Fly-automata for checking monadic second-order graph properties of graphs of bounded tree-width

Bruno Courcelle

Bordeaux University, LaBRI (CNRS laboratory)

Topics

Comparing clique-width to tree-width for sparse graphs

Fixed-parameter tractable (FPT) algorithms based on

graph decompositions + logic + automata on terms

Graph decompositions = tree structuring of graph in terms of "small" graphs and composition operations

Graph structure theory :

tree-decomposition for the Graph Minor Theorem,

modular decomposition for comparability graphs,

ad hoc decompositions for the Perfect Graph Theorem.

Algorithmic meta-theorems give FPT algorithms for parameters *tree-width* and *clique-width* based on graph decompositions; properties to check are expressed in *monadic second-order logic (MSO)*. (Definitions will be given soon).

Theorem : For each k, every MSO graph property P can be checked in (FPT) time O(f(k).n) where n = number of vertices, k = tree-width or clique-width of the input graph, given by a relevant decomposition. This decomposition is formalized by an algebraic term over operations that build graphs (generalizing concatenation of words).

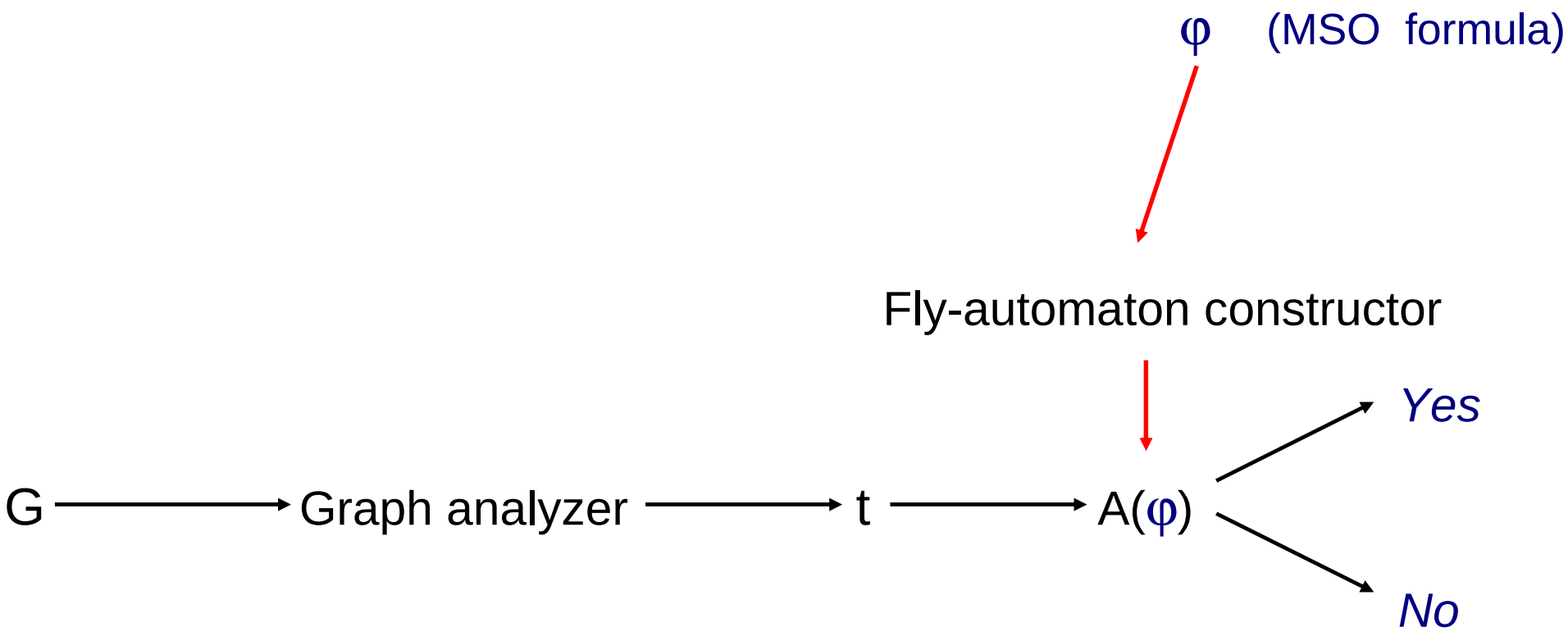
Method : From k and φ expressing P, one builds a finite automaton A(φ,k) to recognize the terms that represent decompositions of width at most k and define graphs satisfying P.

Difficulty : The *finite* automaton A(φ,k) is much too large as soon as $k \geq 2$: 2^(2^(…2^k)..)) states

(because of quantifier alternations)

To overcome this difficulty, we use fly-automata whose states and transitions are described and not tabulated. Only the transitions necessary for an input term are computed "on the fly". Sets of states can be infinite and fly-automata can compute values, *e.g.*, the number of p-colorings or of acyclic p-colorings of a graph. This is a theoretical view of dynamic programming.

The MSO meta-theorem through *fly-automata*



A(φ): *infinite fly-automaton*. The time taken by A(φ) is O(f(k).n) where k depends on the operations occurring in t and bounds the tree-width or clique-width of G.

Computations using fly-automata (by Irène Durand)

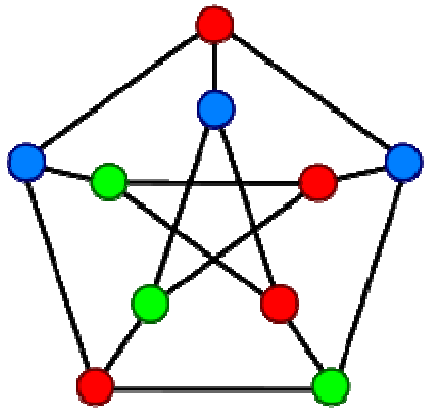
Number of 3-colorings of the 6 x 525 rectangular grid (of clique-width 8) in 10 minutes.

4-acyclic-colorability of the Petersen graph (clique-width 5) in 1.5 minutes.

(3-colorable but not acyclically;

red and **green** vertices

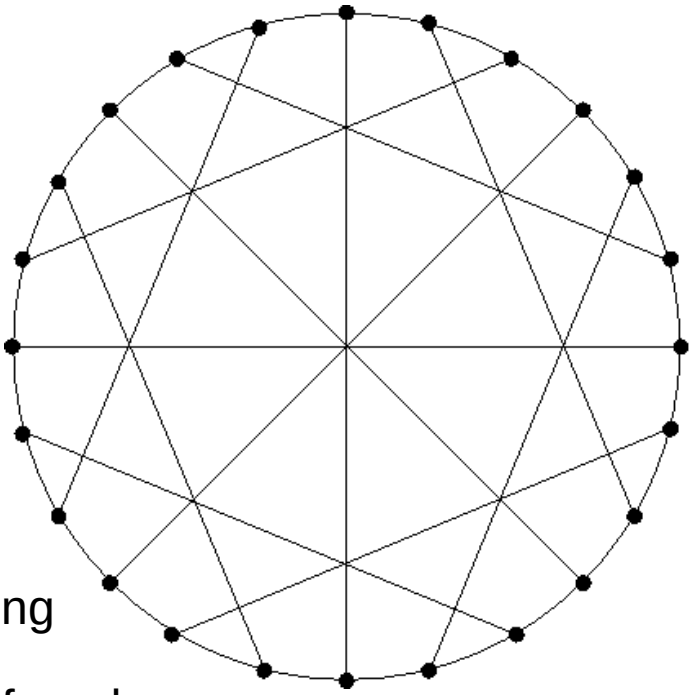
induce a cycle).



The McGee graph

is defined by a term

of size 99 and depth 76.



This graph is 3-acyclically colorable.

Checked in 40 minutes.

Even in 2 seconds by enumerating the accepting runs, and stopping as soon as a success is found.

Definition 1 : Monadic Second-Order Logic

First-order logic extended with (quantified) variables denoting subsets of the domains.

A graph G is given by the logical structure

$(V_G , edg_G(.,.))$ = (vertices, adjacency relation)

Property P is MSO expressible : P(G) $\Leftrightarrow$ G != φ

MSO expressible properties : transitive closure, properties of paths, connectedness, planarity (via Kuratowski), p-colorability.

Examples : G is 3-colorable :

$$\exists X, Y \; (X \cap Y = \emptyset \; \wedge$$
$$\forall u,v \; \{ \; edg(u,v) \Rightarrow$$
$$[(u \in X \Rightarrow v \notin X) \wedge (u \in Y \Rightarrow v \notin Y) \wedge$$
$$(u \notin X \cup Y \Rightarrow v \in X \cup Y) \;]$$
$$\} \;)$$

G is not connected :

$$\exists Z \; (\exists x \in Z \; \wedge \; \exists y \notin Z \; \wedge \; (\forall u,v \; (u \in Z \; \wedge \; edg(u,v) \Rightarrow v \in Z) \;)$$

Planarity is MSO-expressible (no minor K_5 or $K_{3,3}$).

Edge quantifications (MSO_2 graph properties)

If $G = (V_G , edg_G(.,.))$, its *incidence graph* is defined as

$Inc(G) := (V_G \cup E_G , inc_G(.,.))$ with

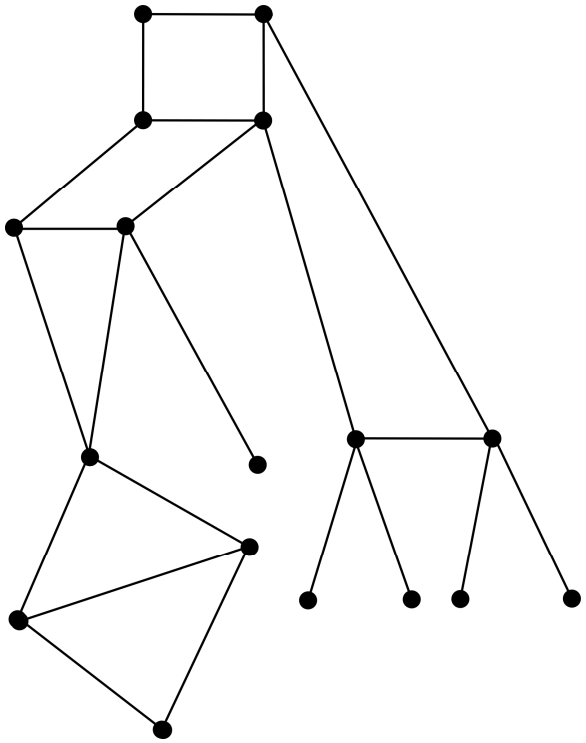
$inc_G(u,e) \Leftrightarrow$ u is the *tail* of edge e,

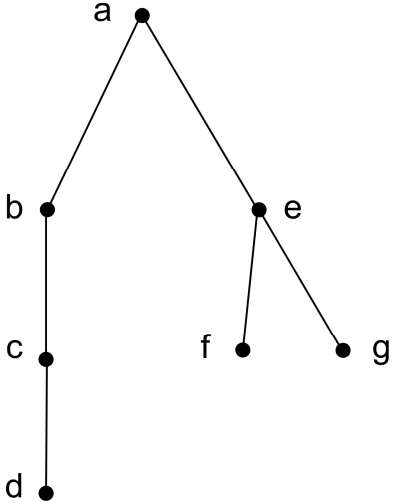
$inc_G(e,u) \Leftrightarrow$ u is the *head* of edge e. (G is directed).

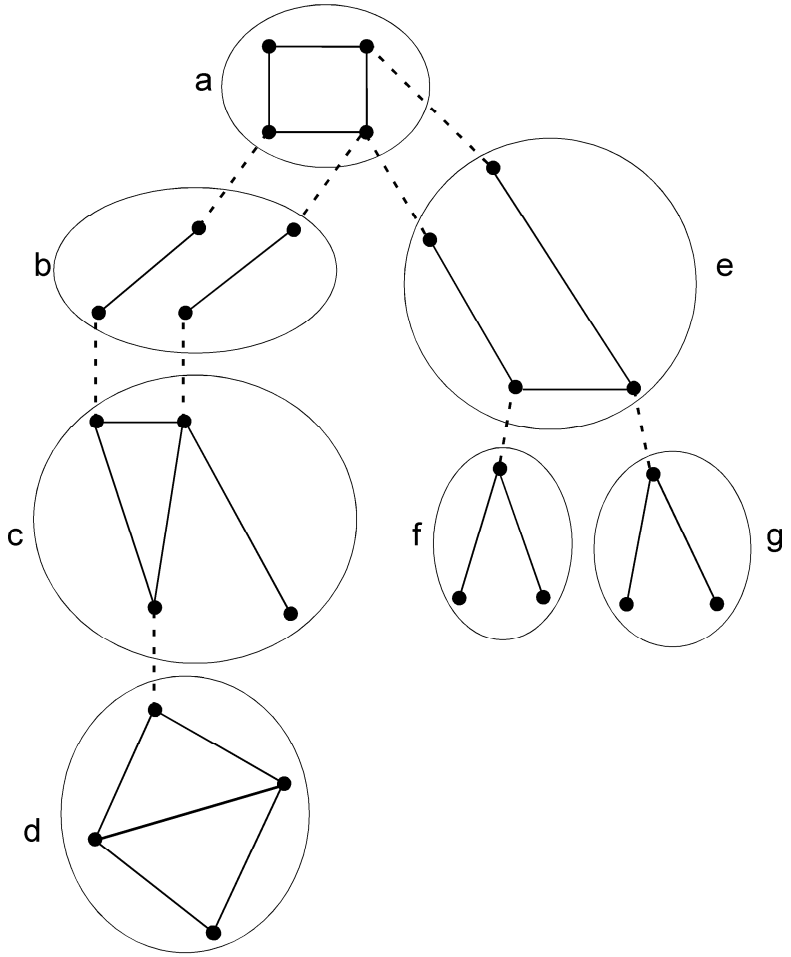
MSO formulas over Inc(G) can use quantifications on edges and express more properties than those over G. MSO_2 graph properties of G are expressed by MSO formulas over Inc(G).

That G is isomorphic to some $K_{p,p}$ is MSO_2 expressible but not MSO expressible.

Definition 2 : Tree-decomposition, tree-width (denoted by twd(G)).







Graph G

a decomposition of G of width 3 (= 4-1)

Definition 3 : Clique-width (denoted by cwd(G)).

Defined from graph operations. Graphs are simple, directed or not, vertices are labelled by *a,b,c, ...* . A vertex labelled by *a* is an *a-vertex*.

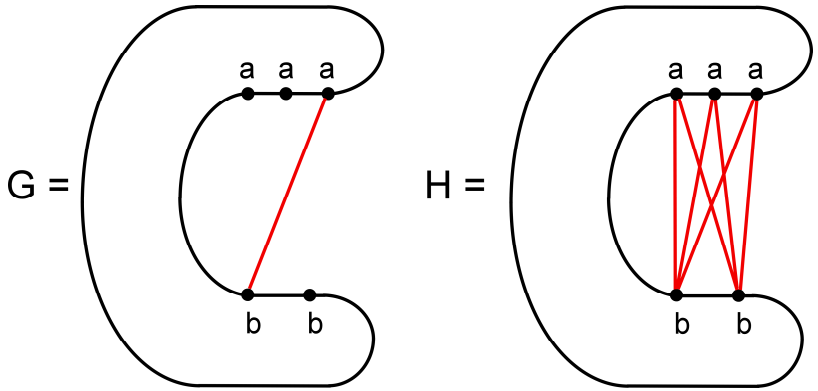
One binary operation: disjoint union : $\oplus$

Unary operations: edge addition denoted by $Add_{a,b}$

$Add_{a,b}$(G) is G augmented with undirected edges between every *a*-vertex and every *b*-vertex.

The number of added edges depends on the argument graph.

Directed edges are defined similarly.



H = $Add_{a,b}$(G) ; only 5 new edges added

Vertex relabellings :

$Relab_{a \to b}(G)$ is G with every *a*-vertex is made into a *b*-vertex

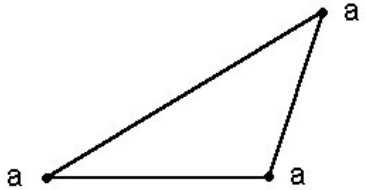
Basic graphs : **a** , a vertex labelled by *a*.

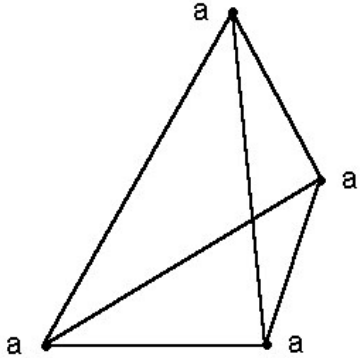
The clique-width of G (denoted by cwd(G)) is the smallest k such that G is defined by a term using k labels.

Example : Cliques have unbounded tree-width and clique-width 2.

K_n is defined by t_n where $t_{n+1} =$

$Relab_{b \to a}(Add_{a,b} (t_n \oplus \mathbf{b}))$





Clique-width compared to tree-width [sparse graphs]

For all graphs G :

$cwd(G) \leq 2^{2.twd(G)+2} + 1$ ($\leq 3.2^{twd(G)-1}$ if G is undirected).

For incidence graphs H = Inc(G) :

$cwd(H) \leq 2.\ twd(G)+4$ ($\leq twd(G) + 3$ if G is undirected).

For planar graphs :

$cwd(G) \leq 32.\ twd(G) - 24$ ($\leq 6.twd(G) - 2$ if G is undirected).

For graphs of degree $\leq$ d : $cwd(G) = O(twd(G))$.

For hereditary average degree d : $cwd(G) = O(twd(G)^{2.d})$.

Meta-theorems : FPT time f(wd(G)).n

(1) MSO properties of graphs of bounded cwd,

(2) MSO_2 properties of graphs of bounded twd.

Notes: - MSO expressible $\Rightarrow$ MSO_2 expressible and

bounded twd $\Rightarrow$ bounded cwd.

(2) reduces to (1) because MSO_2 on G = MSO on Inc(G)

and cwd(Inc(G)) = O(twd(Inc(G))) = O(twd(G))

avoiding the exponential jump cwd(G) = $2^{O(twd(G))}$

- twd(G) = O(cwd(Inc(G)): MSO_2 checking via incidence graphs "only work" for bounded tree-width.

Automata for checking MSO properties

We want to check a property P(G) of a graph G = G(t) given by a term t that is either a *clique-width term* or a term representing a tree-decomposition.

We can construct an automaton A(P) to check that, for given term t that G(t) satisfies P :

either "directly" from our understanding of P and graph operations,

or by an induction on the structure of an MSO formula expressing P.

We need automata for atomic formulas. A conjunction is handled by a product of two automata. An existential quantification introduces non-determinism, but automata are run deterministically : this is the main source of huge sizes. Negation needs determinization.

Definition 4 : Fly-automaton (FA)

$A = < F, Q, \delta,$ Out $>$

F : finite or countable (effective) set of operations,

Q : finite or countable (effective) set of states (integers, pairs of integers, finite sets of integers: states can be encoded as finite words, integers in binary),

Out : Q → *D* (a set of finite words), computable.

δ : computable (bottom-up) transition function

Nondeterministic case : δ is *finitely multi-valued*. Determinization works.

An FA defines a computable function : T(F) → *D* , a decidable property if D = {*True, False*}.

Theorem [B.C & I.D.] : For each MSO property P, one can construct a single infinite FA over F that recognizes the terms t in T(F) such that P(G(t)) holds.

Computation time is f(k).n, n = size of term, k = number of labels in t.

Consequence : The same automaton (the same model-checking program) can be used for graphs of any clique-width.

Application to incidence graphs and MSO_2 properties (edge quantifications) of graphs of bounded tree-width.

1) From of a tree-decomposition T of G of width k, we construct a term t for Inc(G) of "small" clique-width k+3 (2k+4 if G directed).

2) We translate an MSO_2 formula φ for G into an MSO formula θ for Inc(G).

3) The corresponding automaton A(θ) takes term t as input. But an atomic formula edg(X,Y) of φ is translated into $\exists U.\ inc(X,U) \wedge inc(U,Y)$ in θ which adds one level of quantification.

The automaton A(θ) remains manageable.

For certain properties P, for example connectedness, directed cycle, outdegree < p, we have $P(G) \Leftrightarrow P(Inc(G))$.

The automaton for graphs G defined by clique-width terms can be used for the clique-width terms that define the graphs Inc(G).

oOo

Why automata running on clique-width terms rather than on terms representing tree-decompositions ? They are simpler to construct (and smaller). It is practically useful to translate tree-decompositions of sparse graphs (incidence graphs, planar graphs, graphs of bounded degree) into clique-width terms.

Conclusion

In most cases, we get XP or FPT dynamic programming algorithms, that can be obtained independently.

These algorithms are based on fly-automata, that can be quickly constructed from logical descriptions → *flexibility*.

These constructions are implemented. Tests have been made for colorability and connectedness problems.

Thank you for suggesting interesting problems that could fit in this framework.