

Efficient Enumeration of Solutions Produced by Closure Operations

Arnaud Mary, Yann Strozecki

▶ **To cite this version:**

Arnaud Mary, Yann Strozecki. Efficient Enumeration of Solutions Produced by Closure Operations. 33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, Feb 2016, Orléans, France. 10.4230/LIPIcs.STACS.2016.52 . hal-01388505

HAL Id: hal-01388505
https://inria.hal.science/hal-01388505

Submitted on 23 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Efficient enumeration of solutions produced by closure operations

Arnaud Mary[1] and Yann Strozecki[2]

1 **Université Lyon 1 ; CNRS, UMR5558, LBBE / INRIA - ERABLE**
2 **Université de Versailles Saint-Quentin-en-Yvelines, DAVID laboratory**

Abstract

In this paper we address the problem of generating all elements obtained by the saturation of an initial set by some operations. More precisely, we prove that we can generate the closure by polymorphisms of a boolean relation with a polynomial delay. Therefore we can compute with polynomial delay the closure of a family of sets by any set of "set operations" (e.g. by union, intersection, difference, symmetric difference...). To do so, we prove that for any set of operations $\mathcal{F}$, one can decide in polynomial time whether an elements belongs to the closure by $\mathcal{F}$ of a family of sets. When the relation is over a domain larger than two elements, we prove that our generic enumeration method fails, since the associated decision problem is NP-hard.

Keywords and phrases enumeration, set saturation, polynomial delay, Post's lattice

1 Introduction

In enumeration we are interested in listing a set of elements, which can be of exponential cardinality in the size of the input. The complexity of enumeration problems is thus measured in term of the input size and output size. The enumeration algorithms with a complexity polynomial in both the input and output are called output polynomial or total polynomial time. Another, more precise notion of complexity, is the *delay* which measures the time between the production of two consecutive solutions. We are especially interested in problems solvable with a delay polynomial in the input size, which are considered as the *tractable problems* in enumeration complexity. For instance, the maximal independent sets of a graph can be enumerated with polynomial delay [8].

If we allow the delay to grow during the algorithm, we obtain incremental delay algorithms: the first k solutions can be enumerated in a time polynomial in k and in the size of the input. Many problems which can be solved with an incremental delay have the following form: given a set of elements and a polynomial time function acting on tuples of elements, produce the closure of the set by the function. For instance, the best algorithm to generate all circuits of a matroid is in incremental delay because it uses some closure property of the circuits [9].

In this article, we try to understand when saturation problems which are natural incremental delay problems can be in fact solved by a polynomial delay algorithm. To attack this question we need to restrict the saturation operation. In this article, an element will be a vector over some finite set and we ask the saturation operation to act *coefficient-wise* and in the same way on each coefficient. We prove that, when the vector is over the boolean domain, every possible saturation can be computed in polynomial delay. To do that we study a decision version of our problem, denoted by $\textsc{Closure}_{\mathcal{F}}$: given a vector v and a set of vectors $\mathcal{S}$ decide whether v belongs to the closure of $\mathcal{S}$ by the operations of $\mathcal{F}$. We prove $\textsc{Closure}_{\mathcal{F}} \in \mathsf{P}$ for all set of operations $\mathcal{F}$ over the boolean domain.

When the domain is boolean, the problem can be reformulated in term of set systems or hypergraphs. It is equivalent to generating the smallest hypergraph which contains a given

licensed under Creative Commons License CC-BY

hypergraph and which is closed by some operations. We show how to efficiently compute the closure of an hypergraph by any family of set operations (any operation that is the composition of unions, intersections and complementations) on the hyperedges. This extends known methods such as the closure of a hypergraph by union, by union and intersection or the generation of the cycles of a graph by computing the closure of the fundamental cycles by symmetric difference. In general, knowing how to compute a closure may serve as a good tool to design other enumeration algorithms. One only has to express an enumeration problem as the closure of some sufficiently small and easy to compute set of elements and then to apply the algorithms presented in this article.

The closure computation is also related to constraint satisfaction problems (CSP). Indeed, the set of vectors can be seen as a relation R and the problem of generating its closure by some operation f is equivalent to the computation of the smallest relation R' containing R such that f is a polymorphism of R'. There are several works on enumeration in the context of CSP, which deal with enumerating solutions of a CSP in polynomial delay [5, 3]. The simplest such result [5] states that in the boolean case, there is a polynomial delay algorithm if and only if the constraint language is Horn, anti-Horn, bijunctive or affine. Our work is completely unrelated to these results, since we are not interested in the solutions of CSPs but only in generating the closure of relations. However, we use tools from CSPs such as the Post's lattice [11], used by Schaefer in its seminal paper [14], and the Baker-Pixley theorem [2].

The main theorem of this article settles the complexity of a whole family of decision problems and implies, quite surprisingly, that the backtrack search is enough to obtain a polynomial delay algorithm to enumerate any closure of boolean vectors. For all these enumeration problems, compared to the naive saturation algorithm, our method has a better time complexity (even from a practical point of view) and a better space complexity (polynomial rather than exponential). Moreover, besides the generic enumeration algorithm, we try to give for each closure rule an algorithm with the best possible complexity. In doing so, we illustrate several classical methods used to enumerate objects such as amortized backtrack search, hill climbing, Gray code ...

1.1 Organization of the paper

In Sec. 2, we define enumeration complexity, our problem and the backtrack search. In Sec. 3, we use Post's lattice, restricted through suitable reductions between clones, to determine the complexity of $\textsc{Closure}_{\mathcal{F}}$ for all set of binary operations $\mathcal{F}$. It turns out that there are only a few types of closure operations: the monotone operations (Sec. 3.1), the addition over $\mathbb{F}_2$ (Sec. 3.2), the set of all operations (Sec.3.3), two infinite hierarchies related to the majority function (Sec. 3.4) and the limit cases of the previous hierarchies (Sec.3.5). Finally, in Sec. 4, we give polynomial delay algorithms for three classes of closure operation over any domain and prove that the method we use in the boolean case fails.

2 Preliminary

Given $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, ..., n\}$. For a set D and a vector $v \in D^n$, we denote by v_i the i^{th} coordinate of v. Let $i, j \in [n]$, we denote by $v_{i,j}$ the vector (v_i, v_j). More generally, for a subset $I = \{i_1, ..., i_k\}$ of $[n]$ with $i_1 < ... < i_k$ we denote by v_I the vector $(v_{i_1}, ..., v_{i_k})$. Let $\mathcal{S}$ be a set of vector we denote by $\mathcal{S}_I$ the set $\{v_I \mid v \in \mathcal{S}\}$. The characteristic vector v of a subset E of $[n]$, is the vector in $\{0,1\}^n$ such that $v_i = 1$ if and only if $i \in X$.

2.1 Complexity

In this section, we recall basic definitions about enumeration problems and their complexity, for further details and examples see [15].

Let Σ be some finite alphabet. An *enumeration problem* is a function A from Σ^* to $\mathcal{P}(\Sigma^*)$. That is to each input word, A associates a set of words. An algorithm which solves the enumeration problem A takes any input word w and produces the set $A(w)$ word by word and *without redundancies*. We always require the sets $A(w)$ to be finite. We may also ask $A(w)$ to contain only words of polynomial size in the size of w and that one can test whether an element belongs to $A(w)$ in polynomial time. If those two conditions hold, the problem is in the class EnumP which is the counterpart of NP for enumeration. Because of this relationship to NP, we often call solutions the elements we enumerate.

The *delay* is the time between the productions of two consecutive solutions. Usually we want to bound the delay of an algorithm for all pairs of consecutive solutions and for all inputs of the same size. If this delay is polynomial in the size of the input, then we say that the algorithm is in *polynomial delay* and the problem is in the class DelayP. If the delay is polynomial in the input and the number of already generated solutions, we say that the algorithm is in *incremental delay* and the problem is in the class IncP. By definition we have DelayP $\subset$ IncP. Moreover (DelayP $\cap$ EnumP) $\neq$ (IncP $\cap$ EnumP) modulo the exponential time hypothesis [4]. In practice problems in DelayP are much more tractable, often because they can be solved with a memory polynomial in the size of the input. Note that in a polynomial delay algorithm we allow a polynomial precomputation step, usually to set up data structures, which is not taken into account in the delay. This is why we can have a delay smaller than the size of the input.

We now explain a very classical and natural enumeration method called the *Backtrack Search* (sometimes also called the *flashlight method*) used in many previous articles [12, 16]. We represent the solutions we want to enumerate as vectors of size n and coefficients in D. In practice solutions are often subsets of $[n]$ which means that $D = \{0, 1\}$ and the vector is the characteristic vector of the subset.

The enumeration algorithm is a depth first traversal of a tree whose nodes are partial solutions. The nodes of the tree will be all vectors v of size l, for all $l \leq n$, such that $v = w_{[l]}$ and w is a solution. The children of the node v will be the vectors of size $l + 1$, which restricted to $[l]$ are equal to v. The leaves of this tree are the solutions of our problem, therefore a depth first traversal will visit all leaves and yield all solutions. We want an enumeration algorithm with a delay polynomial in n. Since a branch of the tree is of size n, we need to be able to find the children of a node in a time polynomial in n to obtain a polynomial delay. The delay also depends linearly on $|D|$, but in the rest of the paper $|D|$ will be constant. Therefore the problem is reduced to the following *decision* problem: given v of size l is there w a solution such that $v = w_{[l]}$? This problem is called the *extension problem* associated to the enumeration problem.

▶ Proposition 1. Given an enumeration problem A, such that for all w, $A(w)$ can be seen as vectors of size n and coefficients in D, with n and $|D|$ polynomially related to $|w|$. If the extension problem associated to A is in P, then A is in DelayP.

2.2 Closure of families by set operations

We fix D a finite domain. Given a t-ary operation f (a function from D^t to D), f can be naturally extended to a t-ary operation over vectors of the same size. For a t-uples of vectors of size n $v^1, \dots v^t$, f will then acts coefficient-wise, that is for all $i \leq n$, $f(v^1, \dots, v^t)_i =$

$f(v_i^1, \ldots, v_i^t)$.

▶ **Definition 2.** Let $\mathcal{F}$ be a finite set of operations over D. Let $\mathcal{S}$ be a set of vectors of size n over D. Let $\mathcal{F}^i(\mathcal{S}) = \{f(v_1, \ldots, v_t) \mid v_1, \ldots, v_t \in \mathcal{F}^{i-1}(S) \text{ and } f \in \mathcal{F}\}$ and $\mathcal{F}^0(\mathcal{S}) = \mathcal{S}$. The closure of $\mathcal{S}$ by $\mathcal{F}$ is $Cl_{\mathcal{F}}(\mathcal{S}) = \cup_i \mathcal{F}^i(\mathcal{S})$.

Remark that $Cl_{\mathcal{F}}(\mathcal{S})$ is also the smallest set which contains $\mathcal{S}$ and which is closed by the operations of $\mathcal{F}$. The set $Cl_{\mathcal{F}}(\mathcal{S})$ is invariant under the operations of $\mathcal{F}$: these operations are called *polymorphisms* of the set $Cl_{\mathcal{F}}(\mathcal{S})$, a notion which comes from universal algebra.

As an illustration, assume that $D = \{0, 1\}$ and that $\mathcal{F} = \{\vee\}$. Then the elements of $\mathcal{S}$ can be seen as subsets of $[n]$ (each vector of size n is the characteristic vector of a subset of $[n]$) and $\textsc{Closure}_{\{\vee\}}(\mathcal{S})$ is the closure by union of all sets in S. Let $\mathcal{S} = \{\{1, 2, 4\}, \{2, 3\}, \{1, 3\}\}$ then $Cl_{\{\vee\}}(\mathcal{S}) = \{\{1, 2, 4\}, \{1, 2, 3, 4\}, \{2, 3\}, \{1, 3\}, \{1, 2, 3\}\}$. Remark that $Cl_{\{\vee\}}(\mathcal{S})$ is indeed closed by union, that is $\vee$ is a polymorphism of $Cl_{\{\vee\}}(\mathcal{S})$.

The problem we try to solve in this article, for all set of operations $\mathcal{F}$ over D, is $\textsc{EnumClosure}_{\mathcal{F}}$: given a set of vectors $\mathcal{S}$ compute $Cl_{\mathcal{F}}(S)$. We will always denote the size of the vectors of $\mathcal{S}$ by n and the cardinal of $\mathcal{S}$ by m. We introduce two related decision problems. First, the extension problem associated to a set of operations $\mathcal{F}$, is the problem $\textsc{ExtClosure}_{\mathcal{F}}$: given $\mathcal{S}$ a set of vectors of size n, and a vector v of size $l \leq n$, is there a vector $v' \in Cl_{\mathcal{F}}(\mathcal{S})$ such that $v_{[l]} = v'$. Second, the closure problem, denoted by $\textsc{Closure}_{\mathcal{F}}$, is a restricted version of the extension problem where v is of size n. As a direct consequence of Prop. 1, we have the following proposition.

▶ Proposition 3. If $\textsc{Closure}_{\mathcal{F}} \in \mathsf{P}$ then $\textsc{EnumClosure}_{\mathcal{F}} \in \mathsf{DelayP}$.

We have introduced an infinite family of problems, whose complexity we want to determine. Several families of operations may always produce the same closure. To deal with that, we need to introduce the notion of functional clone.

▶ **Definition 4.** Let $\mathcal{F}$ be a finite set of operations over D, the functional clone generated by $\mathcal{F}$, denoted by $< \mathcal{F} >$, is the set of operations obtained by any composition of the operations of $\mathcal{F}$ and of the projections $\pi_k^n : D^n \to D$ defined by $\pi_k^n(x_1, \ldots, x_n) = x_k$.

This notion is useful, because two sets of functions which generate the same clone applied to the same set produce the same closure.

▶ **Lemma 5.** *For all set of operations $\mathcal{F}$ and all set of vectors $\mathcal{S}$, $Cl_{\mathcal{F}}(\mathcal{S}) = Cl_{<\mathcal{F}>}(\mathcal{S})$.*

The number of clones over D is infinite even when D is the boolean domain (of size 2). However, in this case the clones form a countable lattice, called the Post's lattice [11]. Moreover there is a *finite* number of well described clones plus a few very regular infinite family of clones.

3 The Boolean Domain

In this part we will prove our main theorem on the complexity of $\textsc{Closure}_{\mathcal{F}}$, when the domain is boolean. An instance of one such problem, denoted by $\mathcal{S}$, will be indifferently seen as a set of vectors of size n or a set of subsets of $[n]$.

▶ **Theorem 6.** *Let $\mathcal{F}$ be any fixed finite set of operations over the boolean domain, then* $\textsc{Closure}_{\mathcal{F}} \in \mathsf{P}$ *and* $\textsc{EnumClosure}_{\mathcal{F}} \in \mathsf{DelayP}$.

To prove our main theorem, we will prove that $\textsc{Closure}_{\mathcal{F}} \in \mathsf{P}$, for each clone $\mathcal{F}$ of the Post's lattice. We first show that for some $\mathcal{F}$ the problem $\textsc{Closure}_{\mathcal{F}}$ can be reduced to $\textsc{Closure}_{\mathcal{G}}$ where $\mathcal{G}$ is another clone obtained from $\mathcal{F}$. This helps to reduce the number of cases we need to consider.

To an operation f we can associate its dual $\overline{f}$ defined by $\overline{f}(s_1, \dots, s_t) = \neg f(\neg s_1, \dots, \neg s_t)$. If $\mathcal{F}$ is a set of operations, $\overline{\mathcal{F}}$ is the set of duals of operation in $\mathcal{F}$. We denote by $\mathbf{0}$ and $\mathbf{1}$ the constant functions which always return 0 and 1. By a slight abuse of notation, we will also denote by $\mathbf{0}$ the all zero vector and by $\mathbf{1}$ the all one vector.

▶ Proposition 7. The following problems can be polynomially reduced to $\textsc{Closure}_{\mathcal{F}}$:

1. $\textsc{Closure}_{\mathcal{F}\cup\{\mathbf{0}\}}$, $\textsc{Closure}_{\mathcal{F}\cup\{\mathbf{1}\}}$, $\textsc{Closure}_{\mathcal{F}\cup\{\mathbf{0},\mathbf{1}\}}$
2. $\textsc{Closure}_{\overline{\mathcal{F}}}$
3. $\textsc{Closure}_{\mathcal{F}\cup\{\neg\}}$ when $\mathcal{F} = \overline{\mathcal{F}}$

In Fig. 1, we represent the clones which cannot be reduced to another one using Prop. 7 and that we will investigate in this article. For a a modern presentation of all boolean clones, their bases and the Post's lattice see [13].

Clone	Base
I_2	$\emptyset$
L_2	$x + y + z$
L_0	$x + y$
E_2	$\wedge$
S_{10}	$x \wedge (y \vee z)$
S_{10}^k	$Th_k^{k+1}, x \wedge (y \vee z)$
S_{12}	$x \wedge (y \to z)$
S_{12}^k	$Th_k^{k+1}, x \wedge (y \to z)$
D_2	maj
D_1	$maj, x + y + z$
M_2	$\vee, \wedge$
R_2	$x\,?\,y : z$
R_0	$\vee, +$

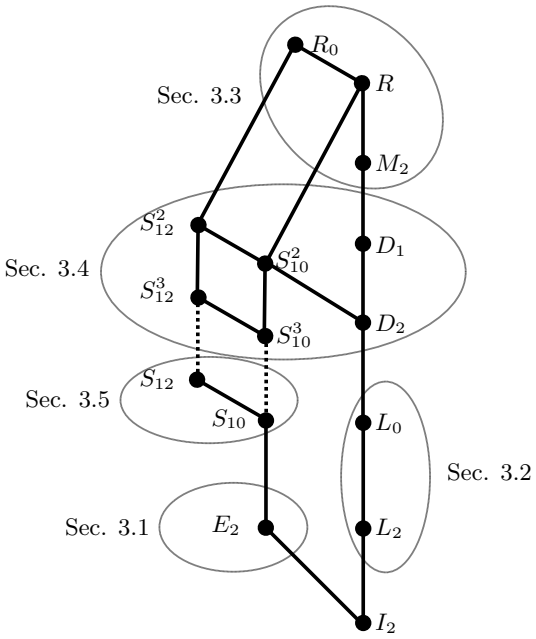


Figure 1 The reduced Post's lattice, the edges represent inclusions of clones

3.1 Conjunction

We first study one of the simplest clone: $E_2 =< \wedge >$. We give an elementary proof that $\textsc{Closure}_{E_2} \in \mathsf{P}$, then we explain how to obtain a good delay for $\textsc{EnumClosure}_{E_2}$. For a binary vector v, let us denote by $\mathbb{0}(v)$ (resp. $\mathbb{1}(v)$) the set of indices i for which $v_i = 0$ (resp. $v_i = 1$).

▶ Proposition 8. $\textsc{Closure}_{E_2} \in \mathsf{P}$.

Proof. Let $\mathcal{S}$ be a set of boolean vectors, if we apply $\wedge$ to a couple of vectors in $\mathcal{S}$ it produces the intersection of two vectors when seen as sets. Since the intersection operation

is associative and commutative, $Cl_{E_2}(\mathcal{S})$ is the set of arbitrary intersections of elements of $\mathcal{S}$. Let v be a vector and let $\mathcal{S}_1$ be the set $\{w \in \mathcal{S} \mid w_{\mathbb{1}(v)} = \mathbf{1}\}$. Assume now that v can be obtained as an intersection of elements $v_1, \ldots, v_t$, those elements must be in $\mathcal{S}_1$ because of the monotonicity of the intersection for the inclusion. On the other hand, by definition of $\mathcal{S}_1$, v will always be smaller or equal to $\cap_{w \in \mathcal{S}_1} w$. Therefore, $v \in Cl_{E_2}(\mathcal{S})$ if and only if $v = \cap_{w \in \mathcal{S}_1} w$. This intersection can be computed in time $O(mn)$ which concludes the proof. ◀

By Prop. 1, we can turn the algorithm for $\textsc{Closure}_{E_2}$ into an enumeration algorithm for $\textsc{EnumClosure}_{E_2}$ with delay $O(mn^2)$. We explain in the next proposition how to reduce this delay to $O(mn)$, which is the best known complexity for this problem.

▶ Proposition 9. There is an algorithm solving $\textsc{EnumClosure}_{E_2}$ with a delay $O(mn)$.

Proof. We use the backtrack search described in Prop. 1 but we maintain data structures which allow to decide $\textsc{Closure}_{E_2}$ quickly. Let $\mathcal{S}$ be the input set of m vectors of size n. During the traversal of the tree we update the partial solution p, represented by an array of size n which stores whether $p_i = 1$, $p_i = 0$ or is yet undefined.

A vector v of $\mathcal{S}$ is compatible with the partial solution if $\mathbb{1}_p \subseteq \mathbb{1}_v$. We maintain an array $COMP$ indexed by the sets of $\mathcal{S}$, which stores whether each vector of $\mathcal{S}$ is compatible or not with the current partial solution. Finally we update an array $COUNT$, such that $COUNT[i]$ is the number of compatible vectors $v \in \mathcal{S}$ such that $v_i = 0$. Remark that a partial solution p can be extended into a vector of $Cl_{E_2}(\mathcal{S})$ if and only if for all $i \in \mathbb{0}_p$ $COUNT[i] > 0$, the solution is then the intersection of all compatible vectors.

At each step of the traversal, we select an index i such that p_i is undefined and we set first $p_i = 0$ then $p_i = 1$. When we set $p_i = 0$, there is no change to do in $COUNT$ and $COMP$ and we can check whether this extended partial solution is correct by checking if $COUNT[i] > 0$ in constant time. When we set $p_i = 1$, we need to update $COMP$ by removing from it every vector v such that $v_i = 0$. Each time we remove such a vector v, we decrement $COUNT[j]$ for all j such that $v_j = 0$. If there is a j such that $COUNT[j]$ is decremented to 0 then the extension of p by $p_i = 1$ is not possible.

When we traverse a whole branch of the tree of partial solutions during the backtrack search, we will set $p_i = 1$ for each i at most once and then we need to remove each vector from $COMP$ at most once. Therefore the total number of operations we do to maintain $COMP$ and $COUNT$ is $O(mn)$ and so is the delay. ◀

The problem $\textsc{EnumClosure}_{E_2}$ is related to several interesting enumeration problems such as listing the solutions of a DNF formula. There is an intriguing open question on its complexity: can we have a delay sublinear in m or only dependent in n, that is a delay polynomial in the size of the solutions. It contrasts with all other clones, for which we give enumeration algorithms with a delay polynomial in the size of the solutions.

3.2 Algebraic operations

We first deal with the clone $L_0 =< + >$ where $+$ is the boolean addition. Note that $Cl_{L_0}(\mathcal{S})$ is the vector space generated by the vectors in $\mathcal{S}$. Seen as an operation on sets, $+$ is the symmetric difference of the two sets.

▶ Proposition 10. $\textsc{Closure}_{L_0} \in \mathsf{P}$.

Proof. Let $\mathcal{S}$ be the set of input vectors, let v be a vector and let A be the matrix whose rows are the elements of $\mathcal{S}$. The vector v is in $Cl_{L_0}(\mathcal{S})$ if and only if there is a solution to

$Ax = v$. Solving a linear system over $\mathbb{F}_2$ can be done in polynomial time which proves the proposition. ◀

The previous proposition yields a polynomial delay algorithm by applying Prop. 1. One can get a better delay, by computing in polynomial time a maximal free family M of $\mathcal{S}$, which is a basis of $Cl_{L_0}(\mathcal{S})$. The basis M is a succinct representation of $Cl_{L_0}(\mathcal{S})$. One can generate all elements of $Cl_{L_0}(\mathcal{S})$ by going over all possible subsets of elements of M and summing them. The subsets can be enumerated in constant time by using Gray code enumeration (see [10]). The sum can be done in time n by adding a single vector since two consecutive sets differ by a single element in the Gray code order. Therefore we have, after the polynomial time computation of M, an enumeration in delay $O(n)$.

With some care, we can extend this result to the clone L_2 generated by the sum modulo two of three elements.

▶ Proposition 11. $\textsc{Closure}_{L_2} \in \mathsf{P}$.

Proof. First remark that any vector in $Cl_{L_2}(\mathcal{S})$ is the sum of an odd number of vectors in $\mathcal{S}$. In other words $v \in Cl_{L_2}(\mathcal{S})$ if and only if there is a x such that $Ax = v$ and that the Hamming weight of x is odd. One can compute a basis B of the vector space of the solutions to the equation $Ax = v$. If all elements of B have Hamming weight even, then their sums also have Hamming weight even. Therefore $v \in Cl_{L_2}(\mathcal{S})$ if and only if there is an element in B with odd Hamming weight, which can be decided in polynomial time. ◀

3.3 Conjunction and disjunction

In this subsection, we deal with the largest possible clones of our reduced Post lattice: $M_2 =< \wedge, \vee >$, $R_2 =< x\,?\,y\,:\,z >$ and $R_0 =< \vee, + >$.

▶ Proposition 12. $\textsc{Closure}_{M_2} \in \mathsf{P}$.

Proof. Let $\mathcal{S}$ be a vector set and for all $i \in [n]$, let $X_i := \{v \in \mathcal{S} \mid v_i = 1\}$. We will show that a vector u belongs to $Cl_{M_2}(\mathcal{S})$ if and only if $u := \bigvee_{i \in \mathbb{1}(u)} \bigwedge_{v \in X_i} v$. Clearly, if $u := \bigvee_{i \in \mathbb{1}(u)} \bigwedge_{v \in X_i} v$ then $u \in Cl_{M_2}(\mathcal{S})$.

Assume first that there exists $i \in \mathbb{1}(u)$ such that $X_i = \emptyset$ i.e. for all $v \in \mathcal{S}$, $v_i = 0$. Then clearly, for all $w \in Cl_{M_2}(\mathcal{S})$, $w_i = 0$ and then $u \notin Cl_{M_2}(\mathcal{S})$. Assume now that $X_i \neq \emptyset$ for all $i \in \mathbb{1}(u)$ and assume that $u \neq t := \bigvee_{i \in \mathbb{1}(u)} \bigwedge_{v \in X_i} v$. So there exists $j \in \mathbb{0}(u)$ such that $t_j = 1$. Thus, there exists $i \in \mathbb{1}(u)$ such that for all $v \in X_i$, $v_j = 1$. We have that for all $v \in \mathcal{S}$, $v_i = 1 \implies v_j = 1$. Let us show that this property is preserved by both operations $\wedge$ and $\vee$ and then that this property holds for all $w \in Cl_{M_2}(\mathcal{S})$. Assume that the property holds for a set $\mathcal{F}$. Let $a, b \in \mathcal{F}$ and let $v := a \wedge b$. If $v_i = 1$, we have $a_i = 1$ and $b_i = 1$ and then $a_j = 1$ and $b_j = 1$. We conclude that $v_j = a_j \wedge b_j = 1$. Assume now that $v = a \vee b$ and that $v_i = 1$. Then either $a_i = 1$ or $b_i = 1$, say w.l.o.g. that $a_i = 1$. Then $a_j = 1$ and we have $v_j = a_j \vee b_j = 1$. We have shown that the property is preserved by both operations, therefore u cannot belong to $Cl_{M_2}(\mathcal{S})$ since $u_i = 1$ and $u_j = 0$. ◀

We can decide $\textsc{Closure}_{M_2}$ is $O(mn^2)$ therefore by applying Prop. 1, we get an enumeration algorithm with delay $O(mn^3)$. We can precompute the n vectors $x^i = \bigwedge_{v \in X_i} v$ and generate their unions in delay $O(n^2)$ thanks to Prop. 9. We can do better by using the inclusion structure of the x^i to obtain a $O(n)$ delay.

▶ Proposition 13. $\textsc{EnumClosure}_{M_2}$ can be solved with delay $O(n)$.

If we consider $\text{EnumClosure}_{M_2\cup\{\neg\}}(\mathcal{S})$, it is very easy to enumerate. Let $X^i = \{v \mid v \in \mathcal{S}, v_i = 1\} \cup \{\neg v \mid v \in \mathcal{S}, v_i = 0\}$ and let $x^i = \bigwedge_{v\in X_i} v$. The set $Cl_{M_2\cup\{\neg\}}(\mathcal{S})$ is in fact a boolean algebra, whose atoms are the x^i. Indeed, either $x^i_{i,j} = x^j_{i,j}$ and they are equal or $\mathbb{1}_{x^i} \cap \mathbb{1}_{x^j} = \emptyset$. Let $A = \{x^i \mid i \in [n]\}$, two distinct unions of elements in A produce distinct elements. Hence by enumerating all possible subsets of A with a Gray code, we can generate $Cl_{M_2\cup\{\neg\}}(\mathcal{S})$ with a delay $O(n)$ (even $O(1)$ when always equal coefficients are grouped together).

The closures by the clones R_2 and R_0 are equal to the closure by $M_2 \cup \{\neg\}$ up to some coefficients which are fixed to 0 or 1, thus they are as easy to enumerate.

▶ Proposition 14. The problems Closure_{R_2}, Closure_{R_0} can be reduced to Closure_{M_2} in polynomial time.

3.4 Majority and threshold

An operation f is a *near unanimity* of arity k if it satisfies $f(x_1, x_2, \ldots, x_k) = x$ for each k-tuple with at most one element different from x. The *threshold* function of arity k, denoted by Th^k_{k-1}, is defined by $Th^k_{k-1}(x_1, \ldots, x_k)$ is equal to 1 if and only if at least $k-1$ of the elements $x_1, \ldots, x_k$ are equal to one. It is the smallest near unanimity operation over the booleans. The threshold function Th^3_2 is the majority operation over three booleans that we denote by maj and the clone it generates is D_2. We first give a characterization of $Cl_{D_2}(\mathcal{S})$ which helps prove that $\text{Closure}_{D_2} \in P$. The characterization is a particular case of a universal algebra theorem that we then use to compute the closure by any clone which contains a threshold function.

▶ **Lemma 15.** *Let $\mathcal{S}$ be a vector set, a vector v belongs to $Cl_{D_2}(\mathcal{S})$ if and only if for all $i, j \in [n]$, $i \neq j$, there exists $x \in \mathcal{S}$ such that $x_{i,j} = v_{i,j}$.*

Proof. ($\Longrightarrow$) Given $a, b \in \{0, 1\}$ and $i, j \in [n]$, $i \neq j$, we first show that if for all $v \in \mathcal{S}$, $v_i \neq a$ or $v_j \neq b$ then for all $u \in Cl_{D_2}(\mathcal{S})$, $v_i \neq a$ or $v_j \neq b$. It is sufficient to prove that this property is preserved by applying maj to a vector set i.e. that if $\mathcal{S}$ has this property, then $maj(\mathcal{S})$ has also this property. Let $x, y, z \in \mathcal{S}$, $v := maj(x, y, z)$, and assume for contradiction that $v_{i,j} = (a, b)$. Since $v_i = a$, there is at least two vectors among $\{x, y, z\}$ that are equal to a at index i. Without loss of generality, let x and y be these two vectors. Since for all $u \in \mathcal{S}$, $u_i \neq a$ or $u_j \neq b$, we have $x_j \neq b$ and $y_j \neq b$ and then $v_j \neq b$ which contradicts the assumption. We conclude that if $v \in Cl_{D_2}(\mathcal{S})$, then for all $i, j \in [n]$, there exists $u \in \mathcal{S}$ with $v_{i,j} = u_{i,j}$.
($\Longleftarrow$) Let $k \leq n$ and let $a_1, ..., a_k \in \{0, 1\}$. We will show by induction on k, that if for all $i, j \leq k$ there exists $v \in \mathcal{S}$ with $v_i = a_i$ and $v_j = a_j$, then there exists $u \in Cl_{D_2}(\mathcal{S})$ with $u_1 = a_1$, $u_2 = a_2$, ..., $u_k = a_k$. The assertion is true for $k = 2$. Assume it is true for $k-1$, and let $a_1, ..., a_k \in \{0, 1\}$. By induction hypothesis there exists a vector $w \in Cl_{D_2}(\mathcal{S})$ with $w_1 = a_1$, ..., $w_{k-1} = a_{k-1}$. By hypothesis, for all $i \leq k$ there exists $v^i \in \mathcal{S}$ with $v^i_i = a_i$ and $v^i_k = a_k$. We then construct a sequence of vectors $(u^i)_{i\leq k}$ as follow. We let $u^1 = v^1$ and for all $1 < i < k$, $u^i = maj(w, u^{i-1}, v^i)$. We claim that $u := u^{k-1}$ has the property sought i.e. for all $i \leq k$, $u_i = a_i$. First let prove that for all $i < k$ and for all $j \leq i$, $u^i_j = a_j$. It is true for u_1 by definition. Assume now that the property holds for u^{i-1}, $i < k$. Then, by construction, for all $j \leq i-1$, we have $u^i_j = a_j$ since $w_j = a_j$ and $u^{i-1}_j = a_j$. Furthermore, we have $u^i_i = maj(w_i, u^{i-1}_i, v^i_i) = a_i$ since $w_i = a_i$ and $v_i = a_i$. We conclude that for all $i \leq k-1$, $u_i = u^{k-1}_i = a_i$.

We claim now that for all $i < k$, $u^i_k = a_k$. It is true for u^1. Assume it is true for u^{i-1}, $i < k$. Then we have $u^i_k = maj(w_k, u^{i-1}_k, v^i_k)$ which is equal to a_k since $u^{i-1}_k = a_k$ by

induction and $v_k^i = a_k$ by definition. We then have $u_i = a_i$ for all $i \leq k$ which concludes the proof. ◀

As an immediate consequence we get the following corollary and proposition.

▶ **Corollary 16.** $\textsc{Closure}_{D_2} \in \mathsf{P}$.

Proof. Using Lemma 15, one decides whether a vector v is in $Cl_{D_2}(S)$, by considering every pair of index i, j and checking whether there is a vector $w \in S$ such that $v_i, j = w_i, j$. The complexity is in $O(mn^2)$. ◀

▶ Proposition 17. $\textsc{EnumClosure}_{D_2}$ can be solved in delay $O(n^2)$.

Proof. We do a backtrack search and we explain how to efficiently decide $\textsc{Closure}_{D_2}$ during the enumeration. We first precompute for each pair (i, j) all values (a, b) such that there exists $v \in \mathcal{S}$, $v_{i,j} = (a, b)$. When we want to decide whether the vector v of size l can be extended into a solution, it is enough that it satisfies the condition of Lemma 15. Moreover, we already know that $v_{[l-1]}$ satisfies the condition of Lemma 15. Hence we only have to check that the values of $v_{i,l}$ for all $i < l$ can be found in $\mathcal{S}_{i,l}$ which can be done in time $O(l)$. The delay is the sum of the complexity of deciding $\textsc{Closure}_{D_2}$ for each partial solution in a branch: $O(n^2)$. ◀

It turns out that Lemma 15 is a particular case of a general theorem of universal algebra which applies to all near unanimity terms. However we felt it was interesting to give the lemma and its proof to get a sense of how the following theorem is proved.

▶ **Theorem 18** (Baker-Pixley, adapted from [2])**.** *Let $\mathcal{F}$ be a clone which contains a near unanimity term of arity k, then $v \in Cl_{\mathcal{F}}(\mathcal{S})$ if and only if for all set of indices I of size $k-1$, $v_I \in Cl_{\mathcal{F}}(\mathcal{S})_I$.*

This allows to settle the case of $D_1 =< maj, x+y+z >$ and of the two infinite families of clones of our restricted lattice $S_{10}^k =< Th_k^{k+1}, x \wedge (y \vee z) >$ and $S_{12}^k =< Th_k^{k+1}, x \wedge (y \to z) >$.

▶ **Corollary 19.** *If a clone $\mathcal{F}$ contains Th_k^{k+1} then $\textsc{Closure}_{\mathcal{F}}$ is solvable in $O(mn^k)$. In particular $\textsc{Closure}(S_{10}^k)$, $\textsc{Closure}(S_{12}^k)$ and $\textsc{Closure}(D_1)$ are in P.*

We have proved that the complexity of any closure problem in one of our infinite families is polynomial. Remark that we can use the method of Prop. 17 to obtain a delay $O(n^k)$ for enumerating the elements of a set closed by a near unanimity function of arity k. Notice that we could have applied Theorem 18 to the clones of Subsection 3.3 which all contain the maj function. However, it was relevant to deal with them separately to obtain a different algorithm with delay $O(n)$ rather than $O(n^2)$.

Notice that the complexity of $\textsc{Closure}_{\mathcal{F}}$ is increasing with the smallest arity of a near unanimity function in $\mathcal{F}$. We should thus investigate the complexity of the uniform problem when the clone is given as input. Let ClosureTreshold be the following problem: given a set $\mathcal{S}$ of vectors and an integer k decide whether the vector $\mathbf{1} \in Cl_{S_{10}^k}(\mathcal{S})$. It is a restricted version of the uniform problem, but it is already hard to solve because we can reduce the Hitting Set problem to its complementary.

▶ **Theorem 20.** ***ClosureTreshold*** *is* coNP*-complete.*

In fact, the result is even stronger. We cannot hope to get an FPT algorithm for ClosureTreshold parametrized by k since the Hitting Set problem parametrized by the size of the hitting set is W[2]-complete [6]. It means that if we want to significantly improve the delay of our enumeration algorithm for the clone S_{10}^k, we should drop the backtrack search since it relies on solving $\textsc{Closure}_{S_{10}^k}$.

3.5 Limits of the infinite parts

Here we deal with the two cases left which are the limits of the two infinite hierarchies of clones we have seen in the previous subsection. Let begin with $S_{12} =< x \wedge (y \to z) >$.

▶ Remark. Let $\mathcal{S}$ be a vector set and assume that there exists a $i \in [n]$ such that for all $v \in \mathcal{S}$, $v_i = 1$ (resp. $v_i = 0$) then for all $w \in Cl_{S_{12}}(\mathcal{S})$ we have $w_i = 1$ (resp. $w_i = 0$). Then we will assume in this section that for all $i \in [n]$ there is at least a vector v in $\mathcal{S}$ with $v_i = 1$ and a vector w with $w_i = 0$.

▶ **Theorem 21.** *Let $\mathcal{S}$ be a vector set, a vector v belongs to $Cl_{S_{12}}(\mathcal{S})$ if and only if*
- *there exists $w \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(w)$*
- *for all $(k,i) \in \mathbb{1}(v) \times \mathbb{0}(v)$ there exists $w \in \mathcal{S}$ with $w_{k,i} = (0,1)$ or $w_{k,i} = (1,0)$*

Proof. Let us start by proving the following claim.
Claim: Let $k, i \in [n]$. Then there exists $u \in Cl_{S_{12}}(\mathcal{S})$ such that $u_{k,i} = (1,0)$ if and only if there exists $v \in \mathcal{S}$ such that $v_{k,i} = (1,0)$ or $v_{k,i} = (0,1)$.

Assume first that there exists $v \in \mathcal{S}$ such that $v_{k,i} = (0,1)$. Let $x \in \mathcal{S}$ such that $x_k = 1$ and $y \in \mathcal{S}$ such that $y_i = 0$. Without loss of generality, such vectors exist by the assumption of Remark 3.5. Then $u := x \wedge (v \to y)$ has the sought property, i.e $u_{k,i} = (1,0)$. Assume now that for all $v \in \mathcal{S}$, $v_{k,i} \neq (1,0)$ and $v_{k,i} \neq (0,1)$. We show that this property is preserved by the application of $x \wedge (y \to z)$. For all $v \in \mathcal{S}$, $v_{k,i} = (1,1)$ or $v_{k,i} = (0,0)$. Since the function $x \wedge (y \to z)$ acts coordinate-wise on the vectors, if we consider $w = x \wedge (y \to z)$ with $x, y, z \in \mathcal{S}$ we must have $w_i = w_k$. Therefore $w_{k,i} \neq (1,0)$ and $w_{k,i} \neq (0,1)$ which implies by induction that there is no v with $v_{k,i} = (0,1)$ and $v \in Cl_{S_{12}}(\mathcal{S})$. We can now prove the theorem.
($\Longleftarrow$) We can simulate $w \wedge v$ with $w \wedge (w \to v)$. We will show that for all $i \in \mathbb{0}(v)$ either there exists a vector $v^i \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(v^i)$ and $v^i_i = 0$ or we can construct it. Notice that it is sufficient in order to prove that $v \in Cl_{S_{12}}(\mathcal{S})$ since we have $v = \bigwedge_{i \in \mathbb{0}(v)} v^i$. So let $i \in \mathbb{0}(v)$ and assume that for all $w \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(w)$ we have $w_i = 1$. Let w be such a vector and let $\mathbb{1}(v) = \{j_1, j_2, ..., j_k\}$. We will construct a sequence of vector $(w^l)_{l \leq k}$ such that for all $l \leq k$ and for all $r \leq l$, $w^l_{j_r} = 1$ and $w^l_i = 0$. Let w^1 be the vector with $w^1_{j_1} = 1$ and $w^1_i = 0$. By the claim, such a vector exists in $Cl_{S_{12}}(\mathcal{S})$. Now for all $l \leq k$, let us define $w^l := w \wedge (u^l \to w^{l-1})$ where u^l is a vector such that $u^l_{j_l} = 0$ and $u^l_i = 1$ and there is such a vector in $Cl_{S_{12}}(\mathcal{S})$ by the claim. Since by induction we have $w^{l-1}_i = 0$, and since $u^l_i = 1$, we have $(u^l \to w^{l-1})_i = 0$ and thus $w^l_i = 0$. Now since $u^l_{j_l} = 0$ and $w_{j_l} = 1$ we have $w^l_{j_l} = 1$. Finally, for all $r < l$, we have w_{j_r} and $w^{l-1}_{j_r} = 1$. Hence $w^l_{j_r} = 1$. We obtain that $\mathbb{1}(v) \subseteq \mathbb{1}(w^k)$ and $w^k_i = 0$.
($\Longrightarrow$) Let $v \in Cl_{S_{12}}(\mathcal{S})$. Notice that if $v = x \wedge (y \to z)$, then $\mathbb{1}(v) \subseteq \mathbb{1}(x)$. Thus, there exists $w \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(w)$. Now, by the claim, for all $k, i \in [n]$ such that $v_{k,i} = (1,0)$ there exists $w \in \mathcal{S}$ such that $w_{k,i} = (1,0)$ or $w_{k,i} = (0,1)$ which conclude the proof. ◀

▶ **Corollary 22.** $\textsc{Closure}_{S_{12}} \in \mathsf{P}$.

Finally, we deal with the clone $S_{10} =< x \wedge (y \vee z) >$. The characterization of $Cl_{S_{10}}(\mathcal{S})$ and its proof are very similar to the one of $Cl_{S_{12}}(\mathcal{S})$.

▶ **Theorem 23.** *Let $\mathcal{S}$ be a vector set, a vector v belongs to $Cl_{S_{10}}(\mathcal{S})$ if and only if*
- *there exists $w \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(w)$*
- *for all $(k,i) \in \mathbb{1}(v) \times \mathbb{0}(v)$ there exists $w \in \mathcal{S}$ with $w_{k,i} = (1,0)$*

▶ **Corollary 24.** $\textsc{Closure}_{S_{10}} \in \mathsf{P}$.

4 Larger Domains

In this section, we try to extend some results of the boolean domain to larger domains.

4.1 Tractable closure

The first tractable case is an extension of the clones of Subsection 3.4. Indeed using Th. 18, we can get an equivalent to Corollary. 19 and to Prop. 17 in any domain size.

▶ **Corollary 25.** *If $\mathcal{F}$ contains a near unanimity operation, then* $\textsc{Closure}_{\mathcal{F}} \in P$.

▶ Proposition 26. If $\mathcal{F}$ contains a near unanimity term of arity k, then $\textsc{EnumClosure}_{\mathcal{F}}$ can be solved in delay $O(n^{k-1})$.

The second tractable case is a generalization of Subsection 3.2.

▶ Proposition 27. Let f be a commutative group operation over D, then $\textsc{Closure}_{<f>} \in P$.

Proof. We want to solve $\textsc{Closure}_{<f>}$, given $\mathcal{S}$ a set of vectors and v a vector. Let A be the matrix which has the elements of $\mathcal{S}$ as rows. The vector v is in $\textsc{Closure}_{<f>}(\mathcal{S})$ if and only there is a vector x with coefficients in $\mathbb{Z}$ such that $Ax = v$. This equation is not over a field so we cannot solve it directly. We apply a classical group theorem to the finite commutative group (D, f), which states that D is a direct sum of cyclic groups $D_1, \dots, D_t$ whose order is the power of a prime. The equation $Ax = v$ can be seen as a set of equations over fields: $A_i x_i = v_i$, for $i \leq t$, where A_i, x_i and v_i are the projection of A, x and v over D_i. We can easily reconstruct an x which have the projections x_i on D_i by the Chinese remainder theorem. Therefore, deciding whether $v \in \textsc{Closure}_{<f>}(\mathcal{S})$ is equivalent to solving a set of linear systems and hence is in polynomial time. ◀

One natural generalization would be to allow the function f to be non commutative. In that case, we conjecture that $\textsc{Closure}_{<f>}$ is NP-hard.

4.2 A limit to the backtrack search

The last case we would like to extend is the clone generated by the conjunction. A natural generalization is to fix an order on D and to study the complexity of $\textsc{Closure}_{<f>}$ with f monotone. Let f be the function over $D = \{0, 1, 2\}$ defined by $f(x, y) = \min(x + y, 2)$. This function is clearly monotone for the usual order. However we can prove that EXACT-3-COVER reduces to $\textsc{Closure}_{<f>}$.

▶ Proposition 28. $\textsc{Closure}_{<f>}$ is NP-complete.

This hardness result implies that we cannot use the backtrack search to solve the associated enumeration algorithm. However, if we allow a space proportional to the number of solutions, we can still get a polynomial delay algorithm for associative functions, a property satisfied by the function f of the last proposition. Remark that the space used can be exponential while the backtrack search only requires a polynomial space.

▶ Proposition 29. If f is an associative function, then $\textsc{EnumClosure}_{<f>} \in$ DelayP.

Proof. Let $\mathcal{S}$ be an instance of $\textsc{EnumClosure}_{<f>}$. Let G be the directed graph with vertices $Cl_{<f>}(\mathcal{S})$ and from each $v \in Cl_{<f>}(\mathcal{S})$, there is an arc to $f(v, s)$ for all $s \in \mathcal{S}$. Since f is associative, by definition of G, every vertex of $Cl_{<f>}(\mathcal{S})$ is accessible from a vertex in $\mathcal{S}$. Therefore we can do a depth-first traversal of the graph G to enumerate all solutions. A step of the traversal is in polynomial time: from an element v we generate its

neighborhood: $f(v, s)$ for $s \in \mathcal{S}$. The computation of $f(v, s)$ is in time $O(n)$ and $|\mathcal{S}| = m$. We must also test whether the solution $f(v, s)$ has already been generated. This can be done in time $O(n)$ by maintaining a self balanced search tree containing the generated solutions, since there are at most $|D|^n$ solutions. In conclusion the delay of the enumeration algorithm is in $O(mn)$ thus polynomial. ◀

To obtain a polynomial space algorithm, we could try to use the *reverse search* method [1]. To do that, we want the graph G to be a directed acyclic graph, which is the case if we require the function to be monotone. The monotonicity also ensures that the depth of G is at most $n(|D| - 1)$. However we also need to be able to compute for each element of G a canonical ancestor in polynomial time and it does not seem to be easy even when f is monotone. We leave the question of finding a good property of f which ensures the existence of an easy to compute ancestor open for future research.

Acknowledgements Authors have been partly supported by the ANR project Aggreg and we thank the members of the project and Mamadou Kanté for interesting discussions about enumeration. We also thank Florent Madelaine for his help with CSP and universal algebra.

References

1 D. Avis and K. Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65(1):21–46, 1996.
2 Kirby A Baker and Alden F Pixley. Polynomial interpolation and the chinese remainder theorem for algebraic systems. *Mathematische Zeitschrift*, 143(2):165–174, 1975.
3 Andrei A Bulatov, Víctor Dalmau, Martin Grohe, and Dániel Marx. Enumerating homomorphisms. *Journal of Computer and System Sciences*, 78(2):638–650, 2012.
4 Florent Capelli, Arnaud Durand, and Yann Strozecki. A note on polynomial delay and incremental delay. *Available on:* `http://www.prism.uvsq.fr/~ystr`, 2015.
5 N. Creignou and JJ Hébrard. On generating all solutions of generalized satisfiability problems. *RAIRO Theoretical Informatics and Applications*, 31(6), 1997.
6 J. Flum and M. Grohe. *Parameterized complexity theory*. Springer-Verlag, 2006.
7 M.R. Garey and D.S. Johnson. *Computers and intractability: a guide to NP-completeness*. WH Freeman and Company, San Francisco, 1979.
8 David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988.
9 L. Khachiyan, E. Boros, K. Elbassioni, V. Gurvich, and K. Makino. On the complexity of some enumeration problems for matroids. *SIAM Journal on Discrete Mathematics*, 19(4):966–984, 2005.
10 Donald E Knuth. Combinatorial algorithms, part 1, volume 4a of the art of computer programming, 2011.
11 Emil Leon Post. *The two-valued iterative systems of mathematical logic*. Princeton University Press, 1941.
12 RC Read and RE Tarjan. Bounds on backtrack algorithms for listing cycles, paths, and spanning trees. *Networks*, 5(3):237–252, 1975.
13 Steffen Reith and Heribert Vollmer. Optimal satisfiability for propositional calculi and constraint satisfaction problems. *Information and Computation*, 186(1):1–19, 2003.
14 T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, page 226. ACM, 1978.
15 Y. Strozecki. *Enumeration complexity and matroid decomposition*. PhD thesis, Université Paris Diderot - Paris 7, 2010.
16 Yann Strozecki. On enumerating monomials and other combinatorial structures by polynomial interpolation. *Theory Comput. Syst.*, 53(4):532–568, 2013.

A Appendix

You can find here the proofs missing in the article. You can also read the long version of the paper on ArXiv.

▶ Proposition 3. If $\textsc{Closure}_{\mathcal{F}} \in \mathsf{P}$ then $\textsc{EnumClosure}_{\mathcal{F}} \in \mathsf{DelayP}$.

Proof. $\textsc{ExtClosure}_{\mathcal{F}}$ can be reduced to $\textsc{Closure}_{\mathcal{F}}$. Indeed, given a vector v of size l, because the operations of $\mathcal{F}$ act coordinate-wise, the two following predicates are equivalent:

- $\exists v' \in Cl_{\mathcal{F}}(\mathcal{S})$ such that $v_{[l]} = v'$
- $v \in Cl_{\mathcal{F}}(\mathcal{S}_{[l]})$

Therefore if $\textsc{Closure}_{\mathcal{F}} \in \mathsf{P}$ then we have also $\textsc{ExtClosure}_{\mathcal{F}} \in \mathsf{P}$. We can use Prop. 1 to conclude. ◀

▶ Proposition 7. The following problems can be polynomially reduced to $\textsc{Closure}_{\mathcal{F}}$:

1. $\textsc{Closure}_{\mathcal{F}\cup\{\mathbf{0}\}}$, $\textsc{Closure}_{\mathcal{F}\cup\{\mathbf{1}\}}$, $\textsc{Closure}_{\mathcal{F}\cup\{\mathbf{0},\mathbf{1}\}}$
2. $\textsc{Closure}_{\overline{\mathcal{F}}}$
3. $\textsc{Closure}_{\mathcal{F}\cup\{\neg\}}$ when $\mathcal{F} = \overline{\mathcal{F}}$

Proof of Prop. 7.

Proof. The reductions follow easily from these observations:

1. $Cl_{\mathcal{F}\cup\{f\}}(\mathcal{S}) = Cl_{\mathcal{F}}(\mathcal{S} \cup \{f\})$ for $f = \mathbf{0}$ or $f = \mathbf{1}$ and $\mathcal{S} \neq \emptyset$.
2. $Cl_{\overline{\mathcal{F}}}(\mathcal{S}) = Cl_{\mathcal{F}}(\overline{\mathcal{S}})$ where $\overline{\mathcal{S}}$ is the set of negation of vectors in $\mathcal{S}$.
3. $Cl_{\mathcal{F}\cup\{\neg\}}(\mathcal{S}) = Cl_{\mathcal{F}}(\mathcal{S}\cup\overline{\mathcal{S}})$ since for every $f \in \mathcal{F}$, there exists $g \in \mathcal{F}$ such that $\neg(f(v_1, \dots, v_t) = \overline{f}(\neg v_1, \dots, \neg v_t) = g(\neg v_1, \dots, \neg v_t)$.

◀

▶ Proposition 13. $\textsc{EnumClosure}_{M_2}$ can be solved with delay $O(n)$.

Proof. Let $\mathcal{S}$ be the input. We first build the $x^i = \bigwedge_{v\in\mathcal{S}, v_i=1} v$. The inclusion is a partial order between the x_i, we extend it into some total order T by topological sorting. We then generate all elements of $Cl_{M_2}(S)$ by an Hill climbing algorithm: we go from one solution to another by adding a single x^i. Let v be the current solution, we maintain a list L of the indices i of v such that $v_i = 0$. At each step we select i the first element of L and we set $v_j = 1$ and remove j from L for all $j \in x^i$. This produces a new solution in time $O(n)$. We then recursively call the algorithm on this new solution and list. When the recursive call is finished, we call the algorithm on v and $L \setminus \{i\}$.

This algorithm is correct, because the solutions generated in the two recursive calls are disjoint. Indeed, in the second call v_i will always be 0, because all indices in L are smaller than i in T. It means that x^j for $j \in L$ is either smaller or incomparable. Since x^i is the smallest element with $x^i_i = 1$ it implies that $x^j_i = 0$. ◀

▶ Proposition 14. The problems $\textsc{Closure}_{R_2}$, $\textsc{Closure}_{R_0}$ can be reduced to $\textsc{Closure}_{M_2}$ in polynomial time.

Proof. Let $\mathcal{S}$ be a set of binary vectors. If for some i, for all $v \in \mathcal{S}$, $v_i = 0$ (resp. 1) then for all $w \in Cl_{R_2}(\mathcal{S})$, $w_i = 0$ (resp. 1). Therefore, we can assume that for all i, there is u and v in $\mathcal{S}$ such that $u_i = 0$ and $v_i = 1$. Remark that $x\,?\,x : y = x \vee y$ thus by the previous assumption we can generate $\mathbf{1}$. Let $w^l = w^{l-1}\,?\,u^l : w^{l-1}$. By assumption, we can chose u^l such that $u^l_l = 0$. We set $w_0 = u_0$ and by a trivial induction $w_n = \mathbf{0}$. Now remark that $x\,?\,\mathbf{0} : \mathbf{1} = \overline{x}$. Therefore we have $Cl_{R_2}(\mathcal{S}) = Cl_{<\vee,\neg>}$ and the problem $\textsc{Closure}_{<\vee,\neg>}$ can be polynomially reduced to $\textsc{Closure}_{M_2}$ by point 3 of Prop. 7.

If for some i, for all $v \in \mathcal{S}$, $v_i = 0$ then for all $w \in Cl_{R_0}(\mathcal{S})$, $w_i = 0$. Therefore we can assume that for all i, there is $u \in \mathcal{S}$ such that $u_i = 1$. Therefore, $\mathbf{1} \in Cl_{R_0}(\mathcal{S})$ by doing the union of the elements u^l such that $u^l_l = 1$. Finally, $x + \mathbf{1} = \overline{x}$ therefore we also have $Cl_{R_0}(\mathcal{S}) = Cl_{<\vee,\neg>}$. ◀

▶ **Corollary 19.** *If a clone $\mathcal{F}$ contains Th^{k+1}_k then* Closure$_{\mathcal{F}}$ *is solvable in $O(mn^k)$. In particular* Closure(S^k_{10}), Closure(S^k_{12}) *and* Closure(D_1) *are in* P.

Proof. Let $\mathcal{S}$ bet a set of vectors and let v be a vector. By Th. 18, $v \in Cl_{\mathcal{F}}(\mathcal{S})$ if and only if for all I, $v_I \in Cl_{\mathcal{F}}(\mathcal{S})_I$. First remark that $Cl_{\mathcal{F}}(\mathcal{S})_I = Cl_{\mathcal{F}}(\mathcal{S}_I)$ because the functions of $\mathcal{F}$ act coefficient-wise on $\mathcal{S}$. The algorithm generates for each I of size k the set $Cl_{\mathcal{F}}(\mathcal{S}_I)$. For a given I, we first need to build the set $\mathcal{S}_I$ in time m and then the generation of $Cl_{\mathcal{F}}(\mathcal{S}_I)$ can be done in constant time. Indeed, we can apply the classical incremental algorithm to generate the elements in $Cl_{\mathcal{F}}(\mathcal{S}_I)$, and the cardinal of $Cl_{\mathcal{F}}(\mathcal{S}_I)$ only depends on k which is a constant. The time to generate all $Cl_{\mathcal{F}}(\mathcal{S}_I)$ is $O(mn^k)$ and then all the tests can be done in $O(n^k)$. ◀

▶ **Theorem 20.** ***ClosureTreshold*** *is* coNP*-complete.*

Proof. First notice that the problem is in coNP since by Theorem 18, the answer to the problem is negative if and only if one can exhibit a subset of indices of I of size k such that no elements of $\mathcal{S}_I$ is equal to $\mathbf{1}$.

Let us show that the Hitting Set problem can be reduced to ClosureTreshold. Given a hypergraph $\mathcal{H} = (V, \mathcal{E})$, the Hitting set problem asks whether there exists a subset $X \subseteq V$ of size k that intersects all the hyperedges of $\mathcal{H}$. This problem is a classical NP-complete problem [7]. Let $\mathcal{H} = (V, \mathcal{E})$ be a hypergraph and k be an integer. Let $\bar{\mathcal{H}}$ be the hypergraph on V whose hyperedges are the complementary of the hyperedges of $\mathcal{H}$, and let $\mathcal{S}$ be the set of characteristic vectors of the hyperedges of $\bar{\mathcal{H}}$. Then $\mathcal{H}$ has a transversal of size k if and only if there is a set I of indices of size k such for all $v \in \mathcal{S}_I$, $v \neq \mathbf{1}$. Indeed, I is a hitting set of $\mathcal{H}$ if for all $E \in \mathcal{E}$, there exists $i \in I$ such that $i \in E$ which implies that $i \notin \overline{E}$ and then the characteristic vector v of $\overline{E}$ is such that $v_i = 0$.

Let us show that a set I of indices of size k is such that no element in $\mathcal{S}_I$ is equal to $\mathbf{1}$ if and only if no element of $Cl_{S^k_{10}}(\mathcal{S}_I)$ is equal to $\mathbf{1}$. We assume that $k \geq 3$ hence $S^k_{10} =< Th^{k+1}_k >$. Remark that if no element in $\mathcal{S}_I$ is equal to $\mathbf{1}$, then the application of Th^{k+1}_k to $\mathcal{S}_I$ preserves this property. Indeed, let consider $Th^{k+1}_k(v^1, \ldots, v^{k+1})$, each v^i has a zero coefficient and since there are $k+1$ such vectors and the vectors are of size k, by the pigeonhole principle, there are i, j, l such that $v^i_l = v^j_l = 0$. This implies that $Th^{k+1}_k(v^1, \ldots, v^{k+1}) \neq \mathbf{1}$.

Since the other direction is straightforward, we have thus proved that there is a set I of indices of size k such that for all $v \in \mathcal{S}_I$, $v \neq \mathbf{1}$ if and only if there is a set I of indices of size k such that for all $v \in Cl_{S^k_{10}}(\mathcal{S}_I)$, $v \neq \mathbf{1}$. By Theorem 18, the later property is equivalent to $\mathbf{1} \notin Cl_{S^k_{10}}(\mathcal{S})$. Therefore we have given a polynomial time reduction from Hitting set to the complementary of ClosureTreshold which proves the proposition. ◀

▶ **Theorem 23.** *Let $\mathcal{S}$ be a vector set, a vector v belongs to $Cl_{S_{10}}(\mathcal{S})$ if and only if*

- *there exists $w \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(w)$*
- *for all $(k, i) \in \mathbb{1}(v) \times \mathbb{0}(v)$ there exists $w \in \mathcal{S}$ with $w_{k,i} = (1, 0)$*

Proof. ($\Longleftarrow$) Assume first that $v \in Cl_{S_{10}}(\mathcal{S})$. Notice that if $v = x \wedge (y \vee z)$, then $\mathbb{1}(v) \subseteq \mathbb{1}(x)$. Thus by a simple induction, there exists $w \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(w)$.

Now let $(k, i) \in \mathbb{1}(v) \times \mathbb{0}(v)$. Let us show that if for all $w \in \mathcal{S}$ $w_{k,i} \neq (1, 0)$, then $u_{k_i} \neq (1, 0)$ for all $u \in Cl_{S_{10}}(\mathcal{S})$ and then $v \notin Cl_{S_{10}}(\mathcal{S})$. It is sufficient to show that this

property is preserved by the operation $x \wedge (y \vee z)$. So let a, b and c be three boolean vectors such that $a_{k,i} \neq (1,0)$, $b_{k,i} \neq (1,0)$, $c_{k,i} \neq (1,0)$ and let $d = a \wedge (b \vee c)$. Assume that $d_i = 0$. Then either $a_i = 0$ or both b_i and c_i are 0. If $a_i = 0$ then $a_k = 0$ since $a_{k,i} \neq (1,0)$ and then $d_k = a_k \wedge (b_k \vee c_k = 0) = 0$. Now if $b_i = 0$ and $c_i = 0$, we have $b_k = 0$ and $c_k = 0$ and then $d_k = a_k \wedge (b_k \vee c_k = 0) = 0$. We conclude that $d_{k,i} \neq (0,1)$

($\Longrightarrow$) Assume that there exists $u \in \mathcal{S}$ such that $\mathbb{1}(v) \subseteq \mathbb{1}(u)$ and for all $(k,i) \in \mathbb{1}(v) \times \mathbb{0}(v)$ there exists $w \in \mathcal{S}$ with $w_{k,i} = (1,0)$. Notice that $\wedge \in S_{10}$ since $a \wedge b = a \wedge (b \vee b)$. Let $t := \bigwedge_{u \in \mathcal{S},\ \mathbb{1}(v) \subseteq \mathbb{1}(u)} u$. We have $\mathbb{1}(v) \subseteq \mathbb{1}(t)$. Either $v = t$ or there is $i \in [n]$ for which $v_i = 0$ and $t_i = 1$. For each such coordinate i, we will show how to construct a vector t' such that $t'_i = 0$ and such that $\mathbb{1}(v) \subseteq \mathbb{1}(t') \subseteq \mathbb{1}(t)$. Let $x := \bigvee_{u \in \mathcal{S},\ u_i = 0} u$. Notice that $x_i = 0$ and since for all $j \in \mathbb{1}(v)$ there exists $w \in \mathcal{S}$ such that $w_{i,j} = (0,1)$ we have $\mathbb{1}(v) \subseteq \mathbb{1}(x)$. Now let us define $t' := t \wedge x$. It is easy to see that t' satisfies the conditions sought. To construct t' we proceed as follow. Let $\{y^1, ..., y^k\} := \{u \in \mathcal{S} \mid u_i = 0\}$. Then let us construct the following sequence of vectors $t^1 := t \wedge (y^1 \vee y^2)$, $t^2 := t \wedge (t^1 \vee y^3)$, ..., $t^{k-1} := t \wedge (t^{k-2} \vee y^k)$. It is easy to see that $t' = t^{k-1}$, and then $t' \in Cl_{S_{10}}(\mathcal{S})$. We conclude that $v \in Cl_{S_{10}}(\mathcal{S})$. Indeed starting from t, we can apply the previous procedure to set to 0 each index i for which $v_i = 0$ and $t_i = 1$. ◀

▶ Proposition 28. $\textsc{Closure}_{<f>}$ is NP-complete, with $f(x,y) = \min(x + y, 2)$.

Proof. We reduce EXACT-3-COVER to $\textsc{Closure}_{<f>}$. Let $\mathcal{S}$ be an instance of EXACT-3-COVER, that is a set of subsets of $[n]$ of size 3. Clearly, $\mathcal{S}$ can be seen as an instance of $\textsc{Closure}_{<f>}$ and we prove that $\mathbf{1} \in \textsc{Closure}_{<f>}(\mathcal{S})$ if and only if there is an exact cover of $\mathcal{S}$. First remark that f associative, therefore any element of $\textsc{Closure}_{<f>}(\mathcal{S})$ can be written $f(v_1, f(v_2, f(v_3, \dots)$ with $v_i \in \mathcal{S}$. It is also commutative therefore we can associate a unique element of $\textsc{Closure}_{<f>}(\mathcal{S})$ to a multiset of elements of $\mathcal{S}$ by the previous construction. Remark that it is never useful to have three times the same element in the multiset since $f(v,v) = f(v, f(v,v))$. If $v_i > 0$ then $f(v_i, v_i) = 2$, therefore the vector $\mathbf{1}$ can only be generated by a set and not a multiset. Moreover a set which generates $\mathbf{1}$ satisfies that for all $i \leq n$ there is one and only one of its elements with a coefficient 1 at the index i. Such a set is an exact cover of $\mathcal{S}$, which proves the reduction.

The problem is in NP because an element v is in $\textsc{Closure}_{<f>}(\mathcal{S})$, if and only if there is a multiset of elements of $\mathcal{S}$ such that applying f to its elements yields v. This witness is of polynomial size since each element is at most twice in the multiset. ◀