $(\sigma(x_1), \ldots, \sigma(x_m))$ belongs to $r \cap box(p)$. If s is a set of mappings from $\mathbf{V}$ to $\mathbf{D}$, the *projection of s on the variable x* is the subset of $\mathbf{D}$

$$\pi_x(s) = \{e \in \mathbf{D} \mid \text{there exists } \sigma \in s \text{ with } e = \sigma(x)\}$$

Since two basic constraints of a composed constraint share no variable, we have:

PROPERTY 11 *Let q be a composed constraint, with $sol(p) \neq \emptyset$, for all its basic constraints p. If $x \in var(q)$, then $\pi_x(sol(q)) = \pi_x(sol(p_x))$, where p_x is the basic constraint of q in which x occurs.*

7.2. Decomposition of a block

THEOREM 4 *Let $a = a_1 \times \cdots \times a_{2n}$ be a $2n$-block of type III and let $w = w_1 \cdot - - - \cdot w_k$ be the n-tuple $(n+1, \ldots, 2n)$ sorted in increasing order for $\leq_a$, where the w_i's are sequences, increasing for $<$ and of maximal lengths n_i. For each $i \in 1..k$ we introduce the $2n_i$-tuple*

$$(h_{(i,1)}, \ldots, h_{(i,2n_i)}), \quad \textit{with} \quad (n+h_{(i,1)}, \ldots, n+h_{(i,n_i)}) = (h_{(i,n_i+1)}, \ldots, h_{(i,2n_i)}) = w_i.$$

Then each $2n_i$-block $b_i = a_{h_{(i,1)}} \times \cdots \times a_{h_{(i,2n_i)}}$ is of type I and we have the equivalence of constraints

$$\begin{bmatrix} (x_1, \ldots, x_{2n}) \\ \in sort \\ \wedge\ x_1 \in a_1 \\ \ldots \\ \wedge\ x_{2n} \in a_{2n} \end{bmatrix} \equiv \begin{bmatrix} (x_{h_{(1,1)}}, \ldots, x_{h_{(1,2n_1)}}) \\ \in sort \\ \wedge\ x_{h_{(1,1)}} \in a_{h_{(1,1)}} \\ \ldots \\ \wedge\ x_{h_{(1,2n_1)}} \in a_{h_{(1,2n_1)}} \end{bmatrix} \wedge \cdots \wedge \begin{bmatrix} (x_{h_{(k,1)}}, \ldots, x_{h_{(k,2n_k)}}) \\ \in sort \\ \wedge\ x_{h_{(k,1)}} \in a_{h_{(k,1)}} \\ \ldots \\ \wedge\ x_{h_{(k,2n_k)}} \in a_{h_{(k,2n_k)}} \end{bmatrix} \tag{18}$$

where the x_i's are distinct elements of $\mathbf{V}$.

Proof: Let i be an element in $1..k$. Since a is of type III, the block b_i is normalized and we have $a_{h_{(i,1)}} \times \cdots \times a_{h_{(i,n_i)}} \cap a_{h_{(i,n_i+1)}} \times \cdots \times a_{h_{(i,2n_i)}} \neq \emptyset$. Since w is the n-tuple $(n{+}1, \ldots, 2n)$ sorted in increasing order of $\leq_a$, for no $j \in (n_i{+}1)..(2n_i{-}1)$ there exists $l \in (n{+}1)..(2n)$ such that $h_{(i,j)} \leq_a l \leq_a h_{(i,j+1)}$ and $h_{(i,j)} < l < h_{(i,j+1)}$. It follows that, for each $j \in (n_i{+}1)..(2n_i{-}1)$, we have $h_{(i,j)} \triangleright_a h_{(i,j+1)}$, that is $a_{h_{(i,j)}-n} \cap a_{h_{(i,j+1)}} \neq \emptyset$. Thus we have also $a_{h_{(i,1)}} \times \cdots \times a_{h_{(i,n_i-1)}} \cap a_{h_{(i,n_i+2)}} \times \cdots \times a_{h_{(i,2n_i)}} \neq \emptyset$ and the block b_i is indeed of type I.

To prove equivalence (18), we first notice that, by Lemma 3, the set $P = \{c_1, \ldots, c_k\}$, with $c_i = \mathit{set}(w_i) \cup \{j - n \mid j \in \mathit{set}(w_i)\}$, is a partition of $1..2n$ into a-autonomous subsets. If j is any element of $(n{+}1)..(2n)$, denote by $\mathit{class}(j, P)$ the set c_i such that $j \in c_i$.

Let σ be a mapping from $\mathbf{V}$ to $\mathbf{D}$. Let $(d_1, \ldots, d_{2n}) = (\sigma(x_1), \ldots, \sigma(x_{2n}))$. It is clear that, if σ is a solution of the first constraint of (18), then σ is a solution of the second one. To show the converse, suppose by way of contradiction that σ is a solution of the second constraint but not of the first one. Then, for each $i \in 1..(2n)$, we have $d_i \in a_i$, there exists a one to one mapping f from $1..n$ to $(n{+}1)..(2n)$ such that $(d_1, \ldots, d_m) = (d_{f(1)}, \ldots, d_{f(m)})$ and, for each pair (j, j') of elements in $(n{+}1)..(2n)$ such that $\mathit{class}(j, P) = \mathit{class}(j', P)$ and $j < j'$, we have $d_j \preceq d_{j'}$. Since σ is not a solution of the first constraint, there exists a pair (j, j') of elements in $(n{+}1)..(2n)$ with $\mathit{class}(j, P) \neq \mathit{class}(j', P)$ and $j < j'$, $d_{j'} \preceq d_j$. Since $d_j \in a_j$, $d_{j'} \in a_{j'}$ and since a is normalized, we have $d_j \in a_{j'}$ and $d_{j'} \in a_j$, thus the $2n$-tuple obtained by exchanging d_j and $d_{j'}$ in $(d_1, \ldots, d_{2n})$ still belongs to a. Let (i, i') be the pair $(f^{-1}(j), f^{-1}(j'))$. The mapping g which differs from f only by the fact that $(g(i), g(i')) = (f(i'), f(i))$ is such that $a_1 \times \cdots \times a_n \cap a_{g(1)} \times \cdots \times a_{g(n)}) \neq \emptyset$ and $\mathit{class}(i, P) \neq \mathit{class}(g(i), P)$. This contradicts the fact that the classes of P are a-autonomous. ■

EXAMPLE: Let us take the $2n$-block of type III in the example of section 6.2:

$a = a_1 \times \cdots \times a_{2n} =$
$[0, 13] \times [4, 6] \times [6, 10] \times [10, 11] \times [4, 16] \times [1, 3] \times [5, 6] \times [6, 9] \times [11, 11] \times [11, 15].$

with $n = 5$ and $w = (6, 9, 10, 7, 8)$. We have $k = 2$ and $w = w_1 \cdot w_2$ with $w_1 = (6, 9, 10)$ and $w_2 = (7, 8)$. We obtain

$(h_{(1,1)}, \ldots, h_{(1,6)}) = (1, 4, 5, 6, 9, 10),$
$(h_{(2,1)}, \ldots, h_{(2,4)}) = (2, 3, 7, 8).$

Thus we have the equivalence

$$\begin{bmatrix}(x_1,\ldots,x_{10})\\ \in sort\\ \wedge\, x_1 \in a_1\\ \ldots\\ \wedge\, x_{10} \in a_{10}\end{bmatrix} \equiv \begin{bmatrix}(x_1,x_4,x_5,\, x_6,x_9,x_{10})\\ \in sort\\ \wedge\, x_1 \in a_1\\ \wedge\, x_4 \in a_4\\ \wedge\, x_5 \in a_5\\ \wedge\, x_6 \in a_6\\ \wedge\, x_9 \in a_9\\ \wedge\, x_{10} \in a_{10}\end{bmatrix} \wedge \begin{bmatrix}(x_2,x_3,\, x_7,x_8)\\ \in sort\\ \wedge\, x_2 \in a_2\\ \wedge\, x_3 \in a_3\\ \wedge\, x_7 \in a_7\\ \wedge\, x_8 \in a_8\end{bmatrix}.$$

Moreover, the blocks $a_1 \times a_4 \times a_5 \times a_6 \times a_9 \times a_{10}$ and $a_2 \times a_3 \times a_7 \times a_8$ are of type I. □

8. Algorithm

Given a $2m$-block b, we have now at our disposal a collection of results for deciding whether the set $sort \cap b$ is not empty and, if it is the case, for computing the least elements of its first m projections and the greatest elements of its last m projections. In order to compute the other endpoints of the projections of $sort \cap b$ we will introduce a duality principle. Then we will detail an algorithm for computing a $2m$-block equal to $apx\,(sort \cap b)$, in $\mathcal{O}(m \log m)$ elementary instructions. Notice that the existence of such an algorithm proves that $apx\,(sort \cap b)$ is always well defined.

8.1. Duality principle

Let us extend our notations. Let ρ be any total order relation on $\mathbf{D}$, let a be a subset of $\mathbf{D}$, let d be an element of $\mathbf{D}^n$ and e, e' be elements of $\mathbf{D}$. We define the global relation $sort^\rho$, the elements $\underline{a}_\rho$, $\overline{a}^\rho$, $\downarrow_\rho d$, $\uparrow^\rho d$ of $\mathbf{D}$ and the subset $[e,e']^\rho$ of $\mathbf{D}$ in the same way as $sort$, $\underline{a}$, $\overline{a}$, $\downarrow d$, $\uparrow d$, $[e,e']$, but by substituting ρ for $\preceq$. Similarly we introduce the notion of a ρ-normalized block and the notions of a block of type ρ-I, of type ρ-II and of type ρ-III.

We denote by ρ^T the relation defined by $e \rho^T e'$ if and only if $e' \rho\, e$. The ordered set $(\mathbf{D}, \rho^T)$ will be called *dual* of $(\mathbf{D}, \rho)$. We see that $(\rho^T)^T = \rho$ and that:

PROPERTY 12 *Let a be a subset of $\mathbf{D}$ and $(d_1,\ldots,d_{2n})$ an element of $\mathbf{D}^{2n}$. The following properties hold:*
(1) a is an interval of $(\mathbf{D},\rho)$ if and only if a is an interval of $(\mathbf{D},\rho^T)$,
(2) the elements $\underline{a}_\rho$ and $\overline{a}^{\rho^T}$ either exist and are equal, or they do not exist,
(3) the elements $\overline{a}^\rho$ and $\underline{a}_{\rho^T}$ either exist and are equal, or they do not exist,
(4) $(d_1,\ldots,d_{2n}) \in sort^\rho$ if and only if $(d_1,\ldots,d_n,d_{2n},\ldots,d_{n+1}) \in sort^{\rho^T}$.

8.2. Description of the algorithm

Algorithm *Given a $2m$-block b on $(\mathbf{D},\preceq)$ of the form $b_1 \times \cdots \times b_{2m}$, we are interested in computing $apx\,(sort \cap b)$. The algorithm consists in computing a sequence*

$q_1, \ldots, q_{12}$ of twelve composed constraints. Each of these constraints involves the same subset $\{y_1, \ldots, y_{2m}\}$ of $2m$ variables, the relation sort or the relation $sort^{\preceq^T}$, and is defined by:

$$q_1 \;=\; (y_1, \ldots, y_{2m}) \in sort \wedge y_1 \in b_1 \wedge \cdots \wedge y_{2m} \in b_{2m},$$

$$q_{i+1} \;=\; \begin{cases} q_i, & \text{if there exists } j \in 1..m \text{ such that } dom\,(y_j, q_i) = \emptyset, \\ \mathcal{T}_i(p_{i1}) \wedge \cdots \wedge \mathcal{T}_i(p_{ik_i}), \text{ with } q_i = p_{i1} \wedge \cdots \wedge p_{ik_i}, & \text{otherwise,} \end{cases}$$

where i is taken in $1..11$, each p_{ij} is a basic constraint and $\mathcal{T}_i(p_{ij})$ is the composed constraint defined below. We then have

$$apx\,(sort \cap b) = dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}).$$

Definition 6 (of $\mathcal{T}_i$). For any $i \in 1..11$ and for any basic constraint p of the form

$$\begin{bmatrix} (x_1, \ldots, x_{2n}) \in sort^\rho \\ \wedge\; x_1 \in a_1 \\ \ldots \\ \wedge\; x_{2n} \in a_{2n} \end{bmatrix}$$

where ρ is $\preceq$ or $\preceq^T$, the composed constraint $\mathcal{T}_i(p)$ is defined as follows in section *(i)*.

(1) Computation of a ρ-normalized block

$$\mathcal{T}_1(p) = \begin{bmatrix} (x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n}) \in sort^\rho \\ \wedge\; x_1 \in a_1 \\ \ldots \\ \wedge\; x_n \in a_n \\ \wedge\; x_{n+1} \in [e_1, e'_1]^\rho \\ \ldots \\ \wedge\; x_{2n} \in [e_n, e'_n]^\rho \end{bmatrix}$$

with $(e_1, \ldots, e_n) = \uparrow^\rho (\underline{a_{n+1}}^\rho, \ldots, \underline{a_{2n}}^\rho)$ and $(e'_1, \ldots, e'_n) = \downarrow_\rho (\overline{a_{n+1}}^\rho, \ldots, \overline{a_{2n}}^\rho)$.

(2) Computation of a block of type ρ-II

$$\mathcal{T}_2(p) = \begin{bmatrix} (x_{\nu(1)}, \ldots, x_{\nu(n)}, x_{n+1}, \ldots, x_{2n}) \in sort^\rho \\ \wedge\; x_1 \in a_1 \\ \ldots \\ \wedge\; x_{2n} \in a_{2n} \end{bmatrix}$$

where ν is a one to one mapping from $1..n$ to $(n+1)..(2n)$, such that $(\overline{a_{\nu(1)}}^\rho, \ldots, \overline{a_{\nu(n)}}^\rho)$ is increasing for ρ.

(3) Top pruning of the ending of the block and computation of a block of type ρ-III

$$\mathcal{T}_3(p) = \begin{cases} \begin{bmatrix} (x_1,\ldots,x_{2n}) \in sort^\rho \\ \wedge\ x_1 \in \emptyset \\ \ldots \\ \wedge\ x_{2n} \in \emptyset \end{bmatrix}, & \text{if } \varphi \text{ does not exist} \\ \\ \begin{bmatrix} (x_{\varphi^{-1}(n+1)},\ldots,x_{\varphi^{-1}(2n)},x_{n+1},\ldots,x_{2n}) \in sort^\rho \\ \wedge\ x_1 \in a_1 \\ \ldots \\ \wedge\ x_n \in a_n \\ \wedge\ x_{n+1} \in [\underline{a_{n+1}}^{\,\rho}, \overline{a_{n+1} \cap a_{\varphi^{-1}(n+1)}}^{\,\rho}]^{\,\rho} \\ \ldots \\ \wedge\ x_{2n} \in [\underline{a_{2n}}^{\,\rho}, \overline{a_{2n} \cap a_{\varphi^{-1}(2n)}}^{\,\rho}]^{\,\rho} \end{bmatrix}, & \text{if } \varphi \text{ exists} \end{cases}$$

where φ is the one to one mapping from $1..n$ to $(n+1)..(2n)$ defined by

$$\begin{array}{rcl} \varphi(1) & = & min\,(rel\,(a,1)), \\ \varphi(i) & = & min\,(rel\,(a,i) - \varphi(1..(i-1))), \quad \text{if } i \in 2..n. \end{array}$$

Recall that

$$rel\,(a,i) = min\,\{j \in (n+1)..(2n) \mid a_i \cap a_j \neq \emptyset\}.$$

(4) Decomposition into blocks of type ρ-I

$$\mathcal{T}_4(p) = \begin{bmatrix} (x_{h_{(1,1)}},\ldots,x_{h_{(1,2n_1)}}) \in sort^\rho \\ \wedge\ x_{h_{(1,1)}} \in a_{h_{(1,1)}} \\ \ldots \\ \wedge\ x_{h_{(1,2n_1)}} \in a_{h_{(1,2n_1)}} \end{bmatrix} \wedge \cdots \wedge \begin{bmatrix} (x_{h_{(k,1)}},\ldots,x_{h_{(k,2n_k)}}) \in sort^\rho \\ \wedge\ x_{h_{(k,1)}} \in a_{h_{(k,1)}} \\ \ldots \\ \wedge\ x_{h_{(k,2n_k)}} \in a_{h_{(k,2n_k)}} \end{bmatrix}$$

where k, the n_i's and the $h_{(i,j)}$'s, with $i \in 1..k$ and $j \in 1..(2n_i)$, are computed as follows. Using the algorithm in Theorem 3, we sort the n-tuple $(n+1,\ldots,2n)$ in increasing order for $\leq_a$ with $a = a_1 \times \cdots \times a_{2n}$. We then put the obtained n-tuple into the form $w_1 \cdot - - - \cdot w_k$, where the n_i-tuples w_i are increasing for $<$ and of maximal lengths n_i. For each $i \in 1..k$, the $2n_i$-tuple $(h_{(i,1)},\ldots,h_{(i,2n_i)})$ is then defined by

$$(n+h_{(i,1)},\ldots,n+h_{(i,n_i)}) = (h_{(i,n_i+1)},\ldots,h_{(i,2n_i)}) = w_i.$$

(5) Bottom pruning of the beginning of the block

$$\mathcal{T}_5(p) = \begin{bmatrix} (x_1,\ldots,x_{2n}) \in sort^{\rho} \\ \wedge\ x_1 \in [\underline{a_1 \cap a_{f(1)}}_{\rho}, \overline{a_1}^{\rho}]^{\rho} \\ \ldots \\ \wedge\ x_n \in [\underline{a_n \cap a_{f(n)}}_{\rho}, \overline{a_n}^{\rho}]^{\rho} \\ \wedge\ x_{n+1} \in a_{n+1} \\ \ldots \\ \wedge\ x_{2n} \in a_{2n} \end{bmatrix}$$

with $f(i) = min\,\{j \in (n+1)..(2n) \mid \underline{a_i}_{\rho}\, \rho\, \overline{a_j}^{\rho}\}$, for $i \in 1..n$.

(6) Computation of a ρ^T-normalized block

$$\mathcal{T}_6(p) = \begin{bmatrix} (x_1, x_2, \ldots, x_n, x_{2n}, \ldots, x_{n+2}, x_{n+1}) \in sort^{\rho^T} \\ \wedge\ x_1 \in a_1 \\ \ldots \\ \wedge\ x_{2n} \in a_{2n} \end{bmatrix}$$

(7) to (11) The sequence $\mathcal{T}_7(p), \mathcal{T}_8(p), \mathcal{T}_9(p), \mathcal{T}_{10}(p), \mathcal{T}_{11}(p)$ equals the sequence $\mathcal{T}_2(p), \mathcal{T}_3(p), \mathcal{T}_4(p), \mathcal{T}_5(p), \mathcal{T}_6(p)$.

8.3. Complexity of the algorithm

In the preceding sections, we have seen how to perform each computation of $\mathcal{T}_i(p)$ in $\mathcal{O}(n)$ or $\mathcal{O}(n \log n)$ elementary instructions. If $m, m_1, \ldots, m_k$ are positive integers such that $\sum_{j=1}^{k} m_j = m$, we have $\sum_{j=1}^{k} m_j \log m_j \leq m \log m$. Given the way the constraints $q_1, \ldots, q_{12}$ are defined, it follows that the computation of $apx\,(sort \cap b)$, with b of dimension $2m$, can be performed in $\mathcal{O}(m \log m)$ elementary instructions.

9. Correctness of the algorithm

The correctness of the algorithm follows from properties and theorems proven in the preceding sections. In order to prove this correctness in detail, we introduce table 1 which, for each i taken in 1..11, states and justifies several properties of q_i and q_{i+1}, which hold only when $dom\,(y_1, q_i) \times \cdots \times dom\,(y_{2m}, q_i) \neq \emptyset$. In this table, $varbegin\,(p)$ and $varend\,(p)$ denote the subsets of variables $\{x_1, \ldots, x_n\}$ and $\{x_{n+1}, \ldots, x_{2n}\}$, with $\{x_1, \ldots, x_{2n}\} = var(p)$.

From the second column of table 1 and from the way q_{i+1} is constructed, we deduce that $q_1 \equiv q_{12}$. If $dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}) \neq \emptyset$ we have thus $dom\,(y_1, q_{11}) \times \cdots \times dom\,(y_{2m}, q_{11}) \neq \emptyset$ and then the properties of q_{12} in the last line of the table hold. For each basic constraint p of q_{12}, we have

$$sol\,(p) \neq \emptyset,\ \underline{\pi_x(sol\,(p))} = \underline{dom\,(x,p)},\ \ \overline{\pi_x(sol\,(p))} = \overline{dom\,(x,p)},\ \text{for each } x \in var\,(p),$$

Table 1. Properties of q_i and q_{i+1}, when $dom\,(y_1, q_i) \times \cdots \times dom\,(y_{2m}, q_i) \neq \emptyset$.

i	*Property of each basic constraint p of q_i*	*Form of box(p) for a basic constraint p of q_{i+1}*	*Properties of each basic constraint p of q_{i+1}*	*Justification*
1	$\mathcal{T}_i(p) \equiv p$	normalized		Property 2
2	$\mathcal{T}_i(p) \equiv p$	of type II		sorting property
3	$\mathcal{T}_i(p) \equiv p$	of type III	$sol\,(p) \neq \emptyset$ and, for each $x \in varend\,(p)$, $\overline{\pi_x(sol\,(p))} = \overline{dom\,(x,p)}$	Theorem 2 and Property 6
4	$\mathcal{T}_i(p) \equiv p$	of type I	same prop. as in $i-1$	Theorems 3 and 4
5	$\mathcal{T}_i(p) \equiv p$	normalized	in addition to prop. in $i-1$, for each $x \in varbegin\,(p)$, $\underline{\pi_x(sol\,(p))} = \underline{dom\,(x,p)}$	Theorem 1
6	$\mathcal{T}_i(p) \equiv p$	$\preceq^T$-normalized	same prop. as in $i-1$	Property 12
7	$\mathcal{T}_i(p) \equiv p$	of type $\preceq^T$-II	same prop. as for $i-1$	sorting property
8	$\mathcal{T}_i(p) \equiv p$	of type $\preceq^T$-III	in addition to prop. in $i-1$, for each $x \in varend\,(p)$, $\underline{\pi_x(sol\,(p))} = \underline{dom\,(x,p)}$	Theorem 2 and Property 6
9	$\mathcal{T}_i(p) \equiv p$	of type $\preceq^T$-I	same prop. as in $i-1$	Theorems 3 and 4
10	$\mathcal{T}_i(p) \equiv p$	$\preceq^T$-normalized	in addition to prop. in $i-1$, for each $x \in varbegin\,(p)$, $\overline{\pi_x(sol\,(p))} = \overline{dom\,(x,p)}$	Theorem 1
11	$\mathcal{T}_i(p) \equiv p$	normalized	same prop. as in $i-1$	Property 12

that is,

$sol\,(p) \neq \emptyset$, $apx\,(\pi_x(sol\,(p)))$ exists and is equal to $dom\,(x,p)$ for each $x \in var\,(p)$

and thus, by Property 11,

$apx\,(\pi_{y_j}(sol\,(q_{12})))$ exists and is equal to $dom\,(y_j, q_{12})$, for each $j \in 1..2m$.

If $dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}) = \emptyset$, for each $j \in 1..2m$, the set $apx\,(\pi_{y_j}(sol\,(q_{12})))$ exists also and is equal to $\emptyset$. It follows that in all cases, the sets $apx\,(\pi_{y_j}(sol\,(q_{12})))$ exist and

$$apx\,(\pi_{y_1}(sol\,(q_{12}))) \times \cdots \times apx\,(\pi_{y_{2m}}(sol\,(q_{12}))) = dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}).$$

Since $sol\,(q_1) = sol\,(q_{12})$, the sets $apx\,(\pi_{y_j}(sol\,(q_1)))$ exist and

$$apx\,(\pi_{y_1}(sol\,(q_1))) \times \cdots \times apx\,(\pi_{y_{2m}}(sol\,(q_1))) = dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}).$$

By construction we have $\pi_{y_j}(sol\,(q_1)) = \pi_j(sort \cap b)$. Thus the sets $\pi_j(sort \cap b)$ exist and

$$apx\,(\pi_1(sort \cap b)) \times \cdots \times apx\,(\pi_{2m}(sort \cap b)) = dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}).$$

According to Property 1, we have indeed

$$apx\,(sort \cap b) = dom\,(y_1, q_{12}) \times \cdots \times dom\,(y_{2m}, q_{12}).$$

10. Example

For a better understanding of the algorithm of section 8, we illustrate it by an example. As an ordered set $(\mathbf{D}, \preceq)$ we take $(\mathbf{R}, \leq)$, and as a $2m$-block we consider

$$b \;=\; [0,13] \times [6,10] \times [10,11] \times [4,16] \times [4,6] \times [1,3] \times [5,10] \times [6,9] \times [11,17] \times [10,15].$$

with $m = 5$.

Initial composed constraint We thus have

$$q_1 \;=\; (y_1, \ldots, y_{10}) \in sort \;\wedge\; y_1 \in b_1 \;\wedge\; \cdots y_{10} \in b_{10},$$

with

$(b_1, \ldots, b_{10}) =$
$([0,13], [6,10], [10,11], [4,16], [4,6],\ [1,3], [5,10], [6,9], [11,17], [10,15])$.

(1) Computation of a normalized block We compute

$$\downarrow(1,5,6,11,10) \;=\; (1,5,6,11,11),$$
$$\uparrow(3,10,9,17,15) \;=\; (3,9,9,15,15).$$

and obtain

$$q_2 \;=\; (y_1, \ldots, y_{10}) \in sort \;\wedge\; y_1 \in b_1 \;\wedge\; \cdots y_{10} \in b_{10},$$

$(b_1, \ldots, b_{10}) =$
$([0,13], [6,10], [10,11], [4,16], [4,6],\ [1,3], [5,9], [6,9], [11,15], [11,15])$.

(2) Computation of a block of type II Next we get

$$q_3 \;=\; (y_5, y_2, y_3, y_1, y_4,\ y_6, y_7, y_8, y_9, y_{10}) \in sort \;\wedge\; y_1 \in b_1 \;\wedge\; \cdots y_{10} \in b_{10},$$

$(b_5, b_2, b_3, b_1, b_4,\ b_6, b_7, b_8, b_9, b_{10}) =$
$([4,6], [6,10], [10,11], [0,13], [4,16],\ [1,3], [5,9], [6,9], [11,15], [11,15])$.

(3) Top pruning of the ending of the block and computation of a block of type III The computation of φ gives

$\varphi \ = \ \{1 \mapsto 7,\ 2 \mapsto 8,\ 3 \mapsto 9,\ 4 \mapsto 6,\ 5 \mapsto 10\}.$

and thus the set $sort \cap b$ is not empty and

$q_4 \ = \ (y_1, y_5, y_2, y_3, y_4,\ y_6, y_7, y_8, y_9, y_{10}) \in sort \ \wedge \ y_1 \in b_1 \ \wedge \cdots y_{10} \in b_{10},$

$(b_1, b_5, b_2, b_3, b_4,\ b_6, b_7, b_8, b_9, b_{10}) =$
$([0, 13], [4, 6], [6, 10], [10, 11], [4, 16],\ [1, 3], [5, 6], [6, 9], [11, 11], [11, 15]).$

(4) Decomposition into blocks of type I Given $b = b_1 \times b_5 \times b_2 \times b_3 \times b_4 \times b_6 \times \cdots \times b_{10}$, the sorting of $(6, \ldots, 10)$ in increasing order for $\leq_b$, is performed by the sequence of transformations

$(\varepsilon, (6, 7, 8, 9, 10), \varepsilon)$
$\longrightarrow \ ((6), (7, 8, 9, 10), \varepsilon)$
$\longrightarrow \ ((6, 7), (8, 9, 10), \varepsilon)$
$\longrightarrow \ ((6, 7, 8), (9, 10), \varepsilon)$
$\longrightarrow \ ((6, 7), (9, 10), (8))$
$\longrightarrow \ ((6), (9, 10), (7, 8))$
$\longrightarrow \ ((6, 9), (10), (7, 8))$
$\longrightarrow \ ((6, 9, 10), \varepsilon, (7, 8))$
$\longrightarrow \ ((6, 9), \varepsilon, (10, 7, 8))$
$\longrightarrow \ ((6), \varepsilon, (9, 10, 7, 8))$
$\longrightarrow \ (\varepsilon, \varepsilon, (6, 9, 10, 7, 8)).$

We obtain $(6, 9, 10) \cdot (7, 8)$ and we then have

$q_5 =$
$(y_1, y_3, y_4,\ y_6, y_9, y_{10}) \in sort \ \wedge \ (y_5, y_2,\ y_7, y_8) \in sort \ \wedge \ y_1 \in b_1 \ \wedge \cdots \wedge \ y_{10} \in b_{10},$

$(b_1, b_3, b_4,\ b_6, b_9, b_{10}) \ = \ ([0, 13], [10, 11], [4, 16],\ [1, 3], [11, 11], [11, 15]),$
$(b_5, b_2,\ b_7, b_8) \ = \ ([4, 6], [6, 10],\ [5, 6], [6, 9]).$

(5) Bottom pruning of the beginning of the blocks We get

$q_6 =$
$(y_1, y_3, y_4,\ y_6, y_9, y_{10}) \in sort \ \wedge \ (y_5, y_2,\ y_7, y_8) \in sort \ \wedge \ y_1 \in b_1 \ \wedge \cdots \wedge \ y_{10} \in b_{10},$

$(b_1, b_3, b_4,\ b_6, b_9, b_{10}) \ = \ ([1, 13], [11, 11], [11, 16],\ [1, 3], [11, 11], [11, 15]),$
$(b_5, b_2,\ b_7, b_8) \ = \ ([5, 6], [6, 10],\ [5, 6], [6, 9]).$

(6) Computation of $\preceq^T$-normalized blocks We get

$q_7 =$
$(y_1, y_3, y_4,\ y_{10}, y_9, y_6) \in sort^{\preceq^T} \ \wedge\ (y_5, y_2, y_8, y_7) \in sort^{\preceq^T} \ \wedge$
$y_1 \in b_1 \ \wedge \cdots \wedge\ y_{10} \in b_{10},$

$$\begin{array}{lcl}(b_1, b_3, b_4,\ b_{10}, b_9, b_6) & = & ([1,13],[11,11],[11,16],\ [11,15],[11,11],[1,3]),\\ (b_5, b_2,\ b_8, b_7) & = & ([5,6],[6,10],\ [6,9],[5,6]).\end{array}$$

(7) Computation of blocks of type $\preceq^T$-II We get

$q_8 =$
$(y_3, y_4, y_1,\ y_{10}, y_9, y_6) \in sort^{\preceq^T} \ \wedge\ (y_2, y_5,\ y_8, y_7) \in sort^{\preceq^T} \ \wedge$
$y_1 \in b_1 \ \wedge \cdots \wedge\ y_{10} \in b_{10},$

$$\begin{array}{lcl}(b_3, b_4, b_1,\ b_{10}, b_9, b_6) & = & ([11,11],[11,16],[1,13],\ [11,15],[11,11],[1,3]),\\ (b_2, b_5,\ b_8, b_7) & = & ([6,10],[5,6],\ [6,9],[5,6]).\end{array}$$

(8) Bottom pruning of the endings and computation of blocks of type $\preceq^T$-III If φ' denotes the φ of $b_3 \times b_4 \times b_1 \times b_{10} \times b_9 \times b_6$ and φ'' the φ of $b_2 \times b_5 \times b_8 \times b_7$, we get

$$\begin{array}{lcl}\varphi & = & \{1 \mapsto 4,\ 2 \mapsto 5,\ 3 \mapsto 6\},\\ \varphi' & = & \{1 \mapsto 3,\ 2 \mapsto 4\}.\end{array}$$

We thus obtain

$q_9 =$
$(y_3, y_4, y_1,\ y_{10}, y_9, y_6) \in sort^{\preceq^T} \ \wedge\ (y_2, y_5,\ y_8, y_7) \in sort^{\preceq^T} \ \wedge$
$y_1 \in b_1 \ \wedge \cdots \wedge\ y_{10} \in b_{10},$

$$\begin{array}{lcl}(b_3, b_4, b_1, b_{10}, b_9, b_6) & = & ([11,11],[11,16],[1,13],\ [11,15],[11,11],[1,3]),\\ (b_2, b_5, b_8, b_7) & = & ([6,10],[5,6],\ [6,9],[5,6]).\end{array}$$

(9) Decomposition into blocks of type $\preceq^T$-I If we set $b' = b_3 \times b_4 \times b_1 \times b_{10} \times b_9 \times b_6$ and $b'' = b_2 \times b_5 \times b_8 \times b_7$, the sorting of $(4, 5, 6)$ in increasing order for $\leq_{b'}$ and the sorting of $(3, 4)$ in increasing order for $\leq_{b''}$ are performed by the two sequences of transformations

$$
\begin{array}{ll}
(\varepsilon,(4,5,6),\varepsilon) & \\
\longrightarrow\ ((4),(5,6),\varepsilon) & \qquad\qquad (\varepsilon,(3,4),\varepsilon) \\
\longrightarrow\ ((4,5),(6),\varepsilon) & \qquad\qquad \longrightarrow\ ((3),(4),\varepsilon) \\
\longrightarrow\ ((4),(6),(5)) \qquad \text{and} & \qquad\qquad \longrightarrow\ ((3,4),\varepsilon,\varepsilon) \\
\longrightarrow\ (\varepsilon,(6),(4,5)) & \qquad\qquad \longrightarrow\ ((3),\varepsilon,(4)) \\
\longrightarrow\ ((6),\varepsilon,(4,5)) & \qquad\qquad \longrightarrow\ (\varepsilon,\varepsilon,(3,4)). \\
\longrightarrow\ (\varepsilon,\varepsilon,(6,4,5)) &
\end{array}
$$

We obtain $(6)\cdot(4,5)$ and $(3,4)$. We then have

$q_{10} =$
$(y_1,\, y_6) \in sort^{\preceq^T} \wedge (y_3, y_4,\, y_{10}, y_9) \in sort^{\preceq^T} \wedge (y_2, y_5,\, y_8, y_7) \in sort^{\preceq^T} \wedge$
$y_1 \in b_1 \wedge \cdots \wedge y_{10} \in b_{10}$,

$$
\begin{array}{lcl}
(b_1,\, b_6) & = & ([1,13],\, [1,3]), \\
(b_3, b_4,\, b_{10}, b_9) & = & ([11,11], [11,16],\, [11,15], [11,11]), \\
(b_2, b_5,\, b_8, b_7) & = & ([6,10], [5,6],\, [6,9], [5,6]).
\end{array}
$$

(10) Top pruning of the beginning of the blocks We get

$q_{11} =$
$(y_1, y_6) \in sort^{\preceq^T} \wedge (y_3, y_4, y_{10}, y_9) \in sort^{\preceq^T} \wedge (y_2, y_5, y_8, y_7) \in sort^{\preceq^T} \wedge$
$y_1 \in b_1 \wedge \cdots \wedge y_{10} \in b_{10}$,

$$
\begin{array}{lcl}
(b_1,\, b_6) & = & ([1,3],\, [1,3]), \\
(b_3, b_4,\, b_{10}, b_9) & = & ([11,11], [11,15],\, [11,15], [11,11]), \\
(b_2, b_5,\, b_8, b_7) & = & ([6,9], [5,6],\, [6,9], [5,6]).
\end{array}
$$

(11) Computation of normalized blocks We get

$q_{12} =$
$(y_1, y_6) \in sort \wedge (y_3, y_4, y_9, y_{10}) \in sort \wedge (y_2, y_5, y_7, y_8) \in sort \wedge$
$y_1 \in b_1 \wedge \cdots \wedge y_{10} \in b_{10}$,

$$
\begin{array}{lcl}
(b_1,\, b_6) & = & ([1,3],\, [1,3]), \\
(b_3, b_4,\, b_9, b_{10}) & = & ([11,11], [11,15],\, [11,11], [11,15]), \\
(b_2, b_5,\, b_7, b_8) & = & ([6,9], [5,6],\, [5,6], [6,9]).
\end{array}
$$

Final result If b again denotes the initial block, we get

$apx\,(sort \cap b) =$
$[1,3]\times[6,9]\times[11,11]\times[11,15]\times[5,6] \times [1,3]\times[5,6]\times[6,9]\times[11,11]\times[11,15]$.

11. Conclusions

11.1. Incremental nature of the algorithm

Our algorithm for computing $apx(sort \cap a)$ has a notable virtue: it is incremental. Namely, given a pair (a, a') of $2n$-blocks on $(\mathbf{D}, \preceq)$ such that $a' \subseteq a$, the computation of $apx(sort \cap a')$ can be performed by taking advantage of the computation of $apx(sort \cap a)$. For this, we proceed as follows.

Let $a = a_1 \times \cdots \times a_{2n}$ and $a' = a'_1 \times \cdots \times a'_{2n}$ and assume that the computation of $apx(sort \cap a)$ has been performed by transforming the basic constraint

$$(x_1, \ldots, x_{2n}) \in sort \ \wedge \ x_1 \in a_1 \ \wedge \ \cdots \ \wedge \ x_{2n} \in a_{2n}$$

into a composed constraint q such that

$$apx(sort \cap a) = dom(x_1, q) \times \cdots \times dom(x_{2n}, q).$$

This constraint q is of the form $p_1 \wedge \cdots \wedge p_k$ with each p_i of the form

$$(x_{h_{(i,1)}}, \ldots, x_{h_{(i,2n_i)}}) \in sort \ \wedge \ x_{h_{(i,1)}} \in b_{h_{(i,1)}} \ \wedge \ \cdots \ \wedge \ x_{h_{(i,2n_i)}} \in b_{h_{(i,2n_i)}}$$

For each constraint p_i, we introduce the constraint p'_i obtained by replacing each $b_{h_{(i,j)}}$ by $b_{h_{(i,j)}} \cap a'_{h_{(i,j)}}$. On each constraint p'_i, which differs from p_i, we apply the 11 transformations of the algorithm and obtain a composed constraint q'_i. In the composed constraint q we then substitute the q'_i for the p'_i and obtain a final composed constraint q' such that

$$apx(sort \cap b) = dom(x_1, q') \times \cdots \times dom(x_{2n}, q').$$

*11.2. Exact projections of sort ∩ **a***

The computation of $apx(sort \cap a)$, for a given $2n$-block a, consists in computing an approximation of the $2n$ projections of the set $sort \cap a$. In fact, the result supplied by the algorithm of section 8 contains an almost explicit representation of the exact projections $\pi_i(sort \cap a)$.

We assume that $sort \cap a \neq \emptyset$ and we let $b = b_1 \times \cdots \times b_{2n} = apx(sort \cap a)$. In order to compute these b_i's, the algorithm transforms an initial basic constraint of the form

$$(x_1, \ldots, x_{2n}) \in sort \ \wedge \ x_1 \in a_1 \ \wedge \ \cdots \ \wedge \ x_{2n} \in a_{2n}$$

into a final composed constraint q such that $b_i = dom(x_i, q)$. This constraint q is of the form $p_1 \wedge \cdots \wedge p_k$, where each p_i is a basic constraint of the form

$$(x_{h_{(i,1)}}, \ldots, x_{h_{(i,2n_i)}}) \in sort \ \wedge \ x_{h_{(i,1)}} \in b_{h_{(i,1)}} \ \wedge \ \cdots \ \wedge \ x_{h_{(i,2n_i)}} \in b_{h_{(i,2n_i)}}.$$

By introducing the partition of $1..2n$,

$$P = \{c_1, \ldots, c_k\}, \text{ with } c_i = \{h_{(i,1)}, \ldots, h_{(i,2n_i)}\}, \tag{19}$$

we have the two following results:

THEOREM 5
(1) *for all* $i \in 1..n$, $\pi_i(sort \cap a) = b_i \cap (\bigcup_{j \in c_i \cap (n+1)..(2n)} b_j)$,
(2) *for all* $j \in (n+1)..(2n)$, $\pi_j(sort \cap a) = b_j$.

Sketch of the proof of (1): First it must be proven that the set P defined in (19) is the finest partition of $1..2n$ into b-autonomous classes. We skip its proof which is rather technical.

Then, according to Property 11, we have $\pi_{h_{(i,j)}}(sort \cap a) = \pi_j(sort \cap a')$, with $a' = b_{h_{(i,1)}} \times \cdots \times b_{h_{(i,2n_i)}}$. If the algorithm computes $b' = apx(sort \cap a')$, it will return $b' = a'$ and a partition P' of $1..2n_i$. Since P is the finest partition of $1..2n$ into b-autonomous classes, the partition P' will have $1..2n_i$ as a unique class.

Thus we can restrict the proof to the case where $a = b$ and the partition P has only one class $1..2n$. Then we must prove that, for each $i \in 1..n$,

$$\pi_i(sort \cap a) = \bigcup_{j \in (n+1)..(2n)} a_i \cap a_j.$$

By Property 4, it suffices to show that, for each $i \in 1..n$,

$$\bigcup_{j \in (n+1)..(2n)} a_i \cap a_j = \bigcup_{f \in \Phi} a_i \cap a_{f(i)},$$

where Φ denotes the set of bijections f from $1..n$ into $(n+1)..(2n)$ such that

$$\text{for each } i' \in 1..n, \quad a_{i'} \cap a_{f(i')} \neq \emptyset. \tag{20}$$

Let p be an element of $1..n$ and q an element of $(n+1)..(2n)$ such that $a_p \cap a_q \neq \emptyset$. It must be shown that there exists $f \in \Phi$ such that (20) and $f(p) = q$. From step (4) of the algorithm, it follows that there exists a bijection g from $1..n$ into $(n+1)..(2n)$, such that, for each $i \in 1..n$, on the one hand $a_i \cap a_{g(i)} \neq \emptyset$ and on the other hand $g(i) < 2n$ entails $a_i \cap a_{g(i)+1} \neq \emptyset$. Thus there exists a bijection g from $1..n$ into $(n+1)..(2n)$ such that

$$\begin{array}{ll} \text{for each } i \in 1..n, & a_i \cap a_{g(i)} \neq \emptyset, \\ \text{for each } i \in 1..n, & q \leq g(i) < 2n \text{ entails } a_i \cap a_{g(i)+1} \neq \emptyset. \end{array} \tag{21}$$

If $q \leq g(p)$, the following bijection f has property (20) and is such that $f(p) = q$:

$$f(i) = \begin{cases} g(i), & \text{if } g(i) \notin q \,..\, g(p), \\ g(i)+1, & \text{if } g(i) \in q \,..\, g(p)-1, \\ q, & \text{if } g(i) = g(p). \end{cases}$$

If $g(p) < q$, we perform successively the following operations:

1. Choose an element p' in $1..n$ such that $g(p) < g(p')$ and $a_{p'} \cap a_{g(p)} \neq \emptyset$. Such a p' always exists, otherwise the set $s \cup g^{-1}(s)$, with $s = (g(p)+1)..(2n)$, would be a strict a-autonomous subclass of $1..2n$.

2. If $g(p') < q$, replace g by the bijection g' which differs from g only by $(g'(p'), g'(p)) = (g(p), g(p'))$ and restart the sequence of operations from 1. Notice that we have still $g(p) < q$ and property (21), and that the iteration will terminate, because $g(q)$ strictly increases.

3. If $q \leq g(p')$, the configuration is such that $g(p) < q \leq g(p')$, $a_p \cap a_q \neq \emptyset$, $a_{p'} \cap a_{g(p)} \neq \emptyset$ and bijection g has property (21). By Property 3, we have also $a_{p'} \cap a_q \neq \emptyset$. By construction, the following bijection f has property (20) and is such that $f(p) = q$:

$$f(i) = \begin{cases} g(i), & \text{if } g(i) \notin g(p) \mathinner{..} g(p'), \\ q, & \text{if } g(i) = g(p), \\ g(i), & \text{if } g(i) \in g(p){+}1 \mathinner{..} q{-}1, \\ g(i){+}1, & \text{if } g(i) \in q \mathinner{..} g(p'){-}1, \\ g(p), & \text{if } g(i) = g(p'). \end{cases}$$

■

Proof of (2): We can suppose that block a equals block b and is of type II. Obviously we have $\pi_j(sort \cap a) \subseteq a_j$, for all $j \in (n{+}1)..(2n)$. It remains to be established that, given $k \in (n{+}1)..(2n)$ and $e \in a_k$, we have $e \in \pi_k(sort \cap a)$. By Lemma 1, Theorem 2 and Property 12, there exists a bijection ψ from $1..n$ into $(n{+}1)..(2n)$ such that the $2n$-tuple

$$(\underline{a_{\psi(1)}}, \ldots, \underline{a_{\psi(n)}}, \underline{a_{n+1}}, \ldots, \underline{a_{2n}}) \tag{22}$$

belongs to $sort \cap a$. Let k' be the smallest element of $(n{+}1)..(2n)$ such that $e \in a_{k'}$ and let $a' = a'_1 \times \cdots \times a'_{2n}$ be the block such that, for each $i \in 1..2n$, we have $a'_i = [\underline{a_i}, e]$ if $i \in k'..k$, and $a'_i = a_i$ otherwise. Since the $2n$-tuple (22) belongs also to $sort \cap a'$, we have $sort \cap a' \neq \emptyset$. The sets $sort \cap a$ and $sort \cap a'$ being non-empty and the blocks a and a' being of type II, by Theorem 2, there exist bijections φ, φ', from $1..n$ into $(n{+}1)..(2n)$, such that, for each $j \in (n{+}1)..(2n)$

$$\overline{\pi_j(sort \cap a)} = \overline{a_j \cap a_{\varphi^{-1}(j)}},$$
$$\overline{\pi_j(sort \cap a')} = \overline{a_j \cap a_{\varphi'^{-1}(j)}}$$

and such that, for each $i \in 1..n$, we have $\varphi(i) = min\,(rel\,(a, i) - \varphi(1..(i{-}1)))$ and $\varphi'(i) = min\,(rel\,(a', i) - \varphi'(1..(i{-}1)))$, that is, by Property 5, such that, for each $j \in (n{+}1)..(2n)$,

$$\varphi^{-1}(j) = min\,(rel\,(a, j) - \varphi^{-1}((n{+}1)..(j{-}1))),$$
$$\varphi'^{-1}(j) = min\,(rel\,(a', j) - \varphi'^{-1}((n{+}1)..(j{-}1)))$$

Since we have supposed that the blocks a and $apx(sort \cap a)$ are equal, for all $j \in (n{+}1)..(2n)$, we have $\overline{a_j} \preceq \overline{a_{\varphi^{-1}(j)}}$ and thus, for all $j \in k'..2n$, we have

$e \preceq \overline{a_{\varphi^{-1}(j)}}$. By construction, the mappings φ^{-1} and φ'^{-1} agree on $(n+1)..(k'-1)$. It follows that, for each $j \in k'..2n$ and in particular for $j = k$, we have $e \preceq \overline{a_{\varphi'^{-1}(j)}}$. Thus $\overline{\pi_k(sort \cap a')} = \overline{a_k} = e$ and $e \in \pi_k(sort \cap a)$. ■

11.3. Experimental complexity of the algorithm

We have tested the performances of our algorithm on $2n$-blocks, randomly constructed on $(\mathbf{Z}, \leq)$. Each of these blocks belongs to the set $\mathcal{B}(n, m)$ of $2n$-blocks $a = a_1 \times \cdots \times a_{2n}$ which are such that
(1) each a_i has at most m elements,
(1) there exists $d \in sort \cap [1, 4n]^{2n}$ with $d \in a$.
The value $4n$ has been chosen to allow the $4n$ endpoints of the block $[1, 4n]^{2n}$ to take all possible configurations, with respect to the relation $\leq$. The parameter m controls the number of elements of the set $sort \cap a$.

The construction of an element $a = a_1 \times \cdots \times a_{2n}$ of $\mathcal{B}(n, m)$ is made by randomly choosing an element d' of $[1, 4n]^n$, sorting it and obtaining d''. The $2n$-tuple $d = d'd''$ belonging to $sort \cap [1, 4n]^{2n}$, it is then easy to construct randomly the block a with $d \in a$, each a_i having at most m elements.

Table 2 below summarizes the behavior of our algorithm for computing $apx\,(sort \cap a)$ with an $a \in \mathcal{B}(n, m)$ for different values of n and m. The algorithm is programmed in Java, using the jdk1.2 development kit from Sun, and executed on a Pentium Pentium II-MMX at 450MHz under Windows NT 4.0. In the table, we denote by
k, the number of basic constraints obtained by the algorithm at the end,
t_1, the time, in milliseconds, for sorting the n elements of the n-tuple d', which is used for constructing a,
t_2, the time, in milliseconds, for calculating $apx\,(sort \cap a)$.

Table 2. Benchmarks.

2n	m	k	t_1	t_2	t_2/t_1
10^4	200	4	10	100	10.0
10^4	20	113	10	100	10.0
10^4	2	4 233	20	90	4.5
10^5	200	7	100	431	4.3
10^5	20	11 634	100	400	4.0
10^5	2	42 579	100	431	4.3
10^6	200	101	1 562	5 769	3.7
10^6	20	11 5654	1 663	4 837	2.9
10^6	2	42 4618	1 542	5 087	3.3
10^7	200	907	21 852	69 600	3.2
10^7	20	1 161 364	21 741	55 880	2.6
10^7	2	4 250 255	21 722	63 472	2.9

In our implementation, we have limited the number of sortings to: (1) one sorting of n intervals, in increasing order of their greatest elements, and (2), for each class of intervals generated in the first decomposition, one sorting of the intervals,

in decreasing order of their least elements. The chosen sorting algorithm is of complexity $\mathcal{O}(n \log n)$. It consists in successively sorting sub-sequences of lengths $2^1, 2^2, \ldots, 2^p$, with $p = \lceil \log_2 n \rceil$. Each sorting of 2^{i+i} elements is then performed by merging two lists of 2^i elements.

The binary balanced tree, of n nodes, necessary for the computation of φ is encoded by a vector of size n, with the classical technique of putting, in position $2i$, the left daughter and, in position $2i+1$, the right daughter of a node in position i.

The conclusion of the tests is that the time necessary for computing $apx\,(sort \cap a)$, with a being a block of dimension $2n$, is about 4 times the time necessary for sorting n integers.

We end with figure 2, which shows transformation $a \Longrightarrow apx\,(sort \cap a)$, for a randomly chosen block a in $\mathcal{B}(50, 50)$. For a better presentation we have rearranged the order of the first 50 intervals of the 100-block a.

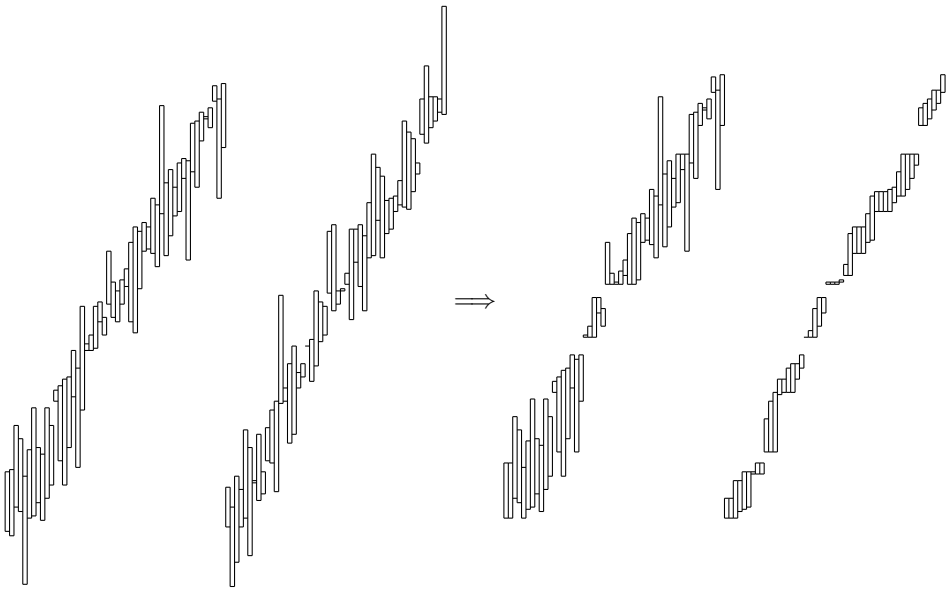

Figure 2. Transformation $a \Longrightarrow apx\,(sort \cap a)$, with a of dimension $50 + 50$.

Acknowledgments

We express our thanks to our colleague Thomas Grainger and to Franz Günthner, from the University of Munich, who helped us correct the English writing of this paper.

References

1. Benhamou F. and W.J. Older, Applying Interval Arithmetic to Real, Integer and Boolean Constraints. *Journal of Logic Programming,* 1997.
2. Knuth D.E., Sorting and Searching, The Art of Computer Programming, volume 3, Addison Wesley, 1973.
3. Older W.J. and A. Vellino, Extending Prolog with Constraint Arithmetic on Real Intervals, Proceedings of the Canadian Conference on Electrical and Computer Engineering, 1990
4. Older W.J., G.M. Swinkels and M.H. van Emden, Getting to the Real Problem: Experience with BNR Prolog in OR, in *Proceedings of the Third International Conference on the Practical Applications of Prolog*, (PAP'95 à Paris), Alinmead Software Ltd, ISBN 0 9 525554 0 9, April 1995.
5. Zhou J., A permutation-based approach for solving the job-shop problem, *Constraints*, vol 2, no 2, 185-213, October 1997.