Theoretical Computer Science 856 (2021) 68–74

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Generating clause sequences of a CNF formula

Kristóf Bérczi [a,*], Endre Boros [b], Ondřej Čepek [c], Khaled Elbassioni [d], Petr Kučera [c], Kazuhisa Makino [e]

[a] MTA-ELTE Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, Hungary
[b] MSIS Department and RUTCOR, Rutgers University, NJ, USA
[c] Charles University, Faculty of Mathematics and Physics, Department of Theoretical Computer Science and Mathematical Logic, Praha, Czech Republic
[d] EECS Department, Khalifa University of Science and Technology, Abu Dhabi, United Arab Emirates
[e] Research Institute for Mathematical Sciences (RIMS), Kyoto University, Kyoto, Japan

ARTICLE INFO

Article history:
Received 16 February 2020
Received in revised form 23 November 2020
Accepted 7 December 2020
Available online 14 December 2020
Communicated by L.M. Kirousis

Keywords:
CNF formulas
Clause sequences
Enumeration
Generation

ABSTRACT

Given a CNF formula Φ with clauses $C_1, \ldots, C_m$ and variables $V = \{x_1, \ldots, x_n\}$, a truth assignment $\boldsymbol{a} : V \to \{0, 1\}$ of Φ leads to a clause sequence $\sigma_\Phi(\boldsymbol{a}) = (C_1(\boldsymbol{a}), \ldots, C_m(\boldsymbol{a})) \in \{0, 1\}^m$ where $C_i(\boldsymbol{a}) = 1$ if clause C_i evaluates to 1 under assignment $\boldsymbol{a}$, otherwise $C_i(\boldsymbol{a}) = 0$. The set of all possible clause sequences carries a lot of information on the formula, e.g. SAT, MAX-SAT and MIN-SAT can be encoded in terms of finding a clause sequence with extremal properties.

We consider a problem posed at Dagstuhl Seminar 19211 "Enumeration in Data Management" (2019) about the generation of all possible clause sequences of a given CNF with bounded dimension. We prove that the problem can be solved in incremental polynomial time. We further give an algorithm with polynomial delay for the class of tractable CNF formulas. We also consider the generation of maximal and minimal clause sequences, and show that generating maximal clause sequences is NP-hard, while minimal clause sequences can be generated with polynomial delay.

1. Introduction

The concept of *well-designed pattern trees* was introduced by Letelier et al. [1] as a convenient graphic representation of conjunctive queries extended by the optional operator. The nodes of such a tree correspond to the queries, while the tree itself represents the optional extensions. Well-designed pattern trees have been studied from a complexity point of view in several aspects. One of the most interesting problems in the context of query languages is the *generation problem*, that is, generating the solutions one after the other without repetition.

Previous work The generation problem was studied for First-Order and Conjunctive Queries [2–5] and for well-designed pattern trees [1]. Recently, Kröll et al. [6] initiated a systematic study of the complexity of the generation problem of well-designed pattern trees. They identified several tractable and intractable cases of the problem both from a classical and from a parameterized complexity point of view. One class of pattern trees however remained unclassified. For a class $\mathcal{C}$ of

* Corresponding author.
E-mail addresses: berkri@cs.elte.hu (K. Bérczi), endre.boros@rutgers.edu (E. Boros), cepek@ktiml.mff.cuni.cz (O. Čepek), khaled.elbassioni@ku.ac.ae (K. Elbassioni), kucerap@ktiml.mff.cuni.cz (P. Kučera), makino@kurims.kyoto.ac.jp (K. Makino).

https://doi.org/10.1016/j.tcs.2020.12.021

conjunctive queries, a well-designed pattern tree T is *globally in* $\mathcal{C}$ if for every subtree T' of T the corresponding conjunctive query is also in $\mathcal{C}$. The *treewidth* of a conjunctive query is the treewidth of its Gaifman-graph [7]. In [6], the complexity of the generation problem for the class of well-designed pattern trees falling globally in the class of queries of treewidth at most k and having c-semi-bounded interface was left open (see [6, Table 1 on page 16]).

At the Dagstuhl Seminar 19211 "Enumeration in Data Management", Kröll proposed an open problem on the generation of clause sequences of CNF formulas [8, Problem 4.7]. The problem is motivated by the fact that it can be reduced to the above mentioned unsolved case of pattern trees, thus any bound on the generation complexity would be helpful in understanding the general problem. A *generation algorithm* outputs the objects in question one by one without repetition. We call it a *polynomial delay* procedure if the computing time between any two consecutive outputs is bounded by a polynomial of the input size. We call it *incrementally polynomial*, if for any k the first k objects can be generated in polynomial time in the input size and k. Finally, it is called *total polynomial* if all N objects are generated in polynomial time in the input size and N.

The problem studied in this paper can be formalized as follows. Let $V = \{x_1, \dots, x_n\}$ be a set of n Boolean variables. We denote by $\bar{x}_j = 1 - x_j$ the *negation* of variable x_j. Variables and their negations together are called *literals*. A *clause* is an elementary disjunction of literals and a CNF is a conjunction of clauses. We view clauses also as subsets of the literals, and CNFs as sets of clauses. Given a CNF $\Phi = C_1 \wedge \cdots \wedge C_m$ and an assignment $\boldsymbol{a} : V \to \{0, 1\}$, the corresponding binary sequence $\sigma_\Phi(\boldsymbol{a}) = (C_1(\boldsymbol{a}), \dots, C_m(\boldsymbol{a}))$ is called a *signature*[1] of Φ, that is, $C_i(\boldsymbol{a}) = 1$ if clause C_i evaluates to 1 under assignment a, and $C_i(\boldsymbol{a}) = 0$ otherwise. In particular, this means that Φ is satisfiable if and only if there exists some assignment $\boldsymbol{a}$ with $\sigma_\Phi(\boldsymbol{a}) = (1, \dots, 1)$. Moreover, MAX-SAT and MIN-SAT can be encoded by asking for a signature with the largest and smallest sum of elements, respectively.

As an example, consider the CNF formula $\Phi = C_1 \wedge C_2 \wedge C_3 \wedge C_4$, where $C_1 = x_1 \vee \bar{x}_3$, $C_2 = \bar{x}_2$, $C_3 = x_1 \vee x_2 \vee x_3$ and $C_4 = x_2 \vee \bar{x}_3$. Then assignment $\boldsymbol{a_1} = \{x_1 \mapsto 1, x_2 \mapsto 1, x_3 \mapsto 1\}$ leads to signature $\sigma_\Phi(\boldsymbol{a_1}) = (1, 0, 1, 1)$, while assignment $\boldsymbol{a_2} = \{x_1 \mapsto 0, x_2 \mapsto 0, x_3 \mapsto 1\}$ leads to signature $\sigma_\Phi(\boldsymbol{a_2}) = (0, 1, 1, 0)$. It is easy to see that Φ has six different signatures. In general, if the number of signatures is $\Omega(2^n)$, then generating them in total polynomial time is not difficult. However, their number may be $o(2^n)$, presenting a potential challenge for generation.

Given a CNF $\Phi = C_1 \wedge \cdots \wedge C_m$, the *number of literals* in clause C_i is denoted by $|C_i|$. We denote by $\dim(\Phi) = \max_{i=1,\dots,m} |C_i|$, where $|C_i|$ denotes the number of literals of C_i. We call Φ a *d-CNF* if $\dim(\Phi) \leq d$. The *number of clauses* and the *number of literals* appearing in Φ are denoted by $|\Phi|$ and $\|\Phi\|$, respectively. The *occurrence* of a variable in a CNF is the number of clauses involving that variable or its negation. Vectors are written using bold fonts throughout, e.g. $\boldsymbol{x}$. The problem asked in [8] is for d-CNF formulas where d is a fixed positive integer, but we also consider the same problem for general CNFs.

Generation of signatures ($GS(\Phi)$)
Input: A CNF Φ.
Output: All possible signatures of Φ.

Motivated by MAX-SAT and MIN-SAT, we also consider maximal and minimal signatures. A signature of a CNF Φ is called *maximal* (resp. *minimal*) if an inclusionwise maximal (resp. minimal) subset of the clauses takes value 1.

Our results We give a polynomial delay algorithm for the class of tractable CNF formulas in Section 2. Section 3 discusses CNFs with bounded dimension. For the class of formulas with bounded dimension and variable occurrences, we give an incremental polynomial algorithm in Section 3.1. In Section 3.2, we show that $GS(\Phi)$ can be solved in incremental polynomial time for formulas with a bounded dimension, thus answering the open problem posed by Kröll. The generation of maximal and minimal signatures is considered in Section 4. Finally, we conclude the paper in Section 5, where a 'reversed' variant of the problem is proposed as an open question.

2. Tractable CNFs

Let Φ and Ψ be two CNFs. If the clauses of Ψ are also clauses in Φ, then Ψ is called a *sub-CNF* of Φ, denoted by $\Psi \subseteq \Phi$. We call a family of CNFs *tractable* if for any CNF Φ in this family the satisfiability of any sub-CNF of Φ can be decided in polynomial time even after fixing any subset of the variables at arbitrary values. For example, the classes of 2-CNFs or Horn CNFs are tractable.

Theorem 1. *If Φ belongs to a tractable family and has m clauses, then its signatures can be generated with polynomial delay.*

[1] We prefer the term *signature* over the term *clause sequence* proposed by Kröll, since it is a binary string, not a sequence of clauses. Therefore we use the term *signature* in the rest of the paper.

Proof. The idea is to apply the so-called 'flashlight' approach in the signature space, using SAT as a 'flashlight' [9]. Let $\Phi = \bigwedge_{i=1}^{m} C_i$. We are going to build a binary tree in which the paths from the root to the vertices of the tree correspond to binary values of initial segments of the set of clauses, that is, $C_1, \ldots, C_k$ for some $1 \leq k \leq m$. The tree is built layer by layer, each new layer corresponding to a new clause being added to the examined prefix. There exists a signature with this prefix if and only if the CNF formed by the clauses set to value one in this sequence is satisfiable even after all the forced fixing of variables that appear in clauses whose value is zero (note that a clause has value 0 if and only if all the literals in it are 0). If such a CNF is not satisfiable, we backtrack and do not explore the subtree rooted at this vertex as there exists no signature with this prefix. If the CNF is satisfiable, we continue building the corresponding subtree which in this case is guaranteed to contain at least one signature. The algorithm will not backtrack above this vertex before outputting all (at least one) signatures in this subtree. It is not difficult to verify that after at most $2m$ calls to SAT we can output a new signature not generated before. After outputting the last signature, the procedure terminates after at most m SAT calls.

By the above, the signatures of Φ can be generated with a delay of $O(m)$ SAT-calls. As Φ belongs to a tractable family, this implies a polynomial delay algorithm, concluding the proof of the theorem. □

Remark 2. Let us remark that the family of monotone CNFs is tractable, but for this case there is a more efficient polynomial delay generation of the signatures. Indeed, in this case we can view a clause as a subset of the variables. Consequently, the set of zeros in a signature corresponds to a union of clauses. It is easy to see that such unions can be generated with $O(nm)$ delay and $O(n)$ average delay, where n is the number of variables, and $m = |\Phi|$ is the number of clauses, see [10, Proposition 11] and [11, Theorem 15].

Note that in this case Theorem 1 guarantees only an $O(\|\Phi\|m)$ delay, because every SAT call requires $O(\|\Phi\|)$ time.

3. CNFs with bounded dimension

3.1. Bounded variable occurrence

Given a CNF Φ, we denote by $H_\Phi = (\Phi, E)$ the *conflict graph* of Φ. The vertices of H_Φ are the clauses of Φ and edges are exactly the *conflicting* pairs of clauses, i.e., pairs (C_i, C_j) for which there exists a literal $u \in C_i$ such that $\bar{u} \in C_j$.

Let $S \subseteq \Phi$ be a maximal independent set of H_Φ, and let $L(S) = \bigcup_{C_i \in S} C_i$ denote the set of literals appearing in the clauses of S. We define a partial assignment $\boldsymbol{a_S} : L(S) \to \{0, 1\}$ by setting all literals of $L(S)$ to zero (and hence the complementary literals are set to 1). The *signature associated to* S is then defined as $\sigma_\Phi(S) := \sigma_\Phi(\boldsymbol{a_S}) = (y_1, \ldots, y_m) \in \{0, 1\}^m$. The coordinates of $\sigma_\Phi(S)$ are well-defined as $y_i = 0$ if and only if $C_i \in S$ for $i = 1, \ldots, m$. We will dismiss the subscript Φ whenever the CNF in question is clear from the context. Note that for different maximal independent sets $S \neq S'$ of H_Φ we have $\sigma(S) \neq \sigma(S')$. It is worth mentioning that all maximal independent sets of H_Φ can be generated with polynomial delay [12–14], which is hence a good start for CNF signature generation.

Assume that Φ has bounded dimension, i.e., for a constant d we have $|C_i| \leq d$ for all $i = 1, \ldots, m$. Let us define $\mathcal{X}_j = \{C_i \in \Phi \mid x_j \in C_i \text{ or } \bar{x}_j \in C_i\}$. We say that Φ is of *ω-bounded occurrence* if $|\mathcal{X}_j| \leq \omega$ for $j = 1, \ldots, n$ and ω is a fixed constant.

Theorem 3. *If Φ has bounded dimension and occurrence, then its signatures can be generated in incremental polynomial time.*

Proof. An *induced matching* of an undirected graph is a matching which forms an induced subgraph, that is, no two of its edges are joined by an edge of the graph. A maximal induced matching can be found by a simple greedy approach: repeatedly select an edge e, add it to the solution, and delete the end-vertices of the edge together with the set of vertices adjacent to at least one of them.

Let $M \subseteq E$ be a maximal induced matching in H_Φ. Let us denote by μ the number of edges in M and by N the 2μ vertices incident to edges in M. Note that H_Φ has at least 2^μ maximal independent sets (by choosing arbitrarily one of the endpoints from each edge of M and then greedily extending this set to a maximal independent set) and hence at least this many signatures can be generated with polynomial delay, as explained above. We denote by $W \subseteq \Phi$ the set of clauses that have edges in H_Φ connecting them to some of the clauses in N, formally

$$W = \{C \in \Phi \mid \exists C' \in N : (C, C') \in E\}$$

Note that $N \subseteq W$. Moreover, let us denote $U = \Phi \setminus W$. It is easy to see that U is an independent set in H_Φ, since any edge with both endpoints in U could be used to enlarge M, thus contradicting its maximality.

Assume that $|C_i| \leq d$ for all $i = 1, \ldots, m$, and $|\mathcal{X}_j| \leq \omega$ for all $j = 1, \ldots, n$, where d and ω are fixed constants. Observe that with these assumptions all clauses in N contain together at most $2\mu d$ literals and so we have $|W| \leq 2\mu d\omega$. Let K be the set of variables involved in clauses of W and let us denote $n' = |K|$. Now we have $n' \leq d|W|$, implying

$$n' \leq 2\mu d^2\omega.$$

Finally, let us denote by L the (possibly empty) set of variables that appear only in clauses of U. Clearly, all variables in L are monotone in Φ (some variables appear only positively while some others appear only negatively) since U is an independent set in H_Φ. Thus all literals in Φ that correspond to variables in L can be simultaneously assigned zero.

The procedure that generates all signatures of Φ works in three steps. In the first step, all literals in L are set to 0 and the resulting CNF in n' variables from K is denoted Φ'. Then we generate with polynomial delay the maximal independent sets S_ℓ, $\ell = 1, ..., k$ of $H_{\Phi'}$ (see [12–14]), and the corresponding signatures $\sigma(S_\ell)$, $\ell = 1, ..., k$ of Φ, where $k \geq 2^\mu$ (note that a signature of Φ' in this case extends uniquely to a signature of Φ by adding zeros for clauses that consist only of variables from L).

In the second step we generate all the remaining signatures stemming from assignments where all literals in L are set to zero. We try all $2^{n'}$ binary assignments to the variables in K and check for each of them whether a new signature was generated. Note that $k \geq 2^\mu \geq (2^{n'})^{1/2d^2\omega}$ using the inequality $n' \leq 2\mu d^2\omega$, which in turn implies $2^{n'} \leq k^{2d^2\omega}$. Hence this step can be done in an incremental polynomial time, in particular in $O(mnk^{2d^2\omega})$ time since generating a signature for a given assignment takes $O(mn)$ time.

For the third step let us assume that $k' \geq k$ distinct signatures were generated in the first and second step. By switching the assignment to literals in L, we may get new signatures, resulting from changing some of the zeros in a signature to one. For any partial assignment to the variables in K we obtain a monotone CNF on variables in L, and hence this is a set-union generation problem that can be solved with polynomial delay as observed in the previous section. Since it suffices to consider only those partial assignments to the variables in K which produced a signature in the first two steps, we may get in this way the same signature multiple times, but no more than k' times, and thus at this step the additional signatures are also generated in incremental polynomial time. □

3.2. Unbounded occurrence

In the previous section, we considered CNFs with bounded dimension and occurrence. The running time of the algorithm provided by Theorem 3 depends exponentially on ω, hence it is not suitable for handling the general case. In the present section, a more general procedure is given based on a different approach.

For a CNF Φ, we denote by $G_\Phi = (\Phi, E)$ the so called *dual graph* of Φ [15]. The vertices of G_Φ are the clauses of Φ and edges are exactly the pairs of clauses (C_i, C_j) for which there exists a variable that occurs in both C_i and C_j (complemented or not). If $S \subseteq \Phi$ is an independent set of G_Φ, then the clauses of S have pairwise disjoint sets of variables involved.

Theorem 4. *There exists an algorithm $\mathfrak{A}$ that generates the signatures of a CNF Φ consisting of m clauses in n binary variables in $O(dm^2nk^{\binom{d}{2}})$ total time, where $d = \dim(\Phi)$ and k is the number of signatures.*

Proof. We prove the claim by induction on d. For $d \leq 2$ the claim follows by Theorem 1.

Assume now that we already proved the claim for all $d' < d$, and let us consider a CNF $\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ with $\dim(\Phi) = d$. Let us associate to Φ its dual graph G_Φ as defined above. Let $S \subseteq \Phi$ be a maximal independent set of G_Φ. Such a set can be obtained by a simple greedy procedure in polynomial time in the size of Φ. Note that clauses in S involve pairwise disjoint sets of variables, due to the fact that S is an independent set of G_Φ. Thus, we can choose a literal $u_C \in C$ for each clause $C \in S$, set all other literals in C to zero, set all other variables not occurring in clauses of S to zero, and make all possible truth assignment to the literals u_C, $C \in S$. This way we obtain $k_0 = 2^{|S|}$ different binary signatures of Φ. Note that we can output these k_0 signatures with polynomial delay.

The total number of variables involved in clauses of S is $n' \leq d|S|$. Hence we can assign in all possible ways values to these variables, and produce $2^{n'} \leq k_0^d$ subproblems Φ_j, $j = 1, ..., 2^{n'}$ in the remaining variables in $O(mn2^{n'}) = O(mnk_0^d)$ time which is polynomial in the input size and k_0, since d is a fixed constant. Note that each of these subproblems is obtained from Φ by fixing the n' variables at a binary assignment, and such a substitution can be done in $O(mn)$ time. Note also, that each of these residual problems is of dimension at most $d - 1$. Indeed, each of the clauses not in S shares at least one variable with the clauses of S, since S is a maximal independent set of G_Φ, and now that shared variable is fixed at a binary value.

We apply algorithm $\mathfrak{A}$ to each of the residual sub-CNFs Φ_j, $j = 1, ..., 2^{n'}$, one by one. This way we produce signatures that extend the pattern on S defined by $x^j \in \{0,1\}^{n'}$, for all $j = 1, ..., 2^{n'}$ one by one. Two of these extended signatures may coincide, but only if they are extensions of two different x^j-s, since for a fixed x^j algorithm $\mathfrak{A}$ generates pairwise distinct extensions. Thus, we may produce the same signature no more than $2^{n'}$ times. Since $2^{n'} = O(k_0^d)$, we can show that this procedure works in total polynomial time.

To see this let us introduce some additional notation. We denote by $X_j \subseteq Y = \{0,1\}^{n'}$, $j = 1, ..., 2^{|S|}$ the nonempty sets of (partial) assignments that produce the same signature on the clauses of S. Note that the X_j-s partition the set Y of partial truth assignments. For $\boldsymbol{x} \in Y$, let us denote by $\Phi(\boldsymbol{x})$ the residual CNF, and by $k(\boldsymbol{x})$ the number of signatures of $\Phi(\boldsymbol{x})$. We denote by $g(\Psi)$ the running time of the above described recursive algorithm on CNF Ψ and let $G(m, n, d, k)$ be the maxima of $g(\Psi)$ over all CNFs with at most m clauses on n variables having $\dim(\Psi) \leq d$ and having at most k signatures.

The total computational time in the first phase of the above procedure that ends with producing a list of $2^{n'}$ residual CNFs, each of $\dim \leq d - 1$ is bounded by $O(m^2n) + O(mnk_0) + O(mnk_0^d) \leq Km^2nk_0^d$ for a suitable constant K that does not depend on m, n, and k_0. The first term on the left hand side is the time to build G_Φ and to find a maximal independent set S. The second term is the time we need to generate the k_0 initial signatures. The third term is the time to generate the $2^{n'} \leq k_0^d$ subproblems.

For $\boldsymbol{x} \in X_j$ and $\boldsymbol{x'} \in X_{j'}$ with $j \neq j'$ the CNFs $\Phi(\boldsymbol{x})$ and $\Phi(\boldsymbol{x'})$ cannot share signatures, since those must already differ on S by the definition of the sets X_j for $j = 1, ..., k_0$. However, for $\boldsymbol{x}, \boldsymbol{x'} \in X_j$ CNFs $\Phi(\boldsymbol{x})$ and $\Phi(\boldsymbol{x'})$ may share (some, even many) signatures. Discounting the one signature we already produced with the given 0-1 values on S, we can still expect k_j different signatures produced by algorithm $\mathfrak{A}$ when we use it for CNFs $\Phi(\boldsymbol{x})$, $\boldsymbol{x} \in X_j$, where $\max_{\boldsymbol{x} \in X_j}[k(\boldsymbol{x}) - 1] \leq k_j \leq \sum_{\boldsymbol{x} \in X_j}[k(\boldsymbol{x}) - 1]$. Thus, in total we get $k = k_0 + k_1 + \cdots + k_{2^{|S|}}$ different signatures for Φ. The total running time on CNFs $\Phi(\boldsymbol{x})$, $\boldsymbol{x} \in X_j$ can be bounded by

$$\sum_{\boldsymbol{x} \in X_j} g(\Phi(\boldsymbol{x})) \leq |X_j| G(m, n, d-1, k_j).$$

Thus, for the total running time of algorithm $\mathfrak{A}$ on Φ we get

$$\begin{aligned} g(\Phi) \leq G(m, n, d, k) &\leq Km^2 nk_0^d + \sum_{j=1}^{k_0} |X_j| G(m, n, d-1, k_j) \\ &\leq Km^2 nk_0^d + k_0^d G(m, n, d-1, k), \end{aligned}$$

where for the last inequality we used $k_j \leq k$ for all $j = 1, ..., k_0$, implying $G(m, n, d-1, k_j) \leq G(m, n, d-1, k)$, which allows this quantity to be factored out of the sum, that can be then upper bounded by $\sum_{j=1}^{k_0} |X_j| = 2^{n'} \leq k_0^d$. Using this we can show by induction on d that $G(m, n, d, k) \leq Ldm^2 nk^{\binom{d}{2}}$ for some constant L (we will choose $L \geq K$) which will complete the proof of our claim. Now

$$\begin{aligned} G(m, n, d, k) &\leq Km^2 nk_0^d + k_0^d G(m, n, d-1, k) \\ &\leq Km^2 nk_0^d + k_0^d L(d-1)m^2 nk^{\binom{d-1}{2}} \\ &\leq Lm^2 nk^d + k^d L(d-1)m^2 nk^{\binom{d-1}{2}} \\ &\leq Lm^2 nk^d + L(d-1)m^2 nk^{\binom{d-1}{2}+d} \leq Ldm^2 nk^{\binom{d}{2}}. \quad \square \end{aligned}$$

Remark 5. Since 2-CNFs are tractable, the running time of the algorithm can be slightly improved by stopping the recursion when $d = 2$, as pointed out by Strozecki [16].

Corollary 6. *The algorithm constructed in the above proof works in incremental polynomial time.*

Proof. Using the above theorem, we can prove this claim by induction on the dimension d. When $d = 1$, the claim is trivially true.

Consider now the general case, as in the proof of the above theorem. As we remarked there, producing the first $k_0 = 2^{|S|}$ signatures in fact can be done with polynomial delay. After this we start processing the CNFs $\Phi(\boldsymbol{x})$ for $\boldsymbol{x} \in X_j$, $j = 1, ..., k_0$. Note that the signatures produced from $\Phi(\boldsymbol{x})$, $\boldsymbol{x} \in X_j$ and $\Phi(\boldsymbol{x'})$, $\boldsymbol{x'} \in X_{j'}$ are all different if $j \neq j'$. Note also that $\dim(\Phi(\boldsymbol{x})) \leq d - 1$ for all $\boldsymbol{x} \in X_j$, $j = 1, ..., k_0$, and thus we can assume by induction that their signatures can be produced in incremental polynomial time in the size of $\Phi(\boldsymbol{x})$, which is bounded by the size of Φ. Thus, if $X_j = \{\boldsymbol{x_1}, ..., \boldsymbol{x_\ell}\}$, then we can produce $k(\boldsymbol{x_1})$ new signatures in incremental polynomial time, in fact regardless how many we produced previously (including the k_0 we have from the first phase). Let us denote by $q(m, n, k(\boldsymbol{x_1}))$ the polynomial bounding the total time processing $\Phi(\boldsymbol{x_1})$. If $k(\boldsymbol{x_2}) > k(\boldsymbol{x_1})$, then maybe the first $k(\boldsymbol{x_1})$ signatures produced from $\Phi(\boldsymbol{x_2})$ coincide with the ones we already generated from $\Phi(\boldsymbol{x_1})$, but still after at most $q(m, n, k(\boldsymbol{x_1}))$ time we get a new signature. In the worst case, we have $k_j = k(\boldsymbol{x_1}) \geq k(\boldsymbol{x_i})$ for all $\boldsymbol{x_i} \in X_j$, $i \neq 1$, in which case processing $\Phi(\boldsymbol{x_i})$, $i = 2, ..., \ell$ may not produce any new signatures. Since $\ell \leq k_0^d$, this means that the largest gap between the output of the last signature of $\Phi(\boldsymbol{x_1})$ and next new signature is not more than $k_0^d q(m, n, k(\boldsymbol{x_1}))$, at a moment when we have already produced $k' \geq k_0 + k(\boldsymbol{x_1})$ signatures. Thus this largest time gap between two outputs is still bounded by a polynomial of m, n, and the number of signatures $k' \geq k_0 + k(\boldsymbol{x_1})$ produced so far. As both n and m are bounded by the input size $\|\Phi\|$, the corollary follows. $\square$

4. Generating maximal and minimal signatures

Generation of maximal signatures is difficult as it includes SAT as a special case.

Theorem 7. *Unless P=NP, the maximal signatures cannot be generated in total polynomial time.*

Proof. Let us consider a CNF Φ, and observe that its unique maximal signature is the all-one vector if and only if Φ is satisfiable. Assume by contradiction that we have a total polynomial time algorithm for generating all maximal signatures,

and denote by $t(k, \ell)$ the polynomial bound for its termination when the input has size k and output involves exactly ℓ signatures. Let us run this algorithm for $t(||\Phi||, 1)$ time, which is polynomial in the input size for input Φ. If this algorithm outputs the all-one vector then Φ is satisfiable, and otherwise it is not. Hence it would decide the satisfiability of Φ in polynomial time. As SAT is NP-complete [17], the theorem follows. □

It turns out that minimal signatures can be generated efficiently.

Theorem 8. *Minimal signatures can be generated with polynomial delay for arbitrary CNF formulas.*

Proof. We claim that there is a one-to-one correspondence between minimal signatures of a CNF Φ and maximal independent sets of its conflict graph H_Φ. Since H_Φ can be built in polynomial time from Φ and maximal independent sets of a graph can be generated with polynomial delay [12–14], this would prove the theorem.

To see the above claim, assume first that a signature $\sigma = \{\sigma_C \mid C \in \Phi\}$ is a minimal signature of Φ. Note that the set $S = \{C \in \Phi \mid \sigma_C = 0\}$ is an independent set in H_Φ. For any $C \in \Phi$ with $\sigma_C = 1$ there must exist a conflict between C and some $C' \in S$, since otherwise we could set σ_C to zero without forcing any of the clauses in S to change their values, contradicting the minimality of σ. Thus S must be a maximal independent set.

The other direction follows from the fact that if S is a maximal independent set of H_Φ and we set all the clauses in S to zero, then all other clauses of Φ are forced to take value one due to the conflicts between S and other vertices of H_Φ. □

5. Conclusions

In this paper we show that all signatures of a given CNF with a bounded dimension can be generated in incremental polynomial time, answering an open problem posed by Kröll [8, Problem 4.7]. A faster incremental polynomial algorithm is provided for the class of formulas where both the dimension and the occurrence are bounded. Moreover, it is also shown that the same task can be done with polynomial delay if the input CNF is from a tractable class (in this case no bound on dimension or occurrence is necessary). Finally, it is proved that maximal signatures cannot be generated in total polynomial time unless $P = NP$, while minimal signatures can be generated with polynomial delay for arbitrary CNF formulas.

In this context it is interesting to note that given a 3-CNF Φ with m clauses and the vector $\boldsymbol{y} = (1, 1, ..., 1) \in \{0, 1\}^m$ it is NP-hard to test whether $\boldsymbol{y}$ is a signature of Φ, or not ($\boldsymbol{y}$ is a signature if and only if Φ is satisfiable). On the other hand, our results show that generating all signatures of Φ can be done in incremental polynomial time. This is a rather unusual behavior for a generation problem. Typically, if all solutions of a given problem can be generated in incremental polynomial time, checking if a given candidate is a solution or not is computationally easy.

An additional problem related to CNF signatures was stated at the Dagstuhl Seminar 19211 by Turán. Given a set $S \subseteq \{0, 1\}^m$, does there exist a CNF with m clauses such that S is exactly its set of all signatures? If yes, can such a CNF be computed efficiently? This 'reverse' problem (get the signatures, output clauses) to the problem presented in this paper (get the clauses, output signatures) is to the best of our knowledge completely open.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We are truly grateful for Yann Strozecki for his valuable observations and suggestions that helped us to improve the paper. We would also like to thank the anonymous referees for their careful reading of the manuscript and their insightful comments.

Kristóf Bérczi was supported by the János Bolyai Research Fellowship of the Hungarian Academy of Sciences and by the ÚNKP-19-4 New National Excellence Program of the Ministry for Innovation and Technology. Ondřej Čepek and Petr Kučera gratefully acknowledge a support by the Czech Science Foundation (Grant 19-19463S). Projects no. NKFI-128673 and "Application Domain Specific Highly Reliable IT Solutions" have been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the FK_18 and the Thematic Excellence Programme TKP2020-NKA-06 (National Challenges Subprogramme) funding schemes, respectively. This work was supported by the Research Institute for Mathematical Sciences, an International Joint Usage/Research Center located in Kyoto University.

References

[1] A. Letelier, J. Pérez, R. Pichler, S. Skritek, Static analysis and optimization of semantic web queries, ACM Trans. Database Syst. (TODS) 38 (4) (2013) 25.
[2] A.A. Bulatov, V. Dalmau, M. Grohe, D. Marx, Enumerating homomorphisms, J. Comput. Syst. Sci. 78 (2) (2012) 638–650.
[3] A. Durand, N. Schweikardt, L. Segoufin, Enumerating answers to first-order queries over databases of low degree, in: Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, ACM, 2014, pp. 121–131.

[4] W. Kazana, L. Segoufin, Enumeration of first-order queries on classes of structures with bounded expansion, in: Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, ACM, 2013, pp. 297–308.
[5] L. Segoufin, Enumerating with constant delay the answers to a query, in: Proceedings of the 16th International Conference on Database Theory, ACM, 2013, pp. 10–20.
[6] M. Kröll, R. Pichler, S. Skritek, On the complexity of enumerating the answers to well-designed pattern trees, in: 19th International Conference on Database Theory, ICDT 2016, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
[7] J.-L. Guigues, V. Duquenne, Familles minimales d'implications informatives résultant d'un tableau de données binaires, Math. Sci. Hum. 95 (1986) 5–18.
[8] E. Boros, B. Kimelfeld, R. Pichler, N. Schweikardt, Enumeration in data management (Dagstuhl seminar 19211), Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
[9] E. Boros, K. Elbassioni, V. Gurvich, Algorithms for Generating Minimal Blockers of Perfect Matchings in Bipartite Graphs and Related Problems, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 3221, 2004, pp. 122–133.
[10] Y. Strozecki, A. Mary, Efficient enumeration of solutions produced by closure operations, Discret. Math. Theor. Comput. Sci. 21 (3) (2019) 1–30.
[11] F. Capelli, Y. Strozecki, Enumerating models of DNF faster: breaking the dependency on the formula size, Discrete Appl. Math. (2020), https://doi.org/10.1016/j.dam.2020.02.014.
[12] S. Tsukiyama, M. Ide, H. Ariyoshi, I. Shirakawa, A new algorithm for generating all the maximal independent sets, SIAM J. Comput. 6 (3) (1977) 505–517.
[13] D.S. Johnson, M. Yannakakis, C.H. Papadimitriou, On generating all maximal independent sets, Inf. Process. Lett. 27 (3) (1988) 119–123.
[14] K. Makino, T. Uno, New algorithms for enumerating all maximal cliques, in: Scandinavian Workshop on Algorithm Theory, Springer, 2004, pp. 260–272.
[15] M. Samer, S. Szeider, Algorithms for propositional model counting, J. Discret. Algorithms 8 (1) (2010) 50–64, https://doi.org/10.1016/j.jda.2009.06.002.
[16] Y. Strozecki, personal communication, 2020.
[17] S.A. Cook, The complexity of theorem-proving procedures, in: Proceedings of the Third Annual ACM Symposium on Theory of Computing, 1971, pp. 151–158.