Faster decision of first-order graph properties

Ryan Williams
Stanford University
rrw@cs.stanford.edu

Abstract

First-order logic captures a vast number of computational problems on graphs. We study the time complexity of deciding graph properties definable by first-order sentences in prenex normal form with k variables. The trivial algorithm for this problem runs in $O(n^k)$ time on n-node graphs (the big-O hides the dependence on k).

Answering a question of Miklós Ajtai, we give the first algorithms running faster than the trivial algorithm, in the general case of arbitrary first-order sentences and arbitrary graphs. One algorithm runs in $O(n^{k-3+\omega}) \leq O(n^{k-0.627})$ time for all $k \geq 3$, where $\omega < 2.373$ is the $n \times n$ matrix multiplication exponent. By applying fast rectangular matrix multiplication, the algorithm can be improved further to run in $n^{k-1+o(1)}$ time, for all $k \geq 9$. Finally, we observe that the exponent of $k-1$ is optimal, under the popular hypothesis that CNF satisfiability with n variables and m clauses cannot be solved in $(2-\epsilon)^n \cdot \text{poly}(m)$ time for some $\epsilon > 0$.

Categories and Subject Descriptors G.2.2 [*Graph Theory*]: Graph algorithms

Keywords first-order logic, graph algorithms, matrix multiplication, satisfiability

1. Introduction

One goal of finite model theory is to achieve a fine-grained understanding of the complexity of deciding first-order sentences on finite structures. In particular, the class of finite graphs is of prime importance. We focus on first order formulas ϕ in prenex normal form (PNF) with k quantifiers, i.e., of the form

$$\phi = (Q_1\ v_1) \cdots (Q_k\ v_k)\psi(v_1, \ldots, v_k),$$

where each $Q_i \in \{\exists, \forall\}$, and the predicate ψ is a *boolean formula over graphs*, i.e., over atoms of the form $(v_i = v_j)$ and the edge relation $E(v_i, v_j)$. The *model checking* problem for first-order logic on graphs is defined as follows:

MC(FO)
Instance: A graph G and first-order sentence ϕ
Problem: Decide if $G \models \phi$.

CSL-LICS '14, July 14 - 18 2014, Vienna, Austria.

ACM 978-1-4503-2886-9/14/07…$15.00.
http://dx.doi.org/10.1145/2603088.2603121

MC(FO) is PSPACE-complete, and therefore appears very unlikely to be solvable efficiently [Sto74, Var82]. The PSPACE-completeness of the problem stems from the fact that the sentence ϕ is given as part of the input. However, model-checking is still very interesting for fixed sentences with k quantifiers, where k is a constant. By exhaustive search, it is straightforward to decide a first-order sentence with k quantifiers in $O(n^k)$ time; that is, for every fixed sentence ϕ, we may always find *some* polynomial time algorithm for model-checking it.

Is n^k the best possible running time we can hope for, in general? To our knowledge, no algorithmic progress on general first-order model checking for graphs has yet been reported. There has been significant prior work on important special cases. A prominent example of algorithmic progress on restricted structures is the work of Courcelle [Cou90], who famously proved that the model-checking problem for monadic second-order logic can be solved in linear time on all graphs of *bounded treewidth*. Dvorak, Král, and Thomas [DKT10] have shown that deciding first-order graph properties in graphs of *bounded expansion* (such as planar graphs) can be done in linear time.

Two examples most relevant to this paper are the k-clique problem and the k-dominating set problem. The *k-clique problem* asks whether a given graph on n nodes contains a clique of size k, and can be expressed as the sentence:

$$(\exists v_1)(\exists v_2) \cdots (\exists v_k) \left[\bigwedge_{i \neq j} E(v_i, v_j) \right].$$

The *k-dominating set problem* asks whether a given graph contains a set S of k nodes such that all nodes not in S have a neighbor in S, and can be expressed with $k+1$ quantifiers as:

$$(\exists v_1)(\exists v_2) \cdots (\exists v_k)(\forall w) \left[\bigvee_i ((w = v_i) \vee E(v_i, w)) \right].$$

Both problems are known to admit algorithms that run in $o(n^q)$ time, where q is the number of quantifiers in the sentence defining the problem. For k-clique, the best known algorithms run in $O(n^{\delta k})$ time for a finite number of $\delta \in (0.8, 1)$ depending on k [NP85, EG04]. For $(k-1)$-dominating set, the best known algorithms run in $O(n^{k-\varepsilon+o(1)})$ time for some $\varepsilon \in (0, 1]$ depending on k [EG04]. In particular, the problem is solvable in $n^{k-1+o(1)}$ for all $k \geq 8$.

Giving a partial negative answer to the above question, Chen et al. [CHKX06] proved that the k-clique problem cannot be solved in $n^{o(k)}$ time assuming a reasonable conjecture in complexity theory: that the 3SAT problem is not solvable in $2^{o(n)}$ time. Results of Frick and Grohe [FG04] show that, under the same conjecture, first-order model checking on the class of binary trees cannot be solved in $2^{2^{2^{o(k)}}} \cdot p(n)$ time (for all polynomials $p(n)$).[1]

1.1 Our Results

We find that the k-clique problem and k-dominating set problem, both of central importance to parameterized complexity [DF99, FG06], are also central to understanding the time complexity of first-order logic on graphs in general.

k-Clique and Existential FOL on Graphs An *existential first order sentence* contains only existentially quantified variables. For instance, the *k-clique problem* is existential. Our first theorem is that k-clique is the "hardest" existential first-order sentence: the time complexity of k-clique determines an upper bound on the time complexity of all other existential queries. This is intriguing, as the predicate in the k-clique sentence is quite simple, yet checking it is sufficient for *all* sentences with k quantifiers.

Theorem 1.1 *Let $T(n,k)$ be the time complexity of deciding whether a given n-node graph contains a k-clique. Then every existential first-order sentence with k quantified variables can be decided on n-node graphs in $2^{O(k^2)}T(n,k)$ time.*

That is, faster algorithms for k-clique imply faster algorithms for *every* existential first-order sentence. The following are immediate corollaries:

Corollary 1.1 *The k-clique problem is in $O(f(k)T(n,k))$ time for some function $f : \mathbb{N} \to \mathbb{N}$ iff the model checking problem for existential first-order logic on graphs is in $O(g(k)T(n,k))$ time for some $g : \mathbb{N} \to \mathbb{N}$ on k quantifier sentences.*

Corollary 1.2 *There is a universal $\delta < 1$ such that every existential first-order sentence with k quantifiers can be decided on n-node graphs in $2^{O(k^2)}n^{\delta k}$ time.*

k-Dominating Set and FOL on Graphs The k-dominating set problem plays a similar, but not analogous, role in the general case of arbitrary sentences. Although the $(k-1)$-dominating set problem has $(k-1)$ existential quantifiers and one universal, we present algorithms for model-checking sentences with k quantifiers that match the known time complexity of the $(k-1)$-dominating set problem, and observe that the *difficulty* of $(k-1)$-dominating set must be overcome to improve this time complexity any further.

Our first algorithm gives a non-trivial time bound for three-quantifier sentences. In the following, let $\omega < 2.373$ be the matrix multiplication exponent [Vas12].

Theorem 1.2 *Every first-order sentence on n-node graphs with three quantifiers can be decided in $\tilde{O}(n^{\omega})$ time.*

Corollary 1.3 *Let $k \geq 3$. Every k-quantifier first-order sentence on n-node graphs can be decided in $\tilde{O}(n^{k-3+\omega})$ time.*

To compare with prior work, Eisenbrand and Grandoni [EG04] show that $(k-1)$-dominating set is solvable in about $n^{k-3+\omega}$ time, for all $k \geq 3$. Applying fast rectangular matrix multiplication [Cop97, HP98, Gal12], the algorithm can be improved for large enough values of k:

[1] Technically, Frick and Grohe prove their results assuming $FPT \neq W[1]$, but this follows from the assumption that 3SAT is not in $2^{o(n)}$ time [DF99].

Theorem 1.3 *Let $k \geq 9$. Every k-quantifier first-order sentence on n-node graphs can be decided in $n^{k-1+o(1)}$ time.*

This algorithm also matches the known running times for solving $(k-1)$-dominating set [EG04] for large k.

Remark 1 *In fact, there is* **no** *special dependency on graph structures in our algorithms, other than the fact that a graph is a binary relation over a vertex set. All our algorithms can be modified to work for first-order prenex sentences over any vocabulary that is defined with a finite set of unary and binary relations.*

The $k-1$ exponent of Theorem 1.3 may look like an incremental advance, which may soon be improved further. However, prior work of Pătraşcu and the author shows that a great advance in theoretical SAT solving would result if this $k-1$ exponent could be improved.

Theorem 1.4 (Pătraşcu-Williams [PW10]) *For every $\varepsilon > 0$ and $k \geq 3$, if the $(k-1)$-dominating set problem can be solved in $O(n^{k-1-\varepsilon})$ time, then there is a $\delta > 0$ such that satisfiability of general CNF formulas with n variables and m clauses can be solved in $(2-\delta)^n \cdot m^{O(1)}$ time.*

It follows that any algorithmic improvement over Theorem 1.3 would resolve a major open question about SAT solving:

Corollary 1.4 *Let $k \geq 4$. If the model-checking problem for k-quantifier first-order sentences over graphs can be solved in $O(n^{k-1-\varepsilon})$ time for some $\varepsilon > 0$, then the Strong Exponential Time Hypothesis (SETH) is false.*

More discussion on these issues can be found in Section V of the paper.

2. Preliminaries

We assume some basic familiarity with algorithms and complexity theory.

Let us outline some notation particularly important for this paper. For a formula ϕ with free variable x, we use $\phi|_{x=a}$ to denote the formula obtained by substituting a in place of the free occurrences of x in ϕ. The notation $\tilde{O}(f(n))$ denotes an upper bound of the form $O(f(n)\log^c n)$ for some constant c. The notation $f(n)^{1+o(1)}$ denotes an upper bound that is less than $f(n)^{1+\varepsilon}$ for all constant $\varepsilon > 0$.

For the purposes of this paper, an *atom* is a relation of the form $(v_i = v_j)$, $(v_i \neq v_j)$, $E(v_i, v_j)$, or $\neg E(v_i, v_j)$ where v_i and v_j are (node) variables. A *boolean formula over graphs* is a boolean formula comprised of such atoms. More formally, the above atoms are boolean formulas over graphs, and given two formulas F_1 and F_2, $F_1 \wedge F_2$, $F_1 \vee F_2$, and $\neg F_1$ are also boolean formulas over graphs.

3. The Case of Existential Sentences

We begin with the algorithm for deciding existential sentences with k quantifiers:

Reminder of Theorem 1.1 *Let $T(n,k)$ be the time complexity of deciding whether a given n-node graph contains a k-clique.*

Then every existential first-order sentence with k quantifiers can be decided on n-node graphs in $2^{O(k^2)}T(n,k)$ time.

Proof. Let $\phi = (\exists\, v_1)\cdots(\exists\, v_k)[\psi(v_1,\ldots,v_k)]$ be a first-order sentence on graphs, where the predicate ψ is a boolean formula over graphs. Let $G = (\{1,\ldots,n\},E)$ be a given graph on which we wish to decide $G \models \phi$.

Without loss of generality, we can assume that the predicate ψ is in disjunctive normal form (DNF): that is, ψ is an OR of ANDs of atoms. Treating k as a constant, we observe that converting ψ into this form only increases the size of ψ by a constant factor. That is, if ψ has k free variables, the size of ψ in DNF is at most $2^{O(k^2)}$, since the number of possible atoms over the variables is $O(k^2)$.

Let C be the set of all conjuncts in the DNF ψ. Each conjunct c in C can be viewed as a set of atoms, each of which are either $E(v_i,v_j)$ or $(v_i = v_j)$, possibly with negations. We therefore can write

$$\phi = (\exists\, v_1)(\exists\, v_2)\cdots(\exists\, v_k)\left[\bigvee_{c\in C}\left(\bigwedge_{a\in c} a\right)\right].$$

By commutativity of OR and existential quantifiers, we can rewrite ϕ equivalently as

$$(\exists\, c\in C)(\exists\, v_1)(\exists\, v_2)\cdots(\exists\, v_{k-1})(\exists\, v_k)\left[\bigwedge_{a\in c} a\right].$$

(Of course this is, strictly speaking, no longer a sentence in first-order logic over graphs; this fact will not trouble us.) We now express the decision of this sentence as an OR of $|C|$ instances of the k-clique problem, as follows. For every $c \in C$, we shall build a graph G_c that contains a k-clique if and only if the conjunct c can be satisfied by the graph G.

First, we define the vertex set of G_c to be k times the size of G, being the union of k disjoint sets $V_1,\ldots,V_k$, each set having n vertices. For each $i = 1,\ldots,k$, we associate an (arbitrary) permutation $\pi_i : V_i \to \{1,\ldots,n\}$ indexing the vertices. For $i \neq j$ and vertices $w_i \in V_i$ and $w_j \in V_j$, put an edge $\{w_i,w_j\} \in E_c$ if and only if every atom in the conjunct $A_c := \bigwedge_{a\in c} a$ referring to v_i and v_j is true when v_i is assigned $\pi_i(w_i)$ and v_j is assigned $\pi_i(w_j)$. More formally, an edge is placed between w_i and w_j if and only if all of the following requirements are met.

- If $(v_i = v_j)$ (respectively, $(v_i \neq v_j)$) is an atom in A_c, then we require that $\pi_i(w_i) = \pi_j(w_j)$ (respectively, $\pi_i(w_i) \neq \pi_j(w_j)$).
- If $E(v_i,v_j)$ (respectively, $\neg E(v_i,v_j)$) is an atom in A_c, then we require that $E(\pi_i(w_i),\pi_j(w_j))$ is true of G (respectively, $E(\pi_i(w_i),\pi_j(w_j))$ is not true of G).

Note that G_c is a k-partite graph, and hence every k-clique $C = \{c_1,\ldots,c_k\}$ contains exactly one vertex c_i from each V_i, for all $i = 1,\ldots,k$. We observe that G_c contains a k-clique C if and only if the conjunct A_c is true of G, under the variable assignment where for all $i = 1,\ldots,k$, variable v_i is assigned the vertex $\pi_i(c_i)$ from G. (Suppose $C = \{c_1,\ldots,c_k\}$ is a k-clique of G_c. Since every possible edge in C is present, it follows that every atom a in A_c must be satisfied under the variable assignment $v_i \mapsto \pi_i(c_i)$. Moreover, for every variable assignment that satisfies A_c, the corresponding subset S of vertices in G_c forms a k-clique.)

Therefore, by determining whether any of the $2^{O(k^2)}$ graphs G_c contain a k-clique, we may determine the truth of the original sentence ϕ on the graph G. $\square$

4. The General Case

Now we turn to the general case of model-checking arbitrary first-order formulas over graphs. It shall be convenient to start with an improved algorithm for the case of three quantified variables, and develop the other algorithms by building on the arguments given in this case.

Reminder of Theorem 1.2 *Every first-order sentence with three quantified variables on n-node graphs can be decided in $\tilde{O}(n^\omega)$ time.*

Proof. Let $\phi = (Q_1\ v_1)(Q_2\ v_2)(Q_3\ v_3)\psi(v_1,v_2,v_3)$ be a first-order sentence on graphs, where each $Q_i \in \{\exists,\forall\}$, and the predicate ψ is a boolean formula over graphs. Without loss of generality, we can massage ϕ to satisfy the conditions:

1. The third quantifier Q_3 is existential. This can be ensured by simply complementing the formula if Q_3 is universal, and deciding the complement instead.
2. The predicate ψ is in disjunctive normal form (DNF). The argument here is analogous to Theorem 1.1; the size of ψ in DNF is at most $2^{O(k^2)}$.

Let C be the set of conjuncts in ψ. Each conjunct $c \in C$ can be construed as a set of atoms, hence ϕ is expressible as

$$\phi = (Q_1\ v_1)(Q_2\ v_2)(\exists\, v_3)\left[\bigvee_{c\in C}\left(\bigwedge_{a\in c} a\right)\right].$$

By the commutativity of existential quantifiers and basic properties of OR, we can rewrite ϕ equivalently as

$$(Q_1\ v_1)(Q_2\ v_2)(\exists\, c\in C)(\exists\, v_3)\left[\bigwedge_{a\in c} a\right].$$

Let

$$\psi'(v_1,v_2) = (\exists\, c\in C)(\exists\, v_3)\left[\bigwedge_{a\in c} a\right],$$

and suppose a graph G is given with vertex set $\{1,\ldots,n\}$. We shall produce an $n \times n$ matrix M over $\{0,1\}$, such that $M(i,j) = 1$ if and only if $\psi'|_{v_1=i,v_2=j}$ is true on graph G. The knowledge of matrix M clearly suffices for determining the truth value of ϕ on G, in $O(n^2)$ time.

For every conjunct $c \in C$, define $n \times n$ matrices X_c, Y_c, and Z_c over $\{0,1\}$ as follows:

- $X_c[i,k] = 1$ iff every atom in c containing v_1 and v_3 is satisfied by the assignment $v_1 = i$ and $v_3 = k$.
- $Y_c[k,j] = 1$ iff every atom in c containing v_3 and v_2 is satisfied by the assignment $v_3 = k$ and $v_2 = j$.
- $Z_c[j,i] = 1$ iff every atom in c containing v_1 and v_2 is satisfied by the assignment $v_2 = j$ and $v_1 = i$.

Now for every $c \in C$, multiply X_c and Y_c, obtaining the $n \times n$ matrix M_c. Replace every nonzero entry of $M_c[i,j]$ with the constant 1, so that M has 0-1 entries. It follows from the definition of matrix multiplication that $M_c[i,j] = 1$ iff there is a $k \in [n]$ such that all atoms in c containing $\{v_1,v_3\}$ are satisfied by $v_1 = i, v_3 = k$, and all atoms in c containing $\{v_2,v_3\}$ are satisfied by $v_2 = j, v_3 = k$.

Compute the $n \times n$ matrix

$$M[i,j] = \sum_{c\in C}\left(M_c[i,j]\cdot Z_c[j,i]\right),$$

and replace all nonzero entries of $M[i,j]$ with the constant 1 to make M a boolean matrix. Observe that M can be computed in $\tilde{O}(n^2)$ arithmetic operations, since $|C|$ is bounded by a constant. The overall running time is at most $\tilde{O}(n^\omega)$.

We claim that $M[i,j] = 1$ if and only if $\psi'|_{v_1=i,v_2=j}$ is true on graph G, which will complete the proof. This follows from the chain of equivalences:

$$\begin{aligned}
&\psi'|_{v_1=i,v_2=j} \text{ is true on } G\\
\iff\quad & (\exists\, c \in C)(\exists\, v_3)\left[\bigwedge_{a\in c} a|_{v_1=i,v_2=j}\right]\\
\iff\quad & (\exists\, c \in C)(\exists\, k \in [n])\\
& [v_1 = i, v_3 = k \text{ satisfies all atoms in } c\\
& \text{containing } \{v_1, v_3\},\\
& \ v_2 = j, v_3 = k \text{ satisfies all atoms in } c\\
& \text{containing } \{v_2, v_3\},\\
& \ v_1 = i, v_2 = j \text{ satisfies all atoms in } c\\
& \ \text{containing } \{v_1, v_2\}]\\
\iff\quad & (\exists\, c \in C)[M_c[i,j] \neq 0 \text{ and } Z_c[j,i] = 1]\\
\iff\quad & M[i,j] \neq 0
\end{aligned}$$

□

The algorithm for the three variable case can be easily extended to sentences with k quantifiers:

Corollary 4.1 *Every k-quantifier first-order sentence on n-node graphs can be decided in $\tilde{O}(n^{k-3+\omega})$ time.*

Proof. Given a graph G, systematically examine all ways to assign vertices to the first $k-3$ variables of the given sentence ϕ. For each of these n^{k-3} assignments, substitute in values for the $k-3$ variables into ϕ, simplifying the formula (removing true and false constants) as necessary. The resulting formula has three variables, and can be solved in $\tilde{O}(n^\omega)$ time by Theorem 1.2. □

For sufficiently large k, rectangular matrix multiplication can be applied to obtain a better running time.

Reminder of Theorem 1.3 *Let $k \geq 9$. Every first-order sentence with k quantifiers can be decided on n-node graphs in $n^{k-1+o(1)}$ time.*

Proof. We reconsider the proof of Theorem 1.2 with the following modifications. Without loss of generality, we have a first-order sentence

$$\phi = (Q_1\, v_1)\cdots(Q_{k-1}\, v_{k-1})(\exists\, v_k)[\psi]$$

where ψ is DNF. Let $k_1 = \lfloor (k-1)/2 \rfloor$, $k_2 = \lceil (k-1)/2 \rceil$, and note that $k_1 + k_2 = k-1$.

Instead of forming the $n \times n$ 0-1 matrices X_c, Y_c, and Z_c as defined in the proof of Theorem 1.2 (for each conjunct c in the DNF), we define 0-1 matrices X_c, Y_c, and Z_c to have dimensions $n^{k_1} \times n$, $n \times n^{k_2}$, and $n^{k_1} \times n^{k_2}$, respectively.

- The rows of X_c are indexed by all possible n^{k_1} assignments $(i_1, \ldots, i_{k_1}) \in \{1, \ldots, n\}^{k_1}$ to the variables $v_1, \ldots, v_{k_1}$ of ϕ, and the columns are indexed by all n possible assignments to the variable v_k. We define $X[(i_1, \ldots, i_{k_1}), j] := 1$ if and only if every atom in c containing a pair of variables from $(v_1, \ldots, v_{k_1}, v_k)$ is satisfied by the variable assignment $(i_1, \ldots, i_{k_1}, j)$.
- Similarly, the rows of Y_c are indexed by all n possible assignments to the variable v_k, and the columns are indexed by all possible n^{k_2} assignments to the variables $v_{k_1+1}, \ldots, v_{k-1}$. We define $Y[j, (i_1, \ldots, i_{k_2})] := 1$ if and only if every atom in c containing a pair of variables from $(v_{k_1+1}, \ldots, v_{k-1}, v_k)$ is satisfied by the assignment $(i_1, \ldots, i_{k_2}, j)$.
- Finally, the rows of Z_c are indexed by all possible n^{k_1} assignments to the variables $v_1, \ldots, v_{k_1}$ of ϕ, the columns are indexed by all possible n^{k_2} assignments to the variables $v_{k_1+1}, \ldots, v_{k-1}$, and $Z[(i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})] :=$ 1 if and only if all atoms in c containing a pair of variables from $(v_1, \ldots, v_{k-1})$ are satisfied by the assignment $(i_1, \ldots, i_{k-1})$.

Each of the matrices can be built in at most $\tilde{O}(n^{k-1})$ time. Analogously with the proof of Theorem 1.3, we observe that the expression $M_c[(i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})] =$

$$((X_c \cdot Y_c)[(i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})] \neq 0)$$

$$\wedge\ Z_c[(i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})]$$

is true if and only if the variable assignment $(v_1, \ldots, v_{k-1}) \mapsto (i_1, \ldots, i_{k-1})$ satisfies $(\exists\, v_k)[c]$.

Now we observe that knowing the value of

$$M_c[(i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})],$$

for all conjuncts c of ψ, and all $((i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})) \in [n]^{k_1} \times [n]^{k_2}$, is sufficient for determining the truth of ϕ. Moreover, we can recover the truth value of ϕ efficiently in $\tilde{O}(n^{k-1})$ time. One way to see this is to build a complete tree with branching factor n, depth $k-1$, and n^{k-1} leaves, corresponding to all possible choices for the first $k-1$ variables of ϕ. That is, each child of the root corresponds to a choice for the variable v_1; each child of those children corresponds to a choice for the variables v_1 and v_2, and so on. At every leaf, there is a choice $(i_1, \ldots, i_{k-1}) \in [n]^{k-1}$ for all variables $v_1, \ldots, v_{k-1}$. Suppose we label each such leaf with the truth value of

$$\bigvee_c M_c[(i_1, \ldots, i_{k_1}), (i_{k_1+1}, \ldots, i_{k-1})].$$

Then, evaluating this tree as a *two-player game tree* (with one player choosing the existential variables, and the other choosing the universal variables) in a standard dynamic programming fashion, we recover the truth value of ϕ in $\tilde{O}(n^{k-1})$ time.

It remains to estimate the complexity of computing the matrices M_c for each c. This corresponds to performing $2^{O(k^2)}$ matrix multiplications, each with dimensions $n^{k_1} \times n$ and $n \times n^{k_2}$, then comparing the entries of the result component-wise with the entries of another $n^{k_1} \times n^{k_2}$ matrix.

LeGall [Gal12], building on Coppersmith [Cop97], recently gave an algorithm for multiplying an $m \times m^\alpha$ matrix with an $m^\alpha \times m$ matrix, which uses only $m^{2+o(1)}$ arithmetic operations (additions and multiplications) when $0 < \alpha < 0.302$.

Suppose first that k is odd, so $k_1 = k_2 = (k-1)/2$. Set $m = n^{k_2}$. When $0.302 \cdot k_2 \leq 1$, or equivalently when $k \geq 7.6$, this matrix multiplication algorithm applies to our situation. The matrix multiplies then take $n^{2\cdot(k-1)/2+o(1)} \leq n^{k-1+o(1)}$ time. When $k \geq 9$ is even, the matrix multiplies have dimension $n^{k/2-1} \times n$ and $n \times n^{k/2}$, respectively. This can be decomposed into n matrix multiplies of dimensions $n^{k/2-1} \times n$ and $n \times n^{k/2-1}$. Each of these are computable separately in $n^{k-2+o(1)}$ time using the rectangular matrix multiply, and can be summed together in $n^{k-1+o(1)}$ time.

It follows that, for every integer $k \geq 9$, the algorithm can be implemented to run in $n^{k-1+o(1)}$ time for any fixed first-order sentence with k quantified variables. $\square$

5. Discussion: Can these running times be improved? In which cases?

An experienced algorithm designer might conjecture that the algorithms of the previous section are suboptimal, since they do not appear to exploit the full power of matrix multiplication. The next logical step would then be to find an algorithm for MC(FO) on graphs that runs in less than n^{k-1} time for sufficiently large k.

However, understanding what sorts of first-order queries can be answered on general graphs faster than n^{k-1} time, and which cannot, is far from straightforward. To illustrate, for some first-order sentences with seemingly complex quantifier prefixes, such as

$$(\exists v_1) \cdots (\exists v_{k/3})(\forall v_{k/3+1}) \cdots (\forall v_{2k/3})$$
$$(\exists v_{2k/3+1}) \cdots (\exists v_k) \left[\bigwedge_{i \neq j} E(v_i, v_j) \right],$$

we can adapt the reduction of Theorem 1.1 and the algorithm of Theorem 1.2 to answer such queries in time that is roughly equal to the running time for k-clique. (Note that the above first-order sentence is a version of the k-clique problem, but with some variables universally quantified.)

Theorem 5.1 *Let k be divisible by 3. For all quantifier types $Q_1, Q_2, Q_3 \in \{\exists, \forall\}$, every first order sentence of the form*

$$(Q_1\ v_1) \cdots (Q_1\ v_{k/3})(Q_2\ v_{k/3+1}) \cdots (Q_2\ v_{2k/3})$$
$$(Q_3\ v_{2k/3+1}) \cdots (Q_3\ v_k)[\bigwedge_{i \neq j} E(v_i, v_j)]$$

can be model-checked with any given n-node graph in $\tilde{O}(n^{\omega k/3})$ time.

That is, the problem can be readily solved in the same running time as the purely existential fragment (Theorem 1.1).

Proof. Following the Nešetřil-Poljak strategy for solving k-clique [NP85], we shall transform the problem of checking ϕ on a given graph G into a three-variable sentence ϕ' to be be evaluated over a larger graph G'. The formula ϕ' is simply a generalization of the 3-clique sentence:

$$\phi' = (Q_1\ v_1)(Q_2\ v_2)(Q_3\ v_3)[\bigwedge_{i \neq j} E(v_i, v_j)].$$

Given a graph $G = (V, E)$ on n nodes, we construct a larger tripartite graph G', where each part V_1, V_2, V_3 contains $O(n^{k/3})$ nodes, indexed by those $k/3$-sets of V corresponding to $k/3$-cliques in G. For every $a \neq b$, put an edge between vertices $\{s_1, \ldots, s_{k/3}\} \in V_a$ and $\{t_1, \ldots, t_{k/3}\} \in V_b$ if and only if $\{s_1, \ldots, s_{k/3}, t_1, \ldots, t_{k/3}\}$ is a $2k/3$-clique in G.

Each vertex in G' corresponds to a $k/3$-clique in G, and each edge in G' corresponds to a $2k/3$-clique in G. We observe that $G' \models \phi'$ if and only if $G \models \phi$. Applying Theorem 1.2, we can check whether $G' \models \phi'$ in time $\tilde{O}(n^{\omega k/3})$ time. $\square$

In contrast to the above positive results, for some first-order sentences with the quantifier prefix

$$\underbrace{\exists \cdots \exists}_{k-1} \forall,$$

it will be *difficult* to obtain even an $n^{k-1-\varepsilon}$ time algorithm for some $\varepsilon > 0$. Noting that the $(k-1)$-dominating set problem is definable with this quantifier prefix, we cite the following result from the literature:

Theorem 5.2 (Pătraşcu-Williams [PW10]) *For every $\varepsilon > 0$ and $k \geq 3$, if the $(k-1)$-dominating set problem can be solved in $O(n^{k-1-\varepsilon})$ time, then there is a $\delta > 0$ such that satisfiability of general CNF formulas with n variables and m clauses can be solved in $(2-\delta)^n \cdot m^{O(1)}$ time.*

This result is intriguing due to the popular Strong Exponential Time Hypothesis posed in the literature:

Conjecture 5.1 (SETH [IP01, IPZ01]) *For every constant $\delta < 1$ there is a clause width k such that the k-SAT problem cannot be solved in $2^{\delta n}$ time on formulas with n variables.*

The conjecture looks rather strong, however all known SAT solving algorithms do not contradict it. A rapidly growing thread of work [CIP09, DW10, PW10, LMS11, CNP+11, CDL+12, PP12, HKN12, Cyg12, CKN13, RV13, FHV13, WY14, AVW14, AV14] has shown that the Strong Exponential Time Hypothesis has many interesting consequences for the complexity of other natural problems.

It is easy to see that SETH implies the negation of the conclusion of Theorem 5.2. Therefore, if SETH is true then the running time exponent of our model-checking algorithm is optimal:

Reminder of Corollary 1.4 *Let $k \geq 4$. If the model-checking problem for k-variable first-order sentences over graphs can be solved in $O(n^{k-1-\varepsilon})$ time for some $\varepsilon > 0$, then the Strong Exponential Time Hypothesis (SETH) is false.*

Corollary 1.4, together with the algorithmic results of this paper, indicates that the tractability landscape of first-order model-checking on arbitrary graphs must have delicate structure; this landscape deserves further inquiry.

We conclude by highlighting two open problems:

1. Can one improve on the n^k running time of the trivial algorithm *without* using $n^{\Omega(k)}$ space and running time exponential in k? For instance, can first-order model checking on graphs be done in $O(n^{k-\varepsilon}k^c)$ time and $O((n+k)^c)$ space, for a fixed constant c? Can one give evidence that such an algorithm will be hard to find?
2. Early in the paper, it was remarked that the algorithms of this paper will work for first-order sentences over any vocabulary with a finite number of unary and binary relations, not just graphs. What about for vocabularies with ternary relations? To achieve a faster algorithm in that case, it appears we will need to compute a three-dimensional generalization of matrix multiplication more efficiently. Let $A, B, C \in \{0,1\}^{n \times n \times n}$; that is, A, B, and C are boolean tensors of order 3, or "3D matrices." We wish to compute the following 3D matrix in $n^{4-\varepsilon}$ time, for some $\varepsilon > 0$:
$$D[i,j,k] = \sum_{\ell=1}^{n} A[i,j,\ell] \cdot B[j,k,\ell] \cdot C[k,i,\ell].$$
Generalizing the ideas presented earlier for the three-variable case, such a product operation would allow us to more efficiently find (for example) a 4-clique in a 3-uniform hypergraph, which can be expressed as a first-order sentence over a ternary relation.

6. Acknowledgements

This work was initially conceived during a conversation with Miklós Ajtai of IBM Almaden. I am grateful to Miki for sharing his open problems with me.

References

[AV14] Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. *arXiv:1402.0054*, 2014.

[AVW14] Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *ICALP, to appear*, 2014.

[CDL+12] Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as CNF-SAT. In *IEEE Conference on Computational Complexity*, pages 74–84, 2012.

[CHKX06] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.*, 72(8):1346–1367, 2006.

[CIP09] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In *Parameterized and Exact Computation*, pages 75–85. Springer, 2009.

[CKN13] Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast Hamiltonicity checking via bases of perfect matchings. In *STOC*, pages 301–310, 2013.

[CNP+11] Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, J.M.M. van Rooij, and J.O. Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *FOCS*, pages 150–159, 2011.

[Cop97] Don Coppersmith. Rectangular matrix multiplication revisited. *Journal of Complexity*, 13:42–49, 1997.

[Cou90] Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.

[Cyg12] Marek Cygan. Deterministic parameterized connected vertex cover. In *Algorithm Theory–SWAT 2012*, pages 95–106. Springer, 2012.

[DF99] Rodney G. Downey and Michael R. Fellows. Springer-Verlag, 1999.

[DKT10] Zdenek Dvorak, Daniel Král, and Robin Thomas. Deciding first-order properties for sparse graphs. In *FOCS*, pages 133–142, 2010.

[DW10] Evgeny Dantsin and Alexander Wolpert. On moderately exponential time for SAT. In *Proc. 13th International Conference on Theory and Applications of Satisfiability Testing*, pages 313–325, 2010.

[EG04] Friedrich Eisenbrand and Fabrizio Grandoni. On the complexity of fixed parameter clique and dominating set. *Theor. Comput. Sci.*, 326(1-3):57–67, 2004.

[FG04] 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.

[FG06] Jörg Flum and Martin Grohe. *Parameterized complexity theory*, volume 3. Springer Heidelberg, 2006.

[FHV13] Henning Fernau, Pinar Heggernes, and Yngve Villanger. A multivariate analysis of some DFA problems. In *Proceedings of LATA*, pages 275–286, 2013.

[Gal12] François Le Gall. Faster algorithms for rectangular matrix multiplication. In *FOCS*, pages 514–523, 2012.

[HKN12] Sepp Hartung, Christian Komusiewicz, and André Nichterlein. Parameterized algorithmics and computational experiments for finding 2-clubs. In *Parameterized and Exact Computation*, pages 231–241. Springer, 2012.

[HP98] Xiaohan Huang and Victor Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14(2):257–299, 1998.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[LMS11] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Known algorithms on graphs on bounded treewidth are probably optimal. In *SODA*, pages 777–789, 2011.

[NP85] Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26(2):415–419, 1985.

[PP12] Marcin Pilipczuk and Michał Pilipczuk. Finding a maximum induced degenerate subgraph faster than 2^n. In *Parameterized and Exact Computation*, pages 3–12. Springer, 2012.

[PW10] Mihai Pǎtraşcu and Ryan Williams. On the possibility of faster sat algorithms. In *SODA*, pages 1065–1075, 2010.

[RV13] Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *STOC*, pages 515–524, 2013.

[Sto74] Larry J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory*. PhD thesis, MIT, 1974.

[Var82] Moshe Y. Vardi. The complexity of relational query languages. In *STOC*, pages 137–146, 1982.

[Vas12] Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *STOC*, pages 887–898, 2012.

[WY14] Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *SODA*, pages 1867–1877, 2014.