On Faster Model Checking of First-Order Graph Properties and Tarski's Calculus of Relations

Yo: Tentative

Yoshiki Nakamura
Science Tokyo, Japan

Yuya Uezato
CyberAgent, Inc., Japan

Abstract

This is a SUPER-STRONG-PAPER and brings a NEW ERA to the field of FO.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases First-order Logic,Fine-grained Complexity

Digital Object Identifier 10.4230/LIPIcs...

1 Notation

（メモ: LIPICSとpLaTeXは異常に組み合わせが悪いので、 コメントなどで日本語を書く時はこのようにします。）

Cute round-corner box
コメントを付ける時には、screenで囲うとこういう感じになります

Yo: コメントテスト

Yo: コメントテスト 2

We write $\mathbb{FO}^k$ to denote the set of formulas of First-order predicate logic with k-variables.

Important Remark (Unary and Binary Predicates)

We allow unary and binary predicates in our $\mathbb{FO}^3$. Therefore, we do not allow terms involving ternary predicates such as $\forall x. \exists y. \forall z. P(x, y, z)$.[1]

On this remark, we can identify each model $\mathcal{M}$ as color graphs as follows:

- Each element corresponds a node (vertex).
- Each unary predicate U_i means "color-i" node.
- Each binary predicate P_j is "color-j" edge.
- For example, on two nodes x, y, a term $(U_i(x) \wedge U_j(y) \wedge P_k(x, y))$ means that x has the color i, y has the color j, and there is a color k edge from x to y.

[1] For each unary predicate U, we can simulate it by a diagonal binary predicate $B_U(x, y) := x = y \wedge U(x)$. However, in our later construction, we use some unary predicates; thus, we also allow unary predicates explicitly.

Quantifier width

Quantifier widthという謎の指標を導入します。
与えられた式の部分項 $\forall/\exists x.(\cdots)$ について $\cdots$ 部分に「直接」出てくるquantifierの個数を、この部分項のquantifier widthと呼ぶことにします。
たとえば、次の部分項

$$\forall x.\Big(\big(\underline{\forall} y.P(y)\big) \vee \big(\underline{\forall} z.(\underline{\forall} x.P(z,x)) \wedge Q(z)\big)\Big)$$

については、内部に3つのquantifierの出現（下線）があるのですが、直接の出現は赤くした2つだけなので、この部分項のquantifier widthは「２」です。
そして、式のquantifier widthというのは、最大のquantifier widthを持つ部分項における、その数として定義します。

2 Introduction

The *model checking problem* is the following problem:

given both a sentence φ and a structure $\mathfrak{A}$, does $\mathfrak{A} \models \varphi$ hold?

The model checking problem for $\mathbb{FO}$ is complete in PSPACE, unfortunately. Nevertheless, the model checking problem for $\mathbb{FO}^k$ can be decided in $\mathcal{O}(\|\varphi\| \cdot n^k)$ time [4, Proposition 3.1] by a naive bottom-up evaluation algorithm, where $\|\varphi\|$ is the length of φ and n is the number of vertices in $\mathfrak{A}$.

A natural question is, whether the complexity $\mathcal{O}(\|\varphi\| \cdot n^k)$ time can be improved. Williams [5] gave a positive answer to the question. He showed that the complexity can be improved to $\mathcal{O}(n^\omega)$ time for $\mathbb{FO}^3$,[2] where ω is the exponent of matrix multiplication.

Namely, the problem

Yo: On data complexity (formula is fixed and structure is given) [5]
- For $\mathbb{FO}^3$, $\mathcal{O}(n^\omega)$ time,
- For $\mathbb{FO}^k$ (where $k \geq 3$), $\mathcal{O}(n^{\omega+k-3})$ time,
- For $\mathbb{FO}^k$ (where $k \geq 9$), $\mathcal{O}(n^{k-4})$ time.

On combined complexity ()
Parameterized complexity

We shed a light the

Related work:

- [1]: Space complexity for $\mathbb{FO}^k$.
-

3 Preliminaries

We write $\models$

[2] More precisely, [2, Theorem 7]

4 from

[5]

5 Conclusion

6 FPT on FO3

We write $\mathbb{FO}^3_{k,w}$ to denote the set of formulas of $\mathbb{FO}^3$ with k-predicates and w-quantifier-width.

▶ **Theorem 1** ($\mathbb{FO}^3_{k,w}$ model checkig is FPT).
Let $\varphi : \mathbb{FO}^3_{k,w}$.
For a given model (graph) $\mathcal{G}$, we can decide if $\mathcal{G} \models \varphi$ in $O(2^{k+w} \cdot |\varphi| \cdot N^{\omega})$ where

- *N is the number of vertices in $\mathcal{G}$; and*
- *ω is a complexity constant for the boolean matrix multiplication (BMM): i.e., $O(N^{\omega})$-time is the time complexity for BMM of two (N, N) matrices.*[3]

▶ **Corollary 2.** *If we fix the number of predicates k and width w, the model checking problem $\mathcal{G} \models \varphi$ in $O(|\varphi| \cdot N^{\omega})$ time-complexity where N is the number of nodes in $\mathcal{G}$.*

kをfixしない、素朴な $\{(x, y, z)\}$ 構築アルゴリズムだと $O(|\varphi| \cdot N^3)$ になって、 その設定だと Theorem 1 は 係数 が大きくなりすぎて遅いということを、 意味のある形で言及した方が良い。

以降、Theorem 1 の証明をやっていきます。

6.1 Proof of Theorem 1

Here we use $P, Q, R \ldots$ to denote binary predicates and $U, \ldots$ for unary predicates.

For a given graph (model) $\mathcal{G}$, we write $N_{\mathcal{G}}$ or simply N to denote the number of vertices of $\mathcal{G}$.

Our idea: Quantifier elimination (Simple case)

Our query evaluation strategy is evaluating the innermost (quantifier-free formula) to the outermost along with quantifier eliminating.

Let us consider the following situation:

$$Qx. \cdots (Qy. \cdots (Qz. \cdots (Qx. \cdots (Qy. \cdots (\exists x.(P(y, x) \wedge R(x, z)))))))$$

To eliminate $\exists z.P(y, x) \wedge R(x, z)$, we introduce a new predicate defined as follows:

$$S(x, y) := \exists z.P(x, z) \wedge Q(z, y).$$

It is worth noting that S is just the composed predicate of P and Q. So, we write $P \odot Q$ instead of S.

Using this predicate, we can rewrite the above one to the following one:

$$Qx \cdots (Qy \cdots (Qz \cdots (Qx. (Qy. (P \odot R)(y, z)))))$$

[3] 何らかの論文をciteして、今の記録がどれくらいか書いておくと分かりやすい

Continuing this process, if we eventually enter $\forall x.\exists y.T(x,y)$:

1. building a unary predicate $T_y(x) := \exists y.T(x,y)$ in $O(N^2)$-time, we rewrite it to $\forall x.T_y(x)$.
2. we then simply evaluate $\forall x.T_y(x)$ as $\bigwedge_{1\leq i\leq N} T_y(i)$ in $O(N)$-time.

Time-complexity of basic operations

Here we enumerate the time-complexity of our basic operations, which will be used later:

- Building a new predicate $Q(x,y) := P(x,y) \bowtie R(x,y)$ ($\bowtie \in \{\wedge, \vee\}$) needs $O(N^2)$-time.
- Transposing a predicate $P^T(x,y) := P(y,x)$ needs $O(N^2)$-time.
- Evaluating unary predicates $\mathcal{Q}x.P(x)$ ($\mathcal{Q} \in \{\forall, \exists\}$) needs $O(N)$-time.
- Building a new predicate $Q(x) := \mathcal{Q}y.P(x,y)$ ($\mathcal{Q} \in \{\exists, \forall\}$) needs $O(N^2)$-time.

So, the remaining operator is predicate composing $(P \odot Q)(x,y) := \exists z.P(x,z) \wedge Q(z,y)$ from predicates P and Q. Although a very simple matrix multiplication requires $O(N^3)$-time, we can build such $P \odot Q$ using fast matrix multiplication algorithms.

▶ **Proposition 3** (Fact). *From two predicates P and Q, we can build the composited predicate $P \odot Q$ in $O(N^\omega)$-time.*

6.2 Quantifier elimination: General case

If we encounter terms of the very restricted form like $\exists z.P(x,z) \wedge Q(z,y)$, we can replace it by $(P \odot Q)(x,y)$ with eliminating the quantifier occurrence $\exists z$.

In other words, if we encounter more general form of $\exists z. \cdots$, we need to translate it to some adequate form for quantifier elimination. Let us consider the following term:

$$\exists z.\Big(\big(P(x,z) \vee P(y,z) \vee P(x,y)\big) \wedge \big(Q(x,z) \vee Q(x,y)\big) \wedge \big(R(y,z) \vee R(x,y)\big) \Big).$$

We cannot directly apply our quantifier elimination strategy because $\exists$-quantifiers do not distribute on $\wedge$: i.e., $(\exists z.\varphi_1 \wedge \varphi_2) \neq (\exists z.\varphi_1) \wedge (\exists z.\varphi_2)$.

To apply our quantifier elimination procedure, DNF is an adequate form. However, as well-known, converting-to-DNF translation generates an exponential-large expression in the worst case. For our example, we fist translate the first $(\cdots) \wedge (\cdots)$ subterm as follows:

$$\begin{aligned} &\big(P(x,z) \vee P(y,z) \vee P(x,y)\big) \wedge \big(Q(x,z) \vee Q(x,y)\big) \\ &\quad \Rightarrow \\ &\quad (P(x,z) \wedge Q(x,z)) \vee (P(y,z) \wedge Q(x,z)) \vee (P(x,y) \wedge Q(x,z)) \\ &\vee (P(x,z) \wedge Q(x,y)) \vee (P(y,z) \wedge Q(x,y)) \vee (P(x,y) \wedge Q(x,y)). \end{aligned}$$

We continue to normalize this term with $R(y,z) \vee R(x,y)$ and it generates DNF with 12 bases of the form $P(\_,\_) \wedge Q(\_,\_)$.

Since our goal is to design an $O(F(k,w) \cdot |\varphi| \cdot N^\omega)$-time algorithm, we cannot adopt converting-to-DNF transformations.

6.3 Tseytin-like (?) transformation for General Cases

As we have seen above, explicit translations to DNF does not work well since it may explode given expression to an exponential size.

We here develop a translation inspired by Tseytin's transformation.[4]

> 実際はインスパイアされたという程でもないんですが、 とはいえよく知られた構成な気もするので、何かciteできるものがあればしたいという感じ。

Let us revisit the above example:

$$\exists z. \Big(\big(P(x,z) \lor P(y,z) \lor P(x,y)\big) \land \big(Q(x,z) \lor Q(x,y)\big) \land \big(R(y,z) \lor R(x,y)\big) \Big).$$

First, we generate all possible *assignments* to predicates. For our example, we have the following assignments:

	$P(x,z)$	$P(y,z)$	$P(x,y)$	$Q(x,z)$	$Q(x,y)$	$R(y,z)$	$R(x,y)$
α_1	0	0	0	0	0	0	0
α_2	0	0	0	0	0	0	1
α_3	0	0	0	0	0	1	0
$\vdots$							
α	1	0	1	0	1	0	0
$\vdots$							
$\alpha_\star$	1	1	1	1	1	1	1

Let $\mathcal{A}$ be the set of all assignments.
Let $[\![\Psi]\!]_\alpha$ be a boolean value obtained by evaluationg Ψ under an assignment α.
For our example expression ψ, $[\![\psi]\!]_\alpha = 0$ using α appearing in the above table.
If we change α as $\alpha(R(y,z)) \leftarrow 1$, then $[\![\psi]\!]_\alpha = 1$.
Introducing assignment leads to the following useful proposition.

▶ **Proposition 4.**

$$\Psi(x,y,z) \iff \bigvee_{\alpha \in \mathcal{A}} \Big(\alpha \land [\![\Psi]\!]_\alpha \Big).$$

Especially, let $\mathcal{V} = \{\alpha \in \mathcal{A} : [\![\Psi]\!]_\alpha\}$, *then we have:*

$$\Psi(x,y,z) \iff \bigvee_{\alpha \in \mathcal{V}} \alpha(x,y,z). \qquad (\star)$$

By the definition of assignments, α takes the following form:

$$\alpha(x,y,z) \equiv P_1(x,y) \land P_1(x,z) \land P_2(y,z) \land \cdots \land P_k(x,z).$$

It means that the right-hand side of $(\star)$ takes DNF. Therefore,

$$\exists z.\Psi(x,y,z) \iff (\exists z.\alpha_1(x,y,z)) \lor (\exists z.\alpha_2(x,y,z)) \lor \cdots \lor (\exists z.\alpha_{|\mathcal{V}|}(x,y,z)).$$

It suffices to eliminating an $\exists$-quantifier from $\exists z.\alpha(x,y,z)$ for an assignment α.
We first divide $\alpha(x,y,z)$ into two parts α_z and α_{-z} as follows:

$$\alpha(x,y,z) \equiv \underbrace{(P_1(x,z) \land P_2(z,x) \land P_1(y,z) \land \cdots)}_{\alpha_z : z-\text{involving-terms}} \land \underbrace{(P_1(x,y) \land P_1(y,x) \land P_2 \land \cdots)}_{\alpha_{-z} : z-\text{non-involving-terms}}$$

[4] https://en.wikipedia.org/wiki/Tseytin_transformation

We now have the following:

$$\Big(\exists z.\alpha(x,y,z)\Big) \iff \Big(\alpha_{-z}(x,y) \wedge (\exists z.\alpha_z(x,y,z))\Big).$$

Finally, we transform α_z in the following steps (order):

1. If α_z has a term $P(z,x)$, we replace it with $\widetilde{P}(x,z)$ where $\widetilde{P}(a,b) = P(b,a)$.
2. Similarly, we change terms of the form $P(y,z)$ to $\widetilde{P}(z,y)$.
3. At this point,

$$\alpha_z(x,y,z) \equiv \underbrace{(P_1(x,z) \wedge P_3(x,z) \wedge P_4(x,z) \wedge \cdots)}_{\Psi_x} \wedge \underbrace{(P_2(z,y) \wedge P_3(z,y) \wedge P_5(z,y) \wedge \cdots)}_{\Psi_y}.$$

4. Let Ψ_x (resp. Ψ_y) be the set of predicates appearing in the above former (resp. latter) part.
5. Let us introduce two new predicates:

$$\pi(x,z) := \bigwedge_{P \in \Psi_x} P(x,z), \qquad \lambda(z,y) := \bigwedge_{Q \in \Psi_y} Q(z,y).$$

6. We reach our goal:

$$\alpha_z(x,y,z) = \pi(x,z) \wedge \lambda(z,y).$$

Using the final form, we obtain the following:

$$\begin{aligned}
\Big(\exists z.\alpha(x,y,z)\Big) &\iff \Big(\alpha_{-z}(x,y) \wedge (\exists z.\pi(x,z) \wedge \lambda(z,y))\Big)\\
&\iff \Big(\alpha_{-z}(x,y) \wedge (\pi \odot \lambda)(x,y)\Big)\\
&\iff (\alpha_{-z} \wedge (\pi \odot \lambda))(x,y)
\end{aligned}$$

6.4 Considering Complexity

To eliminate an $\exists$-quantifier from an assignment α, we introduce a new single predicate $\alpha_{-z} \wedge (\pi \odot \lambda)$.

Now we have a question: To eliminate the $\exists$-quantifier from $\exists z.\Psi(x,y,z)$, from the following characterization

$$\Big(\exists z.\Psi(x,y,z)\Big) \iff \Big(\exists z. \bigvee_{\alpha \in \mathcal{V}} \alpha(x,y,z)\Big) \iff \bigvee_{\alpha \in \mathcal{V}} \Big(\exists z.\alpha(x,y,z)\Big) \iff \bigvee_{\alpha \in \mathcal{V}} (\alpha_{-z} \wedge (\pi \odot \lambda))(x,y)$$

can we avoid to introduce $O(2^k)$ predicates if we have k-predicates?? Please recall that each assignment is a choice from $9k$-literals from $\mathcal{L}$ where

$$\mathcal{L} = \{Q(a,b) : a,b \in \{x,y,z\}, Q \in \{P_1, P_2, \ldots, P_k\}\}.$$

Yes. We can avoid introducing a large number of predicates because we just need to introduce the following single (but large) predicate:

$$(\alpha^1_{-z} \wedge (\pi^1 \odot \lambda^1)) \vee (\alpha^2_{-z} \wedge (\pi^2 \odot \lambda^2)) \vee \cdots \vee (\alpha^n_{-z} \wedge (\pi^n \odot \lambda^n))$$

for $\mathcal{V} = \{\alpha^1, \alpha^2, \ldots, \alpha^n\}$.

Let us consider what happens when repeatedly eliminating quantifiers from the innermost to the outermost:

$$\begin{array}{l} \exists y.\Big((\exists z.\cdots) \bowtie (\exists z.\cdots) \bowtie \cdots \bowtie (\exists z.\cdots)\Big) \\ \quad\Rightarrow \\ \exists y.\left(\begin{array}{l}\text{Q. how many predicates appear in this scope?} \\ \text{A. } k+w \text{ where } w \text{ is the quantifier width of } \psi\end{array}\right) \\ \quad\Rightarrow \\ \text{Eliminating } (\exists y.) \text{ introduces a single new predicate.} \end{array}$$

Since the required number of quantifier elimination (= the quantifier depth) is bounded by the size of a given expression $|\psi|$, we obtain the following our theorem in the end.

▶ **Theorem 1** (Restatement). *Let* $\varphi : \mathbb{FO}^3_{k,w}$.
For a given model (graph) $\mathcal{G}$*, we can decide if* $\mathcal{G} \models \varphi$ *in* $O(2^{k+w} \cdot |\varphi| \cdot N^\omega)$.

7 FO2 model checking

$\mathbb{FO}^2$の論理式 ψ と、グラフ $\mathcal{G}$ について $\mathcal{G} \models \psi$ のモデル検査問題は $O(|\psi| \cdot |\mathcal{G}|)$ で解ける気がします。 ただし、$|\mathcal{G}|$は頂点数ではなく、述語（つまり辺）の定義の記述長も合わせたものです。 面白そうなら書いても良いかと思うのですが、$\mathbb{FO}^3$の話だけにしたほうが、話がぼやけないならなくても良いかなという気がします。
Dense graphについては (x, y) の値の集合を構築する方法で $O(|\psi| \cdot N^2)$ になります。 この N は $\mathcal{G}$ の頂点数の方です。 ただ、sparse graphの時には N^2 は$|\mathcal{G}|$について線形ではないので、そこを改良したいという感じです。
アイディアは、$\mathbb{FO}^3$の時と比べると、$\exists y.P(x, y) \wedge Q(y, x) \simeq (P \odot Q)$ の計算について $(P \odot Q)(x, x)$ の diagonal な形しか要求しないので、行列積を全部やる必要が多分なさそうとかそういう感じです。
Sparse graphの話なので、もしかすると Impagliazzo たちがやっているかもしれません。

Yo: [3] Lemma 9.1 Base Case は $O(|\psi| \cdot |\mathcal{G}|)$??

References

1 Yijia Chen, Michael Elberfeld, and Moritz Müller. The parameterized space complexity of model-checking bounded variable first-order logic. *Logical Methods in Computer Science*, Volume 15, Issue 3, 2019. doi:10.23638/LMCS-15(3:31)2019.
2 Jiawei Gao and Russell Impagliazzo. The fine-grained complexity of strengthenings of first-order logic, 2019. URL: https://eccc.weizmann.ac.il/report/2019/009/.
3 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. *ACM Trans. Algorithms*, 15(2):23:1–23:35, 2018. doi:10.1145/3196275.
4 Moshe Y. Vardi. On the complexity of bounded-variable queries (extended abstract). In *PODS*, PODS, page 266–276. ACM, 1995. doi:10.1145/212433.212474.
5 Ryan Williams. Faster decision of first-order graph properties. In *CSL-LICS*, page 1–6. ACM, 2014. doi:10.1145/2603088.2603121.