First-order query evaluation on structures of bounded degree

Wojciech Kazana, Luc Segoufin

FIRST-ORDER QUERY EVALUATION ON STRUCTURES OF BOUNDED DEGREE

WOJCIECH KAZANA AND LUC SEGOUFIN

INRIA and ENS Cachan
e-mail address: kazana@lsv.ens-cachan.fr

INRIA and ENS Cachan
e-mail address: see `http://pages.saclay.inria.fr/luc.segoufin/`

ABSTRACT. We consider the enumeration problem of first-order queries over structures of bounded degree. In [3] it was shown that this problem is in CONSTANT-DELAY$_{lin}$. An enumeration problem belongs to CONSTANT-DELAY$_{lin}$ if for an input of size n it can be solved by:
- an $O(n)$ precomputation phase building an index structure,
- followed by a phase enumerating the answers with no repetition and a constant delay between two consecutive outputs.

In this article we give a different proof of this result based on Gaifman's locality theorem for first-order logic. Moreover, the constants we obtain yield a total evaluation time that is triply exponential in the size of the input formula, matching the complexity of the best known evaluation algorithms.

1. INTRODUCTION.

Model checking is the problem of testing whether a given sentence is true in a given model. It's a classical problem in many areas of computer science, in particular in verification. If the formula is no longer a sentence but has free variables then we are faced with the query evaluation problem. In this case the goal is to compute all the answers of a given query on a given database.

As for model checking, query evaluation is a problem often requiring a time at least exponential in the size of the query. Even worse, the evaluation often requires a time of the

form $n^{O(k)}$, where n is the size of the database and k the size of the query. This is dramatic, even for small k, when the database is huge.

However there are restrictions on the structures that make things easier. For instance MSO sentences can be tested in time linear in n over structures of bounded tree-width [2] and MSO queries can be evaluated in time linear in $n + m$, where m is the size of the output of the query (note that m could be exponential in the number of free variables of the query, and hence in k) [4].

In this paper we are concerned with first-order logic (FO) and structures of bounded degree. In this case the model checking problem for FO sentences is known to be linear in n [9]. Moreover, the constant factor is at most triply exponential in the size k of the formula [5]. This last algorithm easily extends to query evaluation obtaining an algorithm working in time $f(k)(n + m)$ where f is a triply exponential function.

As we already mentioned, the size m of the output may be exponential in the arity of the formula and therefore may still be large. In many applications enumerating all the answers may already consume too many of the allowed resources. In this case it may be appropriate to first output a small subset of the answers and then, on demand, output a subsequent small number of answers and so on until all possible answers have been exhausted. To make this even more attractive it is preferable to be able to minimize the time necessary to output the first answers and, from a given set of answers, also minimize the time necessary to output the next set of answers - this second time interval is known as the *delay*.

We say that a query can be evaluated in linear time and constant delay if there exists an algorithm consisting of a preprocessing phase taking time linear in n which is then followed by an output phase printing the answers one by one, with no repetition and with a constant delay between each output. Notice that if a linear time and constant delay algorithm exists then the time needed for the total query evaluation problem is bounded by $f(k)(n + m)$ for some function f. Hence this is indeed a restriction of the linear time query evaluation algorithms mentioned above. From the best of our knowledge it is not yet known whether a bound $f(k)(n + m)$ for some function f on a query evaluation problem implies the existence of a linear time and constant delay enumeration algorithm. We conjecture this is not the case.

It was shown in [3] that linear time constant delay query evaluation algorithms could be obtained for FO queries over structures of bounded degree, hence improving the results of [9] and [5].

The proof of [3] is based on an intricate quantifier elimination method. In this paper we provide a different proof of this result based on Gaifman Locality of FO queries. Our algorithm can be seen as an extension of the algorithm of [5] to queries. However the index structure built during the preprocessing phase is more complicated than the one of [5] in order to obtain the constant delay enumeration. Moreover, our constant factor is triply exponential in the size of the formula, while it is not clear whether the constant factor obtained in [3] is elementary. Note that the triply exponential constant factor cannot be significantly improved: it is shown in [5] that a constant factor only doubly exponential in the size of the formula is not possible unless the parametrized complexity class AW[*] collapses to the parametrized class FPT.

2. Definitions.

2.1. **Gaifman locality and first-order logic.** A relational signature is a tuple $\sigma = (R_1, \ldots, R_l)$, each R_i being a relation symbol of arity r_i. A relational structure over σ is a tuple $\mathcal{A} = (A, R_1^{\mathcal{A}}, \ldots, R_l^{\mathcal{A}})$, where $A = \{a_1, \ldots, a_m\}$ is the set of elements of $\mathcal{A}$ and $R_i^{\mathcal{A}}$ is a subset of A^{r_i}. We fix a reasonable encoding of structures by words over some finite alphabet. The *size* of $\mathcal{A}$ is denoted by $||\mathcal{A}||$ and is the length of the encoding of $\mathcal{A}$.

The *Gaifman graph* of a relational structure $\mathcal{A}$, denoted by $G(\mathcal{A})$, is defined as follows: the set of vertices of $G(\mathcal{A})$ is A and there is an edge (a, b) in $G(\mathcal{A})$ iff there exists a relation R_i and a tuple $t \in R_i$ such that both a and b occur in t. Given $a, b \in A$, the *distance* between a and b, denoted $\delta(a, b)$, is the length of a shortest path between a and b in $G(\mathcal{A})$ or ∞ if a and b are not connected. The *distance* between two tuples $\bar{a} = (a_1, \ldots, a_k)$ and $\bar{b} = (b_1, \ldots, b_l)$ of $\mathcal{A}$, denoted $\delta(\bar{a}, \bar{b})$, is the $\min\{\delta(a_i, b_j) : 1 \le i \le k, 1 \le j \le l\}$. For a given $r \in \mathbb{N}$ and a given tuple of elements $\bar{a}$ of some structure $\mathcal{A}$, we denote by $N_r(\bar{a})$ the set of all elements in A such that their distance from $\bar{a}$ is less or equal to r. The *r-neighborhood* of $\bar{a}$, denoted as $\mathcal{N}_r(\bar{a})$, is the substructure of $\mathcal{A}$ induced by $N_r(\bar{a})$ and expanded with one constant for each element of $\bar{a}$. Given two tuples of elements $\bar{a}$ and $\bar{b}$ we say that they have *the same r-neighborhood type*, written $\mathcal{N}_r(\bar{a}) \simeq \mathcal{N}_r(\bar{b})$, if there is an isomorphism between $\mathcal{N}_r(\bar{a})$ and $\mathcal{N}_r(\bar{b})$.

We consider first-order logic (FO) built from atomic formulas of the form $x = y$ or $R_i(x_1, \ldots, x_{r_i})$ for some relation R_i and closed under the usual Boolean connectives ($\neg, \vee, \wedge$) and existential and universal quantifications ($\exists, \forall$). When writing $\phi(\bar{x})$ we always mean that $\bar{x}$ are exactly the free variables of ϕ. Given a structure $\mathcal{A}$ and a tuple $\bar{a}$ of elements of $\mathcal{A}$, we write $\mathcal{A} \models \phi(\bar{a})$ if the formula ϕ is true in $\mathcal{A}$ after replacing its free variables with $\bar{a}$. As usual $|\phi|$ denotes the size of ϕ.

We are now ready to state Gaifman locality for FO.

Theorem 2.1 (Gaifman Locality Theorem [7])**.** *For any first-order formula $\phi(\bar{x})$, for every structure $\mathcal{A}$ and tuples $\bar{a}$, $\bar{b}$, we have $\mathcal{N}_r(\bar{a}) \simeq \mathcal{N}_r(\bar{b})$ implies $\mathcal{A} \models \phi(\bar{a})$ iff $\mathcal{A} \models \phi(\bar{b})$, where $r = 2^{|\phi|}$.*

Given $d \in \mathbb{N}$, a structure is said to be *d-degree-bounded*, if the degree of the Gaifman graph is bounded by d. The following nice algorithmic property of d-degree-bounded structures can be proved using Theorem 2.1.

Theorem 2.2 ([9, 5])**.** *Fix $d \in \mathbb{N}$. The problem of whether a given d-degree-bounded structure $\mathcal{A}$ satisfies a given first-order sentence ϕ is decidable in time $2^{2^{2^{|\phi|^{O(1)}}}} ||\mathcal{A}||$.*

2.2. **Model of computation and Constant-Delay$_{lin}$ class.** We use Random Access Machines (RAM) with addition and uniform cost measure a model of computation. For further details on this model and its use in logic see [3].

An enumeration problem is a binary relation. Given an enumeration problem R and an input x, a *solution for x* is a y such that $(x, y) \in R$. An enumeration problem R induces a computational problem as follows: Given an input x, output all its solutions. An enumeration problem is in the class Constant-Delay$_{lin}$ if on input x it can be decomposed into two steps:

- a precomputation phase that is performed in time $O(|x|)$,

- an enumeration phase that outputs all the solutions for x with no repetition and a constant delay between two consecutive outputs. The enumeration phase has full access to the output of the precomputation phase but can use only a constant total amount of extra memory.

In particular if R is in CONSTANT-DELAY$_{lin}$ then the enumeration problem R can be solved in time $O(|x| + |\{y : R(x,y)\}|)$. From the best of our knowledge it is not known whether the converse is true or not. We conjecture that it is not. More details about CONSTANT-DELAY$_{lin}$ can be found in [3].

We are interested in the following enumeration problem for $\phi(\bar{x}) \in$ FO and $d \in \mathbb{N}$:

$$\text{Enum}_d(\phi) = \{(x,y)\ :\ x \text{ is a } d\text{-degree-bounded structure } \mathcal{A}, y \text{ is a tuple } \bar{a} \text{ of elements of } \mathcal{A}$$
$$\text{and } \mathcal{A} \models \phi(\bar{a})\}$$

We further denote by $\phi(\mathcal{A})$ the set $\{\bar{a}\ :\ \mathcal{A} \models \phi(\bar{a})\}$ and by $|\phi(\mathcal{A})|$ the cardinality of this set. We show that $\text{Enum}_d(\phi)$ is in CONSTANT-DELAY$_{lin}$.

Theorem 2.3 ([3])**.** *There is an algorithm that for all $d \in \mathbb{N}$, all $\phi \in$ FO and all d-degree-bounded structures $\mathcal{A}$ enumerates $\phi(\mathcal{A})$ with a precomputation phase taking time $2^{2^{2^{|\phi|^{O(1)}}}} \cdot ||\mathcal{A}||$ and a delay during the enumeration phase that is triply exponential in $|\phi|$. In particular, for all $d \in \mathbb{N}$ and all $\phi \in$ FO the enumeration problem $\text{Enum}_d(\phi)$ is in CONSTANT-DELAY$_{lin}$. Moreover, if the domain of $\mathcal{A}$ is linearly ordered, the algorithm enumerates $\phi(\mathcal{A})$ in increasing order relative to the induced lexicographical order on tuples.*

Hence the total query evaluation induced by the enumeration procedure of Theorem 2.3 is in time $2^{2^{2^{|\phi|^{O(1)}}}}(||\mathcal{A}|| + |\phi(A)|)$ thus matching the model checking complexity of Theorem 2.2. Our proof of Theorem 2.3 is based on Gaifman Locality Theorem while the proof of [3] uses a quantifier elimination procedure (see also [8] for a similar argument). Note that it is not clear from the proof of [3] that their algorithm is triply exponential in the size of the formula.

3. FO QUERY EVALUATION.

In this section we assume $d \in \mathbb{N}$ to be fixed and all our structures are d-degree bounded.

A formula $\phi(\bar{x})$ with k free variables $\bar{x} = x_1 \ldots x_k$ is said to be *connected around x_1* if $\phi(\bar{x})$ logically implies that $x_2, \ldots, x_k$ are in the (rk)-neighborhood of x_1 for $r = 2^{|\phi|}$.

Let $\mathcal{T}_{rk}$ be the set of all isomorphism types of (rk)-neighborhoods of single elements, i.e. the isomorphism types of structures of the form $\mathcal{N}_{rk}(a)$ for some element a of some structure $\mathcal{A}$. By (rk)-*neighborhood-type* of an element a we mean the isomorphism type of its (rk)-neighborhood. Because our structures are d-degree-bounded each (rk)-neighborhood has at most d^{rk} elements. For each $\tau \in \mathcal{T}_{rk}$ we denote by $\mu_\tau(x)$ the fact that the (rk)-neighborhood-type of x is τ. For each type in $\mathcal{T}_{rk}$ we fix a representative for the corresponding (rk)-neighborhood and fix a linear order among its elements. This way, we can speak of the first, second,..., element of an (rk)-neighborhood. For technical reasons, we actually fix a linear order for each l-neighborhood for $l \leq rk$ such that (i) it is compatible with the distance from the center of the neighborhood: the center is first, then come all the elements at distance 1, then all elements at distance 2 and so on... and (ii) the order of a $(l+1)$-type is consistent with the order on the induced l-type.

For some sequence $F = \{\alpha_2, \ldots, \alpha_m\}$ of $(m-1)$ elements from $[1, \ldots, d^{rk}]$, we write $\bar{x} = F(x_1)$ for the fact that, for $j \in \{2, \ldots, m\}$, x_j is the α_j-th element of the (rk)-neighborhood of x_1. Let $\mathcal{F}_{rk}^m$ be the set of all possible such F. Let $\mathcal{F}_{rk} = \bigcup_{1 \le m \le k} \mathcal{F}_{rk}^m$.

For a given $\bar{x} = x_1 \ldots x_k$ a r-*partition* of $\bar{x}$ is a set of pairs $\{(C_1, F_1), \ldots, (C_m, F_m)\}$ such that $\emptyset \ne C_i \subseteq \bar{x}$, $\bigcup_{1 \le i \le m} C_i = \{x_1, \ldots, x_k\}$, $C_i \cap C_j = \emptyset$ for $i \ne j$, and $F_i \in \mathcal{F}_{rk}^{|C_i|}$. For a given r-partition C of $\bar{x}$ and $(C_i, F_i) \in C$ we write $\bar{x}^i$ to represent variables from C_i, x_1^i to represent the first variable from C_i, x_2^i to represent second variable and so on.

For a given r-partition $C = \{(C_1, F_1), \ldots, (C_m, F_m)\}$ of $\bar{x}$ by $Div_r^C(\bar{x})$ we mean a conjunction of formulas saying that $N_r(\bar{x}^i) \cap N_r(\bar{x}^j) = \emptyset$ for all $1 \le i \ne j \le m$ and formulas $\bigwedge_{(C_i, F_i) \in C} \bar{x}^i = F_i(x_1^i)$. Note that the latter part implies that $\bar{x}^i$ is connected around x_1^i.

The following is an immediate consequence of Theorem 2.1.

Lemma 3.1. *Fix a structure $\mathcal{A}$. Then any formula $\phi(\bar{x})$ with k free variables is equivalent over $\mathcal{A}$ to a formula of the form*

$$\bigvee_{C \in C_r(\bar{x})} \left[Div_r^C(\bar{x}) \wedge \bigvee_{(\tau_1, \ldots, \tau_{|C|}) \in S_C} \bigwedge_{i \le |C|} \mu_{\tau_i}(x_1^i) \right] \tag{3.1}$$

where $r = 2^{|\phi|}$, $C_r(\bar{x})$ is the set of all r-partitions of $\bar{x}$, and $S_C \subseteq (\mathcal{T}_{rk})^{|C|}$ is finite.

Proof. Let $\phi(\bar{x})$ be a formula with k free variables and $r = 2^{|\phi|}$. As in the statement of this lemma, we denote by $C_r(\bar{x})$ the set of all partitions $C = \{(C_1, F_1), \ldots, (C_m, F_m)\}$ of $\bar{x}$ with $C_i = \left\{ x_1^i, \ldots, x_{|C_i|}^i \right\}$.

By taking all possible r-partitions over $\bar{x}$ we see that $\phi(\bar{x})$ is equivalent to:

$$\bigvee_{C \in C_r(\bar{x})} \left(\mathrm{Div}_r^C(\bar{x}) \wedge \phi(\bar{x}) \right)$$

Let $\bar{a}$ be a tuple of $\mathcal{A}$ such that $\mathcal{A} \models \phi(\bar{a})$. Thus there exists $C \in C_r(\bar{x})$, such that $\mathcal{A} \models Div_r^C(\bar{a}) \wedge \phi(\bar{a})$. As Div_r^C induces that variables from each C_i for some $(C_i, F_i) \in C$ are connected, the r-neighborhood of each $\bar{a}^i$ is completely included into the (rk)-neighborhood of a_1^i. Let $m = |C|$. For $1 \le i \le m$ let τ_i be the rk-neighborhood-type of a_1^i. We now take S_C as the set of all such tuples $(\tau_1, \ldots, \tau_m)$ for all tuples $\bar{a}$ and all r-partitions C such that $\mathcal{A} \models Div_r^C(\bar{a}) \wedge \phi(\bar{a})$. By construction we have $\phi(\bar{x})$ implies (3.1). The reverse inclusion is an immediate consequence of Gaifman Locality Theorem: When $Div_r^C(\bar{a})$ holds, $\mathcal{N}_r(\bar{a}^i)$ is induced by $\mathcal{N}_{rk}(a_1^i) = \tau_i$ and F_i. Moreover, $\mathcal{N}_r(\bar{a})$ is the disjoint union of $\mathcal{N}_r(\bar{a}^i)$ and is therefore induced by C. $\square$

We are now ready to prove Theorem 2.3.

Proof of Theorem 2.3. Fix a formula $\phi(\bar{x})$ with k free variables. Let $\mathcal{A}$ be a structure. Let $r = 2^{|\phi|}$. By Lemma 3.1, $\phi(\bar{x})$ is equivalent over $\mathcal{A}$ to a formula of the form given by (3.1). We assume that $\mathcal{A}$ comes with a linear order over its elements. If not, we use the linear order induced by the encoding of $\mathcal{A}$.

Intuitively the precomputation phase determines the disjunction given by (3.1) and precomputes the (rk)-neighborhoods of each element of $\mathcal{A}$. The fact that this can be done in time linear in $||\mathcal{A}||$ and triply exponential in $|\phi|$ will make use of Theorem 2.2.

In a first step, for each $i \leq rk$ we precompute the pairs of nodes at distance i. In other words, for each a in $\mathcal{A}$, we compute the set of elements b such that $\delta(a,b) = i$. This can easily be done in time linear in $rk \cdot ||\mathcal{A}||$ by induction on i: during the base case we compute the Gaifman graph of $\mathcal{A}$ and then we perform the classical computation of the transitive closure of this graph up to depth rk.

In a second step, the precomputation phase computes for each element a of $\mathcal{A}$ its (rk)-neighborhood: for each element a of $\mathcal{A}$, we compute its (rk)-neighborhood-type and for all $i \leq d^{rk}$ a pointer from a to the i-th element of its (rk)-neighborhood. We use an induction on the radius of the neighborhood to achieve this goal within the desired time constraints.

As 0-neighborhoods all share the same isomorphism type and have just one pointer to their centers, the induction base is obvious. So let's assume that in linear time in the size of $\mathcal{A}$ we have computed all l-neighborhoods for all nodes. With one more linear pass we now compute the $(l+1)$-neighborhoods. Fix $a \in \mathcal{A}$. From the first step, we have all the elements of $\mathcal{A}$ at distance $l+1$ from a. As we already have computed the l-neighborhood, it remains to try all possible orders among those elements and test isomorphism with the ordered types we have initially fixed.

There are at most d^{l+1} nodes at distance $l+1$ and $l < rk$. Hence the number of orders we need to test is bounded by $(d^{rk})!$. Once the order is fixed we try all possible (rk)-neighborhood-types that we have initially fixed (there are $|\mathcal{T}_{rk}|$ possibilities) and then test that the two orders induce an isomorphism (each test simply requires going through all tuples of the neighborhood). Let $s(r,k,d)$ be the maximal size of a (rk)-neighborhood. Thus this step is altogether achieved in time $O((d^{rk})! \cdot |\mathcal{T}_{rk}| \cdot s(r,k,d))$ which is triply exponential in $|\phi|$ because $r = 2^{|\phi|}$, $|\mathcal{T}_{rk}| = O(2^{s(r,k,d)})$ and $s(r,k,d) = O(d^{rk|\sigma|})$.

During the third step of the precomputation we determine the (rk)-neighborhood-types that are relevant for ϕ over $\mathcal{A}$. Fix a r-partition $C = \{(C_1, F_1), \ldots, (C_m, F_m)\}$ of $C_r(\bar{x})$ and a sequence $\tau_1, \ldots, \tau_m \in \mathcal{T}_{rk}$. This sequence is *relevant for C* if $\mathcal{A} \models \exists \bar{x} \left[\mathrm{Div}_r^C(\bar{x}) \wedge \bigwedge_j \mu_{\tau_j}(x_1^j) \right] \wedge \phi(\bar{x})$. Notice that the tests of the form $\mu_{\tau_j}(x_1^j)$ have been precomputed during the second step and can therefore now be treated as unary symbols. Similarly the tests $\mathrm{Div}_r^C(\bar{x})$ can be expressed using the graph computed during the first phase. Altogether, the first and second phase has replaced $\left[\mathrm{Div}_r^C(\bar{x}) \wedge \bigwedge_j \mu_{\tau_j}(x_1^j) \right]$ with a formula of size polynomial in k. Hence we can apply Theorem 2.2 in order to test whether the sequence is relevant for C in time linear in $||\mathcal{A}||$ and triply exponential in the size of the formula. We do this for all possible C, investigating at most $(|\mathcal{T}_{rk}|)^k = 2^{2^{2^{O(|\phi|)}}}$ cases. The number of possible C is the number of possible splits of k variables into disjoint and nonempty subsets multiplied by $(|\mathcal{F}_{rk}|)^k$, which altogether is again $2^{2^{2^{O(|\phi|)}}}$. For each C we store a list of all sequences relevant for it. We call a r-partition C *relevant* if that list is nonempty.

The fourth step of the precomputation phase orders, for each $\tau \in \mathcal{T}_{rk}$, the elements of $\mathcal{A}$ having that particular (rk)-neighborhood-type and stores a pointer from one element to the next one according to the linear order on the elements of $\mathcal{A}$. To do that, we just need to enumerate through all the elements in $\mathcal{A}$, in the order provided by the linear order on its elements, and, using information obtained in the second step, add each of them to a proper list. In order to do this we need to be able to sort a set of elements in linear time and this can be done in our RAM model as explained in [6].

The rest of the proof is an induction on the number k of free variables in ϕ.

Fix relevant r-partition $C = \{(C_1, F_1), \ldots, (C_m, F_m)\}$ in $C_r(\bar{x})$. We show how to enumerate in lexicographical order, with no repetition, constant memory and constant delay, all the tuples $\bar{a}$ such that $\mathcal{A}, \bar{a} \models \text{Div}_r^C(\bar{x}) \wedge \bigvee_i \bigwedge_j \mu_{\tau_{ij}}(x_1^{ij})$. The result will then follow from the following simple lemma, whose proof consist in merging two ordered lists.

Lemma 3.2 ([1]). *If there is a linear order $<$ such that R, R' are in* CONSTANT-DELAY$_{lin}$ *and both output their answers in increasing order relative to $<$, then $R \cup R'$ is also in* CONSTANT-DELAY$_{lin}$ *and the answers can be enumerated in increasing order relative to $<$.* $\square$

The base case $k = 1$ is trivial: since $k = 1$, it must be the case that also $m = 1$ and recall that the fourth step of the precomputation phase already computed the desired list of all elements of type τ_1. Assume now that we have the desired enumeration procedure for formulas with less than k free variables and we want to extend it to the ones with k free variables.

Without loss of generality assume that the most significant variable of $\bar{x}$ is in the first variable of $\bar{x}^1$, that the most significant variable of $\bar{x} \setminus \bar{x}^1$ is the first variable of $\bar{x}^2$ and so on. Let $\bar{x}' = C_1 \cup \ldots \cup C_{m-1}$ and $\bar{x}'' = C_m$. Consider query $\psi(\bar{x}') = \exists \bar{x}'' \text{Div}_r^C(\bar{x}) \wedge \bigvee_i \bigwedge_j \mu_{\tau_{ij}}(x_1^{ij})$. It has less free variables than ϕ and so the inductive hypothesis holds for it.

The last step of the precomputation phase performs the precomputation phase as given by the inductive hypothesis for ψ.

Altogether we have a precomputation phase of the desired properties: it works in time linear in $|\mathcal{A}|$ and triply exponential in $|\phi|$. We now turn to the enumeration phase.

We simultaneously do the following for each sequence $\tau_1, \ldots, \tau_m$ relevant for C and use Lemma 3.2 to avoid duplicate answers.

Fix $\tau_1, \ldots, \tau_m$ relevant for C. We enumerate, by induction, the solutions for ψ and recall that ψ implies $\text{Div}_r^{C'}(\bar{x}) \wedge \bigvee_i \bigwedge_j \mu_{\tau_{ij}}(x_1^{ij})$, where C' is C with (C_m, F_m) removed. For each solution $\bar{b}$ obtained by induction, using the precomputed pointers, we nest the enumeration of all elements a_m of $\mathcal{A}$ whose (rk)-neighborhood-type is τ_m. For each such element let $\bar{a}^m = F_m(a_m)$. Given a_m whose (rk)-neighborhood-type is τ_m, we check whether $N_r(\bar{a}^m)$ intersects with $N_r(\bar{b})$ or not (recall that this information has been precomputed during the first phase and therefore requires only constant time). If it does not, we have a solution $\bar{b}, \bar{a}^m$ for ϕ because of (3.1). If it does then we move to the next element whose (rk)-neighborhood-type is τ_m. Notice that the size of $N_r(\bar{b})$ is bounded by kd^r and $\bar{a}^m$ is r-connected around a_1 (thus included in $N_{rk}(a_m)$), hence the length of false hits is bounded by $kd^{r(k+1)}$. From the definition of ψ, for each $\bar{b}$ we are certain to find at least one matching $\bar{a}^m$ that gives us a solution $\bar{b}, \bar{a}^m$ to ϕ. Altogether we get the desired constant delay for the enumeration process.

The enumeration phase needs to process all possible r-partitions C and all relevant sequences of $\mathcal{T}_{rk}$, i.e. a number of cases triply exponential in $|\phi|$. For a fixed sequence $\tau_1, \ldots, \tau_m$ relevant for C the output is clearly returned in lexicographical order and Lemma 3.2 justifies that it is also the case for the enumeration procedure for ϕ. Altogether this yields a procedure linear in the size of the output and triply exponential in $|\phi|$. $\square$

4. CONCLUSION

We have given a new proof of the linear time and constant delay enumeration problem of first-order queries over structures of bounded degree. Our procedure is based on Gaifman's

locality theorem for first-order logic and our constants are triply exponential in the size of the query, and therefore induces the known complexity of the associated model checking problem.

Acknowledgement

References

[1] Guillaume Bagan. *Algorithmes et complexité des problèmes d'énumération pour l'évaluation de requêtes logiques*. PhD thesis, Université de Caen, 2009.

[2] Bruno Courcelle. Graph Rewriting: An Algebraic and Logic Approach. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 193–242. 1990.

[3] Arnaud Durand and Etienne Grandjean. First-order queries on structures of bounded degree are computable with constant delay. *ACM Trans. on Computational Logic (ToCL)*, 8(4), 2007.

[4] Jörg Flum, Markus Frick, and Martin Grohe. Query evaluation via tree-decompositions. *J. of the ACM*, 49(6):716–752, 2002.

[5] Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Logic*, 130(1-3):3–31, 2004.

[6] Etienne Grandjean. Sorting, Linear Time and the Satisfiability Problem. *Annals of Mathematics and Artificial Intelligence*, 16:183–236, 1996.

[7] Leonid Libkin. *Elements of Finite Model Theory*. Springer, 2004.

[8] Steven Lindell. A Normal Form for First-Order Logic over Doubly-Linked Data Structures. *Int. J. Found. Comput. Sci.*, 19(1):205–217, 2008.

[9] Detlef Seese. Linear Time Computable Problems and First-Order Descriptions. *Mathematical Structures in Computer Science*, 6(6):505–526, 1996.