Tractable constructions of finite automata from monadic second-order formulas

Bruno Courcelle, Irène Durand

Université Bordeaux-1, LaBRI, CNRS
351, Cours de la Libération
33405, Talence cedex, France
courcell@labri.fr ; idurand@labri.fr

It is well-known from [DF, FG, CMR] that the model-checking problem for MSO logic on graphs is fixed-parameter tractable (FPT) with respect to tree-width and clique-width. The proof uses tree-decompositions (for tree-width as parameter) and k-expressions (for clique-width as parameter; see below), and the construction of a finite tree-automaton from an MSO sentence, expressing the property to check.

These two points are difficult : although tree-width $\leq k$ can be checked in linear time, the corresponding algorithm (by Bodlaender, see [DF]) is not practically usable. The situation is similar for bounded clique-width (see [HliOum]). Graphs can be given *with* their decompositions witnessing tree-width or clique-width $\leq k$, but another difficulty arises : the automata to be constructed are much too large and computations abort by lack of memory space. This is actually unavoidable if one wants an algorithm taking as input any MSO sentence (see, e.g., [FriGro]). One possibility is to forget the idea of implementing the general theorem, and to work directly on particular problems: see [G1,G2] or [GH]. Another one, explored here (also in [KL] in a different way) consists in finding fragments of MSO logic having an interesting expressive power, and for which automata constructions (or other constructions) are tractable.

What we propose is based on the following ideas :
(1) Do not alternate quantifiers and do not determinize automata.
(2) Write MSO formulas with Boolean set terms (see below definitions).
(3) Precompile basic graph properties into "small" finite automata.

We do not capture all MSO graph properties, but we can formalize in this way coloring and partitionning problems, and also some domination problems to take a few examples. We only discuss graphs of bounded clique-width, but the ideas extend to graphs of bounded tree-width and MSO formulas with edge set quantifications. The problems considered successfully in [BK] use automata with smaller numbers of states than what we need.

Definition 1 : Clique-width and k-expressions.
Graphs are finite, simple, directed, loop-free. Each vertex has a label in $[k] := \{1, ..., k\}$. The operations on graphs are $\oplus$, the union of disjoint graphs, the unary edge-addition $\overrightarrow{add}_{a,b}$ that adds the missing edges from every vertex labelled a to every vertex labelled b, the relabelling $relab_{a\to b}$ that changes a to b (with $a \neq b$ in both cases). The constant a denotes one vertex (with no edge) labelled by $a \in [k]$.Let F_k be the set of these operations and constants. Every term t in $T(F_k)$ called a *k-expression* defines a graph $G(t)$ with vertex set equal to the set of occurrences of constants in t. A graph has *clique-width* at most k if it is defined by some t in $T(F_k)$.

As a logical structure, a graph is defined as $\langle V_G, edg_G\rangle$ where V_G is the vertex set and edg_G the binary relation that describes edges.

Definition 2 : **The set of terms representing a graph property.**
Let $P(X_1, ..., X_n)$ be a property of sets of vertices $X_1, ..., X_n$ of a graph G denoted by a term t in $T(F_k)$. Examples are : $E(X, Y)$: there is an edge from some x in X to some y in Y; $H(X, Y)$: for every x in X, there is an edge from some y in Y to x ; $Path(X, Y)$: X has two vertices linked by a path in $G[Y]$, the subgraph of G induced by Y and $Conn(X)$: $G[X]$ is connected.

Let $F_k^{(n)}$ be obtained from F_k by replacing each constant a by the constants (a,w) where $w \in \{0,1\}^n$. For fixed k, let $L_{P,(X_1,...,X_n),k}$ be the set of terms t in $T(F_k^{(n)})$ such that $P(A_1,...,A_n)$ is true in $G(t)$, where A_i is the set of vertices which are occurrences of constants (a,w) where the i-th component of w is 1. Hence t in $T(F_k^{(n)})$ defines a graph $G(t)$ and an assignment of sets of vertices to the set variables $X_1,...,X_n$.

Definition 3 : $\exists MSO(\mathcal{P})$ **sentences**

We let $\mathcal{P}$ be a set of basic graph properties like those of Definition 2 together with the atomic formulas $X_1 \subseteq X_2$, $X_1 = \emptyset$, $Sgl(X_1)$ (the last one means that X_1 denotes a singleton set). Let $\{X_1,...,X_n\}$ be a set of set variables. A *Boolean set term* is a term over these variables, $\cap, \cup$ and complementation (example below). A $\mathcal{P}$ *-atomic formula* is a formula of the form $P(S_1,...,S_m)$ where $S_1,...,S_m$ are Boolean set terms and P belongs to $\mathcal{P}$. An $\exists MSO(\mathcal{P})$ sentence is a sentence of the form $\exists X_1,...,X_n.\varphi$ where φ is a positive Boolean combination of $\mathcal{P}$ -atomic formulas.

Examples 4 :

We give some examples of $\exists MSO(\mathcal{P})$ sentences.

(1) The property of *p-vertex colorability* can be expressed as follows :

$$\exists X_1,...,X_p.(Part(X_1,...,X_p) \wedge St(X_1) \wedge \cdots \wedge St(X_p)),$$

where $Part(X_1,...,X_p)$ expresses that $X_1,...,X_p$ define a partition of the vertex set and $St(X_i)$ expresses that X_i is *stable*, i.e., that the induced graph $G[X_i]$ has no edge.

A p-vertex coloring defined by $X_1,...,X_p$ is *acyclic* if furthermore, each induced graph $G[X_i \cup X_j]$ is acyclic (is a forest). These properties are thus in $\exists MSO(\mathcal{P})$ if we let in $\mathcal{P}$ the properties $Part(X_1,...,X_p)$, $St(X_1)$ and $NoCycle(X_1)$ expressing that $G[X_1]$ is acyclic.

(2) *Minor inclusion.* That a graph G contains a fixed simple loop-free graph H with vertex set $\{v_1,...,v_p\}$ as a minor, can be expressed by the sentence μ :

$$\exists X_1,\cdots,X_p.(Disjoint(X_1,...,X_p) \wedge Conn(X_1) \wedge \cdots \wedge Conn(X_p) \wedge \cdots Link(X_i,X_j) \wedge \cdots)$$

where $Disjoint(X_1,...,X_p)$ expresses that $X_1,...,X_p$ are pairwise disjoint, $Conn(X_i)$ expresses that $G[X_i]$ is connected and $Link(X_i,X_j)$ expresses that there exists an edge between a vertex of X_i and one of X_j; in μ, there is one formula $Link(X_i,X_j)$ for each edge $\{v_i,v_j\}$ of H.

(3) *Constrained domination.* The sentence $\exists X_1.(P(X_1) \wedge Dom(\overline{X_1},X_1))$ expresses that there exists a set X_1 satisfying a property P and which also *dominates all other vertices.* The formula $Dom(Y,X)$ expresses that every vertex of Y is linked by an edge to some vertex of X.

(4) Many vertex partitionning problems considered by Rao in [Rao] can be expressed in this way.

Definition 5 : From $\exists MSO(\mathcal{P})$ **sentences to "reasonable sized" automata.**

Let us assume that for each basic property $P(X_1,...,X_m)$ we have constructed a finite automaton $\mathcal{A}_{P,(X_1,...,\ X_m),k}$ that accepts the set $L_{P,(X_1,...,X_m),k}$.

Claim 6 : For set terms $S_1,...,S_m$ over $\{X_1,...,X_n\}$, the set of terms $L_{P(S_1,...,S_m),(X_1,...,X_n),k}$ is $h^{-1}(L_{P,(X_1,...,X_m),k})$ where h is the *alphabetic homomorphism* : $T(F_k^{(n)}) \rightarrow T(F_k^{(m)})$ that replaces a constant symbol (a,w) for $w \in \{0,1\}^n$ by (a,w') for some $w' \in \{0,1\}^m$ and does not modify the nonnullary function symbols. We give an example : consider $P(X_1)$, $n=3$ and $S = X_1 \cup \overline{X_3}$. Then $L_{P(S),(X_1,X_2,X_3),k} = h^{-1}(L_{P,(X_1),k})$ where $h(1x0) = h(1x1) = 1$, $h(0x0) = 1, h(0x1) = 0$, for every $x = 0,1$, i.e., $h(x_1,x_2,x_3) = x_1 \vee \neg x_3$.

Claim 7 : From an automaton $\mathcal{A}_{P,(X_1,...,X_m),k}$ that accepts $L_{P,(X_1,...,X_m),k}$ one gets an automaton $\mathcal{A}_{P(S_1,...,S_m),(X_1,...,X_n),k}$ with same number of states that accepts $L_{P(S_1,...,S_m),(X_1,...,X_n),k}$. If $\mathcal{A}_{P,(X_1,...,X_m),k}$

is deterministic, then $\mathcal{A}_{P(S_1,...,S_m),(X_1,...,X_n),k}$ is also deterministic.

Claim 8 : If φ is a positive Boolean combination of $\mathcal{P}$ -atomic formulas $\alpha_1, ..., \alpha_d$ for which we have constructed non-deterministic (resp. deterministic) automata $\mathcal{A}_1, ..., \mathcal{A}_d$ with respectively $N_1, ..., N_d$ states, one can construct a "product" non-deterministic (resp. deterministic) automaton for φ with $N_1 \times ... \times N_d$ states (perhaps less after deletion of useless states).

Claim 9 : If θ is the sentence $\exists X_1, ..., X_n.\varphi$, and we have constructed an automaton $\mathcal{A}$ for φ, we obtain one, usually not deterministic even if $\mathcal{A}$ is, for θ with same number of states, by the classical "projection" that deletes the Boolean sequences from the constant symbols of $F_k^{(n)}$.

Theorem 10 : Let $\mathcal{P}$ be a set of basic graph properties. For each $P \in \mathcal{P}$ and for each k, let a nondeterministic automaton $\mathcal{A}_{P,(X_1,...,X_m),k}$ with at most $N(k)$ states be known. For every sentence θ of the form $\exists X_1, ..., X_n.\varphi$ where φ is a positive Boolean combination of d $\mathcal{P}$-atomic formulas, a non-deterministic automaton $\mathcal{A}_{\theta,\varepsilon,k}$ (over the signature F_k) with at most $N(k)^d$ states can be constructed.

In many cases (see below) deterministic automata for the properties of $\mathcal{P}$ are such that $N(k) = 2^{O(k^2)}$. The only nondeterministic transitions of $\mathcal{A}_{\theta,\varepsilon,k}$ are those associated with the constants. Then, with these hypotheses and the notation of Theorem 6:

Theorem 11 : For every term t in $T(F_k)$, one can decide in time $O(\mid t \mid .N(k)^{2d})$ if the graph $G(t)$ satisfies θ.

Proof : One can decide in time $O(\mid t \mid .N^2)$ if a nondeterministic automaton with N states over a binary signature and nondeterministic transitions limited to constants accepts a term t. In this computation, one considers θ as fixed, and the time to fire a transition constant.

Application 12: Some basic graph properties and their automata; experiments[1]

We classify graph properties in terms of the numbers of states $N(k)$ of deterministic automata that check them.

Polynomial-sized automata.

The automata for $X_1 \subseteq X_2$, $X_1 = \emptyset$, $Sgl(X_1)$ have no more than 4 states. For $E(X_1, X_2)$ we can build an automaton with $k^2 + k + 2$ states.

Single-exponential sized automata.

For $Part(X_1, ..., X_p)$, $Disjoint(X_1, ..., X_p)$ and $St(X_i) : 2^k$ states.

For $Link(X_i, X_j)$ and $Dom(X_i, X_j) : 2^{2k}$ states.

For $Path(X_1, X_2)$, we have constructed a (non-minimal) automaton with 2^{k^2} states. Its minimization (with the software ATUOWRITE [Dur]) has given the results shown in the following table.

$Path(X_1, X_2)$		
cwd	A	min(A)
2	25	12
3	214	128
4	3443	2197

For $NoCycle(X_1) : 2^{O(k^2)}$ states.

For d-vertex coloring, we get, by Theorem 10, a nondeterministic automaton with 2^{kd} states.

[1] The tables contain the number of states of the considered automata. The symbol ∞ means that the computation did not finish but did not run out of memory.

We have been able to construct minimal deterministic automata for the following values of (k, d) shown in the following table.

Vertex coloring				
cwd	d	A	det(A)	min(A)
2	2	16	12	8
2	3	64	37	14
2	4	256	96	23
2	5	1024	213	36
3	2	64	406	56
3	3	512	∞	

Double-exponential sized automata.

For checking connectivity, one can build a deterministic automaton with 2^{2^k} states. If $k = 2p + 1$, the corresponding minimal deterministic automaton has more than 2^{2^p} states. However, for connectivity of graphs of degree at most d, which may be enough in practice, we can build a single-exponential sized automaton with $2^{O((dk)^2)}$ states.

For having a circuit, we get 9 states for $cwd = 2$ and 81 for $cwd = 3$ and the program runs out of memory for $cwd = 4$. For indegree at most 1, we get :

cwd	2	3	4	5
A	24	123	621	3120

Perspectives: To make these constructions usable, we will not try to tabulate and minimize automata, but rather, we will describe their transitions by clauses. We will compute a transition each time it is necessary for running the automaton on a given term.

References

[BK] D.Basin, N. Klarlund, Automata based reasoning in hardware verification, *J. of Formal Methods in System Design*, 13 (1998) 255-288.

[CMR] B. Courcelle, J. A. Makowsky, U. Rotics, On the fixed parameter complexity of graph enumeration problems definable in monadic second-order logic. *Discrete Applied Mathematics* 108 (2001) 23-52

[DF] R. Downey et M. Fellows, *Parameterized complexity*, Springer-Verlag, 1999

[Dur] I. Durand, Autowrite: A Tool for Term Rewrite Systems and Tree Automata, *Electronic Notes TCS* 124(2005) 29-49

[FG] J. Flum, M. Grohe, *Parametrized complexity theory*, Springer, 2006.

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

[GH] R. Ganian, P. Hlineny, On Parse Trees and Myhill-Nerode-type Tools for handling Graphs of Bounded Rank-width.

Discrete Applied Maths, In press.

[G1] G. Gottlob, R. Pichler, F. Wei: Abduction with Bounded Treewidth: From Theoretical Tractability to Practically Efficient Computation. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, Chicago, July 2008, AAAI Press, 2008, pp. 1541-1546

[G2] G. Gottlob, R. Pichler, F. Wei: Monadic Datalog over Finite Structures with Bounded Treewidth, 2008, ArXiv, CoRR abs/0809.3140

[HliOum] P. Hlineny, S. Oum: Finding Branch-Decompositions and Rank-Decompositions. *SIAM J. Comput.* 38(2008) 1012-1032.

[KL] J. Kneis, A. Langer, A Practical Approach to Courcelle's Theorem, *Electronic Notes TCS*, 251 (2009) 65-81.

[Rao] M. Rao, MSOL partitioning problems on graphs of bounded treewidth and clique-width. *Theor. Comput. Sci.* 377(2007) 260-267.