Monadic second-order logic

and the verification of graph properties

Bruno Courcelle

Université Bordeaux 1, LaBRI
and
Institut Universitaire de France

References : Chapter 5 in : Handbook of graph grammars vol.1, 1997,

Book in progress, Articles with J. Makowsky, U. Rotics, P. Weil, S. Oum, A. Blumensath ;

See : http://www.labri.fr/perso/courcell/ActSci.html

Monadic second-order logic : expression of properties, queries, optimization functions, number of configurations.

Mainly useful for tree-structured graphs (Second-order logic is useless)

Two types of questions :

Checking $G \models \varphi$ for fixed formula φ, given G , (*Model checking*)

Deciding if $\exists G,\ G \in \mathcal{C}, G \models \varphi$ for fixed $\mathcal{C}$, given formula φ.

Tools to be presented

Algebraic setting for tree structuring of graphs

Recognizability = finite congruence $\equiv$ inductive computability

$\equiv$ finite deterministic automaton on terms

Fefermann-Vaught : MS definability $\Rightarrow$ recognizability.

Verification of graph properties.

History : Confluence of 4 independent research directions, now intimately related :

1. Polynomial algorithms for NP-complete and other hard problems on particular classes of graphs, and especially hierarchically structured ones : series-parallel graphs, cographs, partial k-trees, graphs or hypergraphs of tree-width < k, graphs of clique-width < k.
2. Excluded minors and related notions of forbidden configurations (matroid minors, « vertex-minors »).
3. Decidability of Monadic Second-Order logic on classes of finite graphs, and on infinite graphs.
4. Extension to graphs and hypergraphs of the main concepts of Formal Language Theory : grammars, recognizability, transductions, decidability questions.

Summary

1. Introduction

Extension of Formal Language Theory notions

2. Context-free sets defined by equation systems.
3. The graph algebras HR and VR. Tree-width.

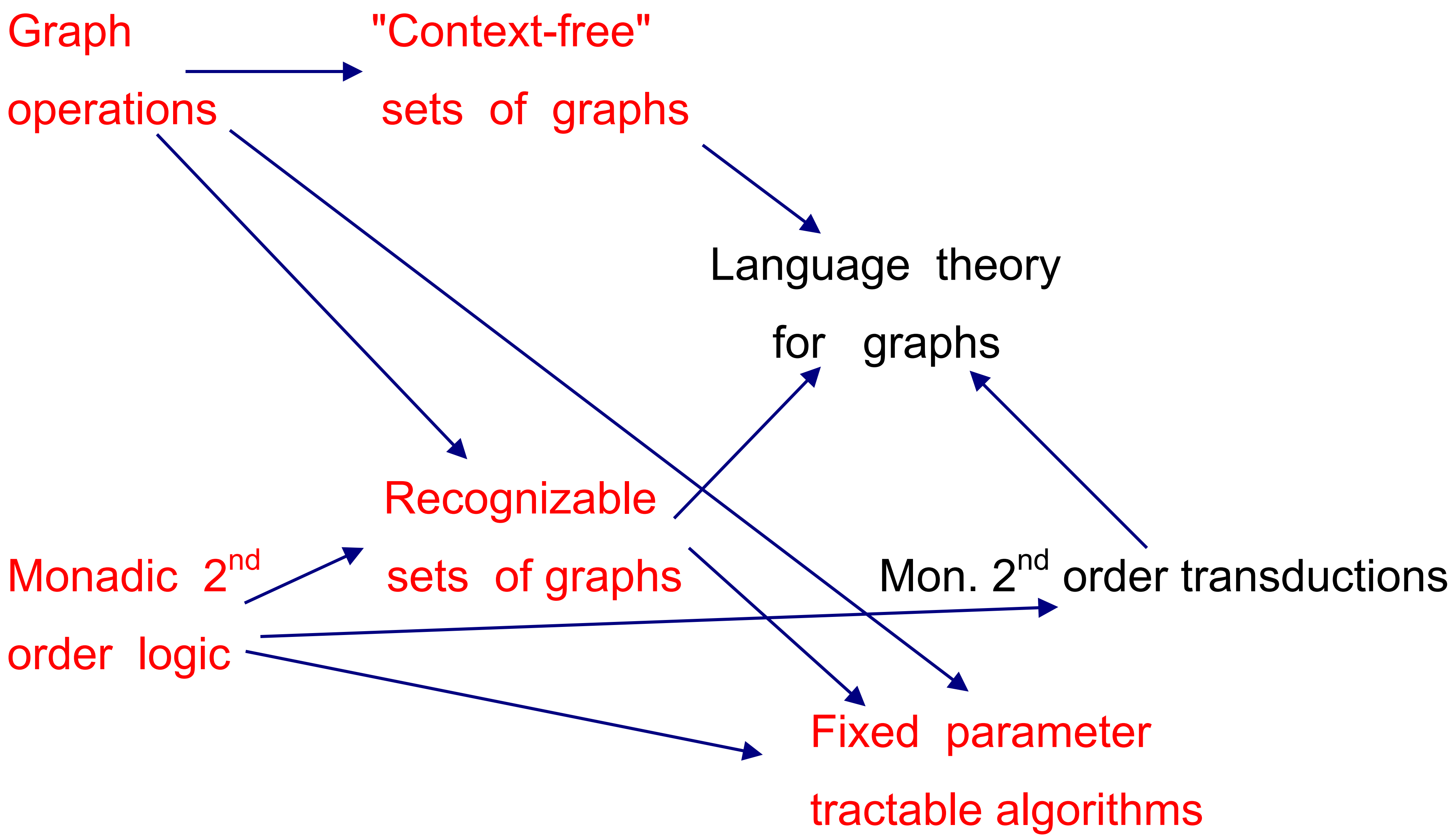
Algorithmic applications :

4. Inductive computations and recognizability; fixed-parameter tractable algorithms.
5. Monadic second-order logic defines inductive properties and functions

Formal language theory extended to graphs

6. Closure and decidability properties ; generation of classes of graphs by monadic second-order transductions.
7. Graph rewriting.

Introduction : An overview chart :



Key concepts of FLT and their extensions

Languages	*Graphs*
Algebraic structure : monoid $(X^*,*,\varepsilon)$	Algebras based on graph operations : $\oplus$, $\otimes$, // quantifier-free definable operations Algebras : HR, VR
Context-free languages : Equational subsets of $(X^*,*,\varepsilon)$	Equational sets of the algebras HR, VR
Regular languages : Finite automata $\equiv$ Finite congruences $\equiv$ Regular expressions $\equiv$	Recognizable sets of the algebras HR, VR defined by congruences
$\equiv$ Monadic Second-order definable sets of words or terms	$\cup$ Monadic Second-order definable sets of graphs
Rational and other types of transductions	Monadic Second-order transductions

Equational (context-free) sets

Equation systems = Context-Free (Graph) Grammars
in an algebraic setting

In the case of words, the set of context-free rules

$$S \to a\,S\,T\ ; \quad S \to b\ ; \quad T \to c\,T\,T\,T\ ; \quad T \to a$$

is equivalent to the system of two set equations:

$$\mathit{S} = a\,\mathit{S}\,\mathit{T} \quad \cup \quad \{ b \}$$

$$\mathit{T} = c\,\mathit{T}\,\mathit{T}\,\mathit{T} \quad \cup \quad \{ a \}$$

where S is the language generated by S (idem for T and T).

For graphs (or other objects) we consider systems of equations like:

$$S = f(k(S), T) \qquad \cup \{ b \}$$

$$T = f(T, f(g(T), m(T))) \cup \{ a \}$$

where f is a binary operation, g, k, m are unary operations on graphs,

a, b denote basic graphs (up to isomorphism).

An *equational set* is a component of the least (unique) solution of such an equation system. This is well-defined in any algebra.

Logical expression of graph properties

Cf. Descriptive complexity, theory of data bases.

A graph G can be given as a logical structure

$$< V_G , edg_G(.,.) >.$$

Typical properties expressible by First-order formulas :

G has no loop, or has degree at most 5, or has indegree at most 3.

G has an induced subgraph isomorphic to a fixed finite graph (useful for graph rewriting systems)

Such properties are characterized (Gaifman) as *local properties*.

Monadic Second-Order (MS) Logic

= First-order logic on power-set structures

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

MS properties : transitive closure, properties of paths, connectivity,

planarity (via Kuratowski, uses connectivity), k-colorability.

Examples of formulas for $G = \langle V_G , edg_G(.,.) \rangle$, undirected

Non connectivity :

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

2-colorability (i.e. G is bipartite) :

$\exists X \,(\forall u,v\, (u \in X \wedge edg(u,v) \Rightarrow v \notin X) \wedge \forall u,v\, (u \notin X \wedge edg(u,v) \Rightarrow v \in X))$

Edge set quantifications

Provably more powerful.

Incidence graph of G undirected, $\mathrm{Inc}(G) = < V_G \cup E_G, \mathrm{inc}_G(.,.) >$.

$\mathrm{inc}_G(v,e) \Leftrightarrow$ v is a vertex of edge e.

Monadic second-order (MS_2) formulas written with inc can use quantifications on sets of edges.

Existence of a Hamiltonian circuit is expressible by an MS_2 formula, but not by an MS formula.

Theorem : MS_2 formulas are no more powerful than MS formulas :
for graphs of degree $\leq d$, or of tree-width $\leq k$,
or for planar graphs, or for graphs without some fixed H as a minor,
or graphs of average degree $\leq k$ (uniformly k-sparse).

The two types of questions we would like to solve :

1) Checking $G \models \varphi$ for fixed formula φ, given G , (Model checking)

Polynomial time $O(n^s)$ for each first-order formula with s variables.

Linear for each first-order formula on graphs of bounded degree.

NP-complete problems (3-coloring) can be expressed by MS formulas.

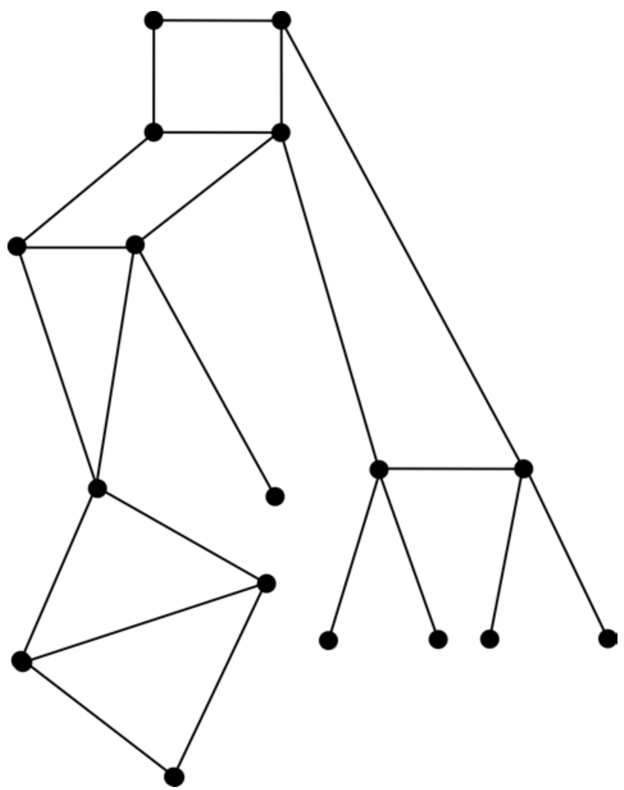
Linear for each MS formula on graphs of bounded tree-width.

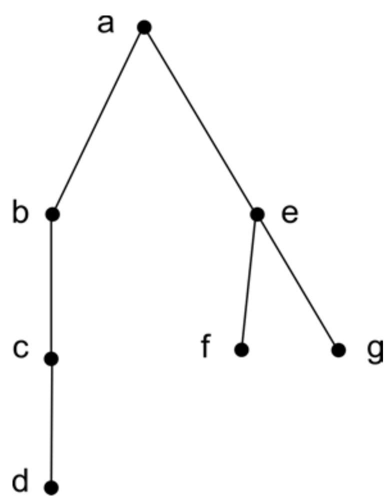
2) Deciding if $\exists$ G, G $\in$ *C*, $G \models \varphi$ for fixed *C*, given formula φ.

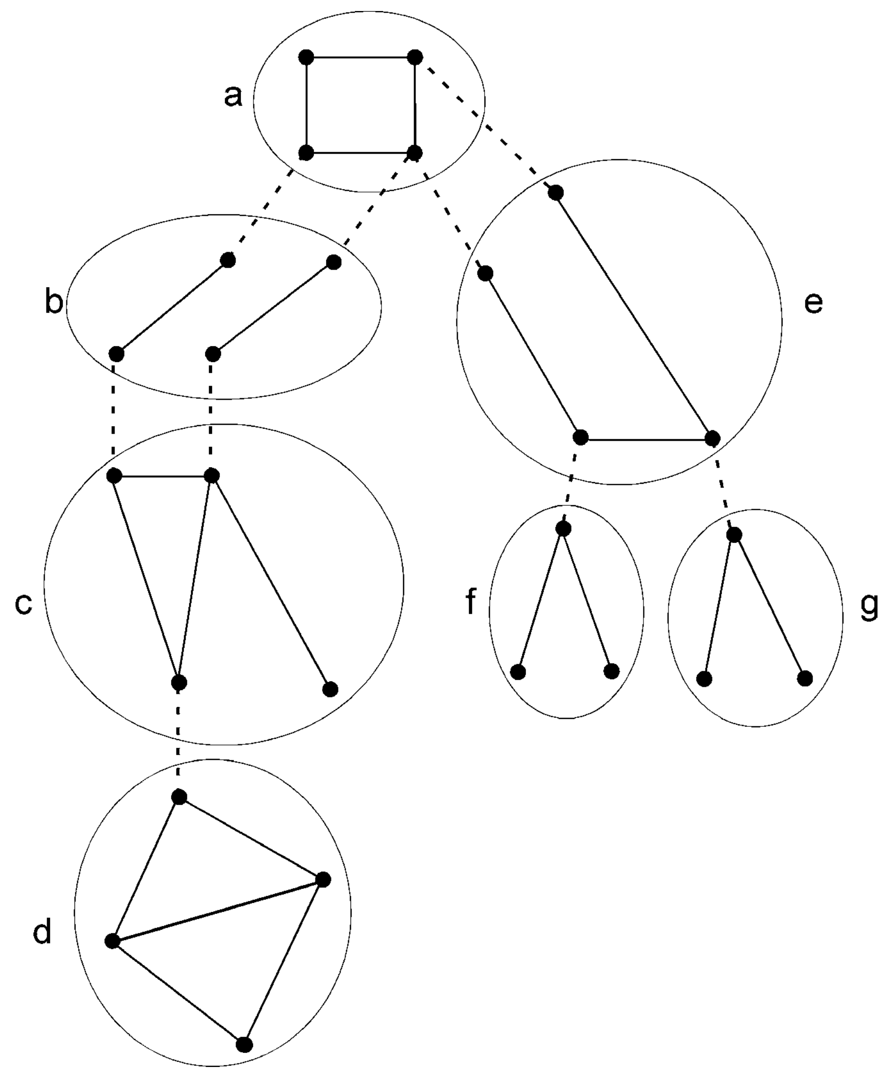
Even for first-order formulas, undecidable on the class of all finite graphs, and even of all finite planar graphs of degree at most 3.

Decidable for MS formulas with edge set quantifications on the class of graphs of tree-width $\leq$ k, and for each fixed k (untractable). Also on certain context-free sets of graphs (defined by HR grammars).

Tree-decompositions and their algebraic definition.







Tree-width

Tree-decomposition of width k : k+1 = maximum size of a box

Tree-width : twd(G) = minimum width of a tree-decomposition

Trees have tree-width 1,

K_n has tree-width n-1,

the n x n grid has tree-width n

Outerplanar graphs have tree-width at most 2.

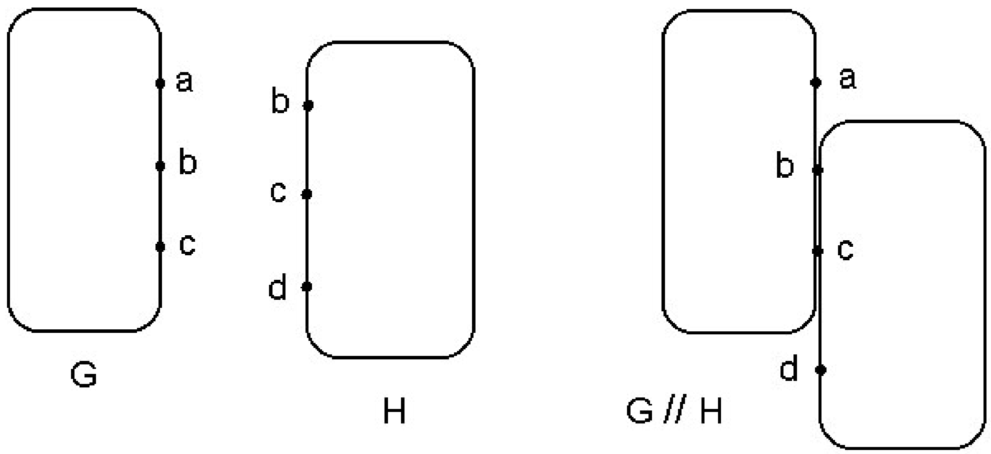
HR operations : Origin : **H**yperedge **R**eplacement hypergraph grammars ; associated complexity measure : tree-width

Graphs have distinguished vertices called *sources,* pointed to by labels from a set of size k : {*a, b, c, ..., h*}.

Binary operation(s) : *Parallel composition*

G // H is the disjoint union of G and H and sources with same label are fused.

(If G and H are not disjoint, one first makes a copy of H disjoint from G).



Unary operations : *Forget a source label*

Forget$_a$(G) is G without *a*-source : the source is no longer distinguished ; it is made "internal".

Source renaming :

Ren$_{a,b}$(G) exchanges source names *a* and *b* (replaces *a* by *b* if *b* is not the name of a source)

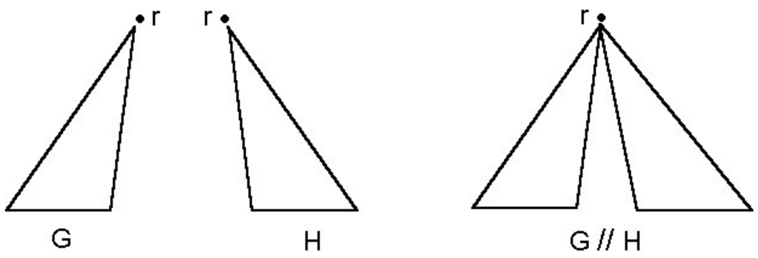
Nullary operations denote *basic graphs* : the connected graphs with at most one edge. *For dealing with hypergraphs one takes more nullary symbols for denoting hyperedges.*

More precise algebraic framework : a many sorted algebra where each finite set of source labels is a sort. The above operations are overloaded.

Proposition: A graph has tree-width $\leq$ k if and only if it can be constructed from basic graphs with $\leq$ k+1 labels by using the operations // , $Ren_{a,b}$ and $Forget_a$.

Example : Trees are of tree-width 1, constructed with two source labels, *r* (root) and *n* (new root):

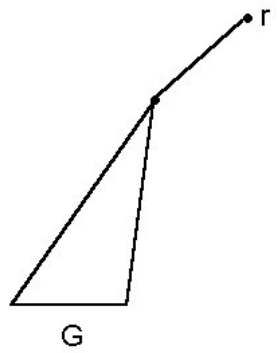
Fusion of two trees at their roots :



Extension of a tree by parallel composition with a new edge, forgetting the old root, making the "new root" as current root :

$$E = r \bullet\!\!-\!\!-\!\!-\!\!\bullet\ n$$

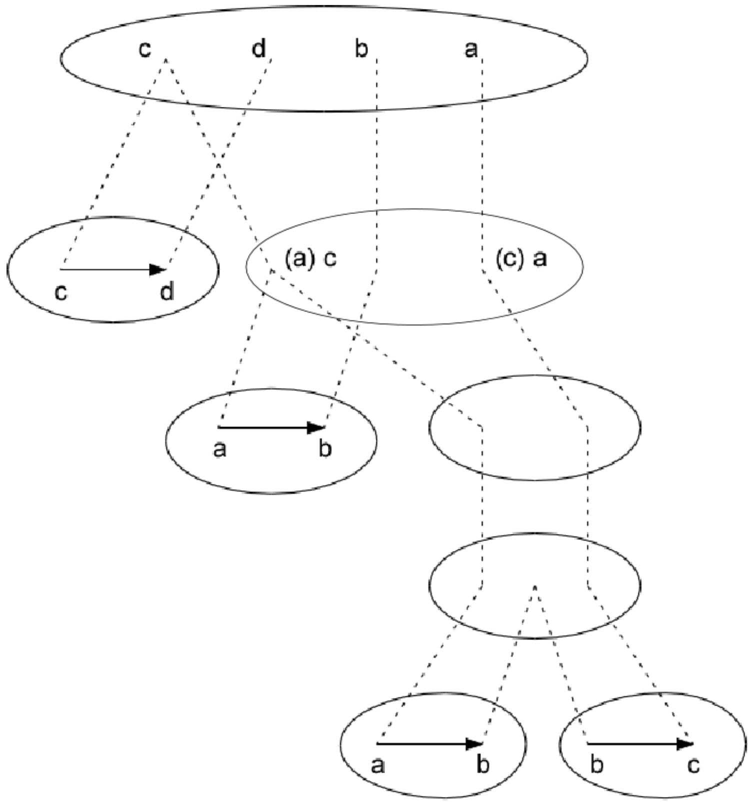
$$Ren_{\boldsymbol{n,r}}\,(Forget_{\boldsymbol{r}}\,(G\ //\ E))$$



From an algebraic expression to a tree-decomposition

Example : cd // $Ren_{a,c}$ (ab // $Forget_b$(ab // bc))

Constant ab denotes a directed edge from a to b.



The tree-decomposition associated with this term.

VR operations

Origin : **V**ertex **R**eplacement graph grammars

Associated complexity measure : clique-width, has no combinatorial characterization but is defined in terms of few very simple graph operations (whence easy inductive proofs).

Equivalent notion : rank-width (Oum and Seymour) with better structural and algorithmic properties.

Graphs are simple, directed or not.

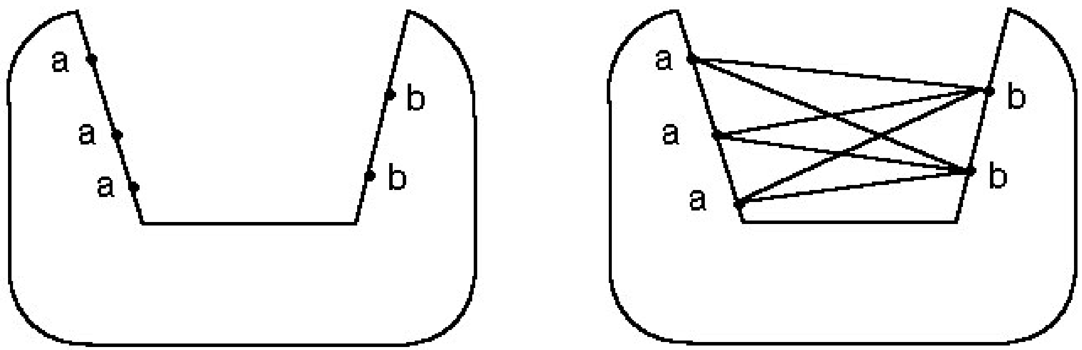
k labels : *a* , *b* , *c*, ..., *h*. Each vertex has one and only one label ;

a label p may label several vertices, called the *p-ports*.

One binary operation: *disjoint union* : $\oplus$

Unary operations: Edge addition denoted by *Add-edg*$_{a,b}$

Add-edg$_{a,b}$(G) is G augmented with (un)directed edges from every *a*-port to every b-port.



G *Add-edg*$_{a,b}$(G)

Vertex relabellings : *Relab*$_{a,b}$(G) is G with every vertex labelled by *a* relabelled into *b*

Basic graphs are those with a single vertex.

Definition: A graph G has clique-width $\leq$ k $\Leftrightarrow$ it can be constructed from basic graphs by means of k labels and the operations $\oplus$, *Add-edg*$_{a,b}$ and *Relab*$_{a,b}$

Its (exact) clique-width, *cwd*(G), is the smallest such k.

Proposition : (1) If a set of simple graphs has bounded tree-width, it has bounded clique-width, but not *vice-versa*.

(2) Unlike tree-width, clique-width is sensible to edge directions : Cliques have clique-width 2, tournaments have unbounded clique-width.

(3) a. Deciding "Clique-width $\leq$ 3" is a polynomial problem. (Habib *et al*.)

b. The complexity (polynomial or NP-complete) of "Clique-width = 4" is unknown.

c. It is NP-complete to decide for given k and G if $cwd(G) \leq k$. (Fellows *et al*.)

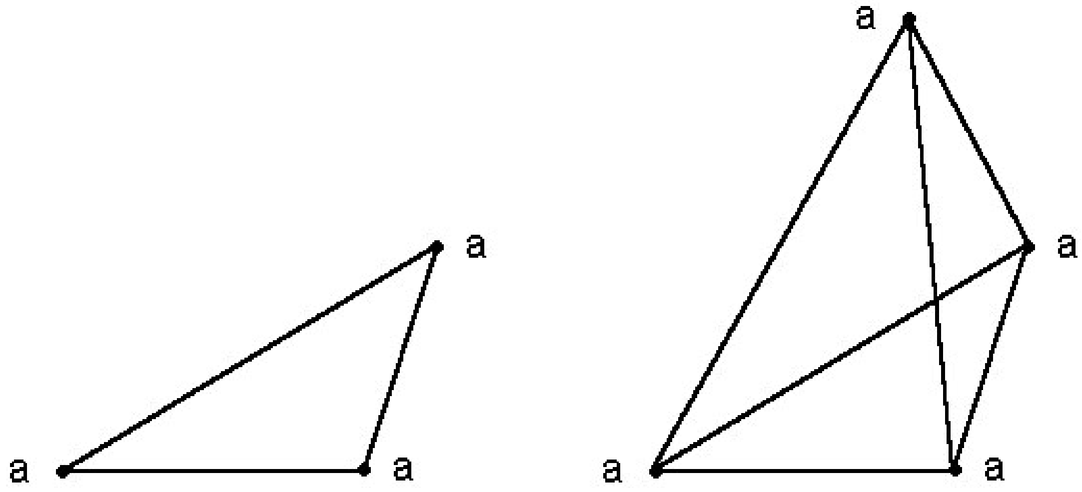
d. There exists a cubic approximation algorithm that for given k and G

answers (correctly) : either that $cwd(G) > k$,

or produces a clique-width algebraic term using 2^{24k} labels. (Oum)

This yields Fixed Parameter Tractable algorithms for many hard problems.

Example : Cliques have clique-width 2.



K_n is defined by t_n where t_{n+1} = $Relab_{b,a}(Add\text{-}edg_{a,b}(t_n \oplus \mathbf{b}))$

Example : Cographs are generated by $\oplus$ and $\otimes$ defined by :

$G \otimes H$ = $Relab_{b,a} (Add\text{-}edg_{a,b} (G \oplus Relab_{a,b}(H))$

= $G \oplus H$ with "all edges" between G and H.

Algorithmic applications

Fixed parameter tractability results

Theorem (B.C.) : For graphs of tree-width $\leq$ k ,

each monadic second-order property, (ex. 3-colorability),

each monadic second-order optimization function, (ex. distance),

each monadic second-order counting function, (ex. # of paths)

is evaluable in linear time with help of a result by Bodlaender (1996).

Similar results hold for clique-width bounded graphs and monadic second-order logic without edge set quantifications with cubic time because of the parsing step.

Applications to the decidability of logical formulas on classes of graphs

Theorem (B.C.) : The following problems can be solved by an algorithm :

Twd-MS2 : *Input :* k and a monadic second-order formula (with edge set quantifications).

Questions : Does the corresponding property hold for graphs of tree-width $\leq$ k ?

Does the corresponding property hold some graph of tree-width $\leq$ k ?

(tree-width $\leq$ k can be replaced by : in a given HR-equational set).

Cwd-MS : *Input* : k and a monadic second-order formula (without edge set quantifications).

Questions : Does the corresponding property hold for graphs of clique-width $\leq$ k ?

Does the corresponding property hold some graph of clique-width $\leq$ k ?

(clique-width $\leq$ k can be replaced by : in a a given VR-equational set).

(Limited) application : Checking the 4-Color Theorem for graphs of cwd $\leq$ k.

Inductive computations

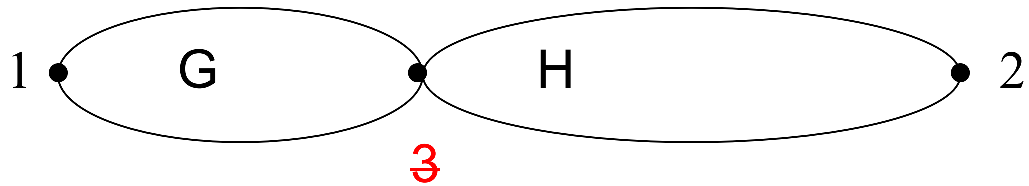
Example : *Series-parallel graphs*, defined as graphs with sources 1 and 2,

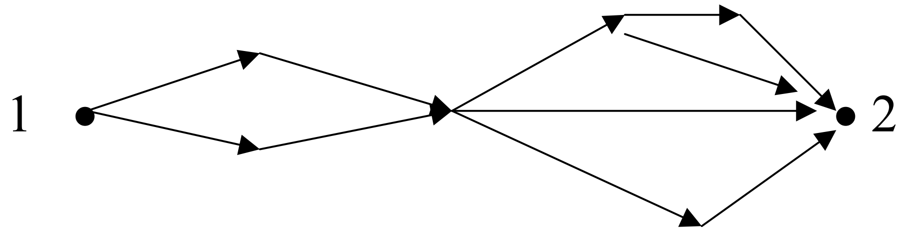
generated from e $= 1 \longrightarrow 2$ and the operations // (parallel-composition) and

series-composition defined from other operations by :

$$G \bullet H = Forget_3(Ren_{2,3}(G) // Ren_{1,3}(H))$$

Example :





Inductive proofs :

1) G, H connected implies : G//H and $G \bullet H$ are connected, (induction)

e is connected (basis) :

$\Rightarrow$ All series-parallel graphs are connected.

2) It is not true that :

G and H planar implies : G//H is planar (K_5 = H//e).

A stronger property for induction :

G has a planar embedding with the sources in the same "face"

$\Rightarrow$ All series-parallel graphs are planar.

Inductive computation : Test for 2-colorability

1) Not all series-parallel graphs are 2-colorable (see K_3)
2) G, H 2-colorable does not imply that G//H is 2-colorable (because $K_3=P_3//e$).

3) One can check 2-colorability with 2 auxiliary properties :

Same(G) = G is 2-colorable with sources of the same color,
Diff(G) = G is 2-colorable with sources of different colors

and by using rules :

Diff(e) = True ; Same(e) = False

Same(G//H) $\Leftrightarrow$ Same(G) $\wedge$ Same(H)

Diff(G//H) $\Leftrightarrow$ Diff(G) $\wedge$ Diff(H)

Same(G•H) $\Leftrightarrow$ (Same(G) $\wedge$ Same (H)) $\vee$ (Diff(G) $\wedge$ Diff(H))

Diff(G•H) $\Leftrightarrow$ (Same(G) $\wedge$ Diff(H)) $\vee$ (Diff(G) $\wedge$ Same(H))

We can compute for every SP-term t, by induction on the structure of t the pair of Boolean values (Same(Val(t)) , Diff(Val(t))). We get the answer for G = Val(t) (the graph that is the *value* of t) regarding 2-colorability.

Important facts :

1) The existence of properties forming an inductive set (w.r.t. operations of F) is equivalent to recognizability in the considered F-algebra.

2) The simultaneous computation of m inductive properties can be implemented by a "tree" automaton with 2^m states working on terms t. This computation takes time $O(|t|)$.

3) An inductive set of properties can be constructed (at least theoretically) from every monadic-second order formula.

4) This result extends to the computation of values (integers) defined by monadic-second order formulas.

Definition : A set L of words, of trees, of graphs or relational structures is Monadic Second-Order (MS) definable iff

$$L = \{ S \ / \ S \models \varphi \} \text{ for an MS formula } \varphi$$

Theorem : (1) A language (set of words or *finite terms*) is recognizable $\Leftrightarrow$ it is MS definable

(2) A set of finite graphs is VR-recognizable $\Leftarrow$ it is MS definable

(3) A set of finite graphs is HR-recognizable $\Leftarrow$ it is MS_2 definable

Proofs:

(1) Doner, Thatcher, Wright, see W. Thomas, Handbook formal languages, vol.3.

(2) (3) There are two possible proofs . I sketch the informative one.

Basic facts for (2) :

Let F consist of $\oplus$ and unary quantifier-free definable operations f.

For every MS formula φ of quantifier-height k, we have

(a) for every f , one can construct a formula $f^{\#}(\varphi)$ such that :

$$f(S) \models \varphi \Leftrightarrow S \models f^{\#}(\varphi)$$

(b) (Hanf, Fefermann and Vaught, Shelah) one can construct formulas $\psi_1,\ldots,\psi_n,\theta_1,\ldots,\theta_n$

such that :

$$S \oplus T \models \varphi \Leftrightarrow \text{for some } i,\ S \models \psi_i \wedge T \models \theta_i$$

where $f^{\#}(\varphi)$, $\psi_1,\ldots,\psi_n$, $\theta_1,\ldots,\theta_n$ have quantifier-height $\leq$ k.

(c) Up to equivalence, there are finitely many formulas of quantifier-height $\leq$ k forming a set Φ_k.

One builds an automaton with states the subsets of Φ_k : the MS-theories of quantifier-height $\leq$ k of the graphs defined by the subterms of the term to be processed.

Feferman-Vaught for $\oplus$:

$$S \oplus T \models \varphi \iff \bigvee_i S \models \psi_i \wedge T \models \theta_i$$

with $qh(\psi_i), qh(\theta_i) \leq qh(\varphi)$

$$Th_h(S \oplus T) = \Phi_{\oplus, h}(Th_h(S), Th_h(T))$$

We need to handle formulas with free variables

$$(S \oplus T, \bar{A}) \models \varphi \iff \bigvee_i (S, \bar{A} \upharpoonright S) \models \psi_i \wedge (T, \bar{A} \upharpoonright T) \models \theta_i$$

$$sat(S \oplus T, \varphi, \bar{X}) = \biguplus_i sat(S, \psi_i, \bar{X}) \boxtimes sat(T, \theta_i, \bar{X})$$

where $\mathcal{B} \boxtimes \mathcal{C} = \{ (B_1 \cup C_1, \ldots, B_n \cup C_n) \ / \ \bar{B} \in \mathcal{B}, \bar{C} \in \mathcal{C} \}$

MS logic with only set variables :

$X \subseteq Y$, $X = \emptyset$, $Sgl(X)$, $Card_{p,q}(X)$

$$R(X, Y, Z) \iff \exists x \in X, y \in Y, z \in Z \,.\, R(x, y, z)$$

$$\exists x \,.\, \varphi(x, \ldots) \iff \exists X \,.\, Sgl(X) \wedge \tilde{\varphi}(X, \ldots)$$

Proof by induction on φ :

There exists a decomposition

$$[(\psi_1, \theta_1), \ldots, (\psi_m, \theta_m)]_\varphi \quad qh(\psi_i), qh(\theta_i) \leq qh(\varphi)$$

s.t.

$$sat(S \oplus T, \varphi) = \bigcup_i sat(S, \psi_i) \boxtimes sat(T, \theta_i)$$

Cases :

$X \subseteq Y$: $[(X \subseteq Y, X \subseteq Y)]$

$Sgl(X)$: $[(Sgl(X), X = \emptyset), (X = \emptyset, Sgl(X))]$

$R(X, Y)$: $[(R(X,Y), True), (True, R(X,Y))]$

$\varphi \vee \varphi'$: $[\ldots\ldots]_\varphi \cdot [\ldots\ldots]_{\varphi'}$

$\neg \varphi$: We use a lemma:

$$\bigwedge_i \neg\alpha_i \vee \neg\beta_i \equiv \bigvee_I \left(\bigwedge_{i \in I} \neg\alpha_i \wedge \bigwedge_{i \notin I} \neg\beta_i \right)$$

gives $\wedge$

$(S \oplus T, \bar{A}) \models \exists X. \varphi(X)$

$\iff (S \oplus T, \bar{A}B) \models \varphi$ for some B

$\iff \bigvee_i (S, \bar{A}\restriction_S B\cap D_S) \models \psi_i \wedge (T, \bar{A}\restriction_T B\cap D_T) \models \theta_i$

$\iff \bigvee_i (S, \bar{A}\restriction S) \models \exists X. \psi_i(X) \wedge (T, \bar{A}\restriction T) \models \exists X. \theta_i(X)$

Hence we get $[\cdots\cdots]_{\exists X. \varphi(X)}$ from $[\cdots]_\varphi$ by adding $\exists X.$ to formulas.

The condition on quantifier-height is satisfied

For $\forall X. \varphi(X)$ we use $\neg \exists X. \neg \varphi(X)$.

<u>Exercises</u> : • $Card_{p,q}(X)$

• Replace $\bigcup_i$ by $\biguplus_i$

8.

Graph Rewriting

1) graph rewriting rules : local graph transformations

2) grammars based on graph rewriting rules
   - either context-free : there are 2 types :
     -- hyperedge replacement rewriting rules, fairly manageable, equivalent to HR-equation systems
     -- vertex replacement, very complicated, equivalent to VR-equation systems
   - or more general : hard to obtain general results

3) possible applications :
   - graph properties invariant under application of local graph transformations,
   - verification of programs with complex data structures and pointers.

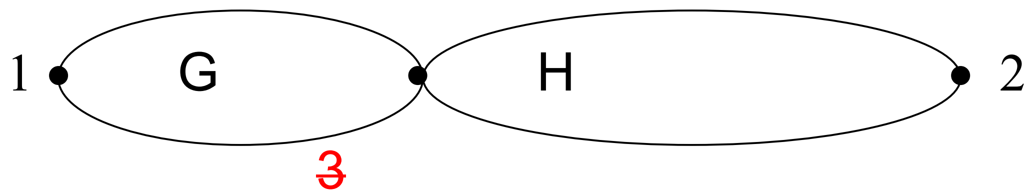
Example : *Series-parallel graphs*

defined as the set of graphs with sources 1 and 2,

generated from e $= 1 \longrightarrow 2$ and the operations // (parallel-composition) and

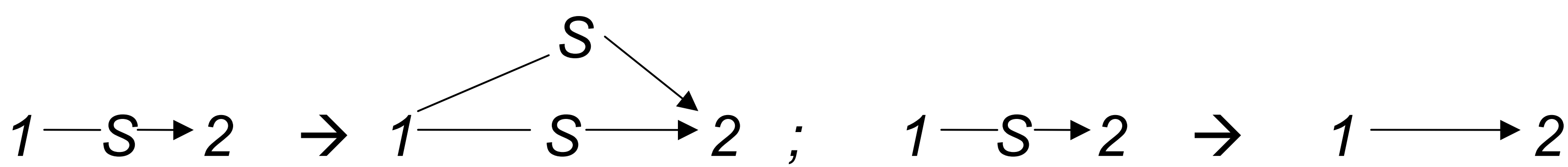
series-composition defined from other operations by :

$$G \bullet H = \mathit{Forget}_3(\mathit{Ren}_{2,3}(G) \,//\, \mathit{Ren}_{1,3}(H))$$



Equation : $S = S//S \cup S \bullet S \cup e$

Rewriting rules $1 \text{—} S \rightarrow 2 \;\Rightarrow\; 1 \text{—} S \rightarrow \bullet \text{—} S \rightarrow 2$



Local graph transformations :

Rule : L → R , L and R are graphs with same source names.

Application of the rule :

G → H if G = $Forget_{All}$(K//L) and H = $Forget_{All}$(K//R)

for some graph K with same source names as L,R.

Intuition : in G, a subgraph isomorphic to L is replaced by R ; the "gluing vertices" are sources.

Questions to ask :

For a property P :

1) Is it true that if G satisfies P, and L→R is applicable, the resulting graph satisfies P?

   If P is first-order expressible or monadic second-order expressible, then this is decidable on graphs of tree-width $\leq$ k, for each k.

2) Is it true that all graphs derivable from G by a given finite set of rules satisfy P ?

   One must give more hypotheses.

Monadic second-order transductions

STR(Σ): the set of finite Σ-relational structures (or finite directed ranked Σ-hypergraphs).

MS transductions are multivalued mappings : τ : STR(Σ) $\rightarrow$ STR(Γ)

$$S \longmapsto T = \tau(S)$$

where T is :

a) defined by MS formulas

b) inside the structure: $S \oplus S \oplus \ldots \oplus S$

(fixed number of disjoint "marked" copies of S)

c) in terms of "parameters", subsets $X_1, \ldots, X_p$ of the domain of S.

Proposition : The composition of two MS transductions is an MS transduction.

Remark : For each tuple of parameters $X_1, \ldots, X_p$ satisfying an MS property, T is uniquely defined. τ is multivalued by the different choices of parameters.

Examples : $(G,\{x\}) \longmapsto$ the connected component containing x.

$(G,X,Y) \longmapsto$ the minor of G resulting from contraction of edges in X and deletion of edges and vertices in Y.

Example of an MS transduction (without parameters) : The *square* mapping δ on words: $u \mapsto uu$

For u = aac, we have S $\quad \bullet \to \bullet \to \bullet$

$$\begin{array}{ccc} \bullet \to & \bullet \to & \bullet \\ a & a & c \end{array}$$

$S \oplus S$

$$\begin{array}{ccc} \bullet \to & \bullet \to & \bullet \\ a & a & c \\ p_1 & p_1 & p_1 \end{array} \qquad \begin{array}{ccc} \bullet \to & \bullet \to & \bullet \\ a & a & c \\ p_2 & p_2 & p_2 \end{array}$$

δ(S)

$$\begin{array}{cccccc} \bullet \to & \bullet \to & \bullet \to & \bullet \to & \bullet \to & \bullet \\ a & a & c & a & a & c \end{array}$$

In δ(S) we redefine Suc (i.e., $\to$) as follows :

Suc(x,y) : $\Leftrightarrow$ p_1 (x) & p_1 (y) & Suc(x,y) v p_2 (x) & p_2 (y) & Suc(x,y)
v p_1 (x) & p_2 (y) & "x has no successor" & "y has no predecessor"

We also remove the "marker" predicates p_1, p_2.

The fundamental property of MS transductions :

$$\begin{array}{ccc} S & \longmapsto & \tau(S) \\ \tau^{\#}(\psi) & \longleftarrow\!\shortmid & \psi \end{array}$$

Every MS formula ψ has an effectively computable *backwards translation* $\tau^{\#}(\psi)$, an MS formula, such that :

$$S \models \tau^{\#}(\psi) \quad \text{iff} \quad \tau(S) \models \psi$$

The verification of ψ in the object structure $\tau(S)$ reduces to the verification of $\tau^{\#}(\psi)$ in the given structure S.

Intuition : S contain all necessary information to describe $\tau(S)$; the MS properties of $\tau(S)$ are expressible by MS formulas in S

Consequence : If $L \subseteq STR(\Sigma)$ has a decidable MS satisfiability problem, so has its image under an MS transduction.

Other results

1) A set of graphs is VR -equational iff it is the image of (all) binary trees under an MS transduction. VR-equational sets are stable under MS-transductions.

A set of graphs has bounded clique-width iff it is the image of a set of binary trees under an MS transduction.

2) A set of graphs is HR-equational iff it is the image of (all) binary trees under an MS_2 transduction.

HR-equational sets are stable under MS_2-transductions.

A set of graphs has bounded tree-width iff it is the image of a set of binary trees under an MS_2 transduction.

Relationships between algebraic and logical notions

Algebraic notions	Algebraic characterizations	Logical characterizations	Closure properties
EQ			union, $\cap$ Rec
	equation systems	MS-trans(Trees)	homo
	Val(REC(Terms))		MS-trans
REC			Boolean opns
	congruences	MS-def $\subset$ REC	homo^{-1}
			MS-trans^{-1}

Signatures for graphs and hypergraphs :

HR : graphs and hypergraphs with “sources”
VR : graphs with vertex labels (“ports”)
VR^+ : VR with quantifier-free operations (ex. edge complement)

Links between MS logic and combinatorics:

Seese's Theorem and Conjecture

Theorem (Seese 1991): If a set of graphs has a decidable MS_2 satisfiability problem, it has bounded tree-width.

Conjecture (Seese 1991): If a set of graphs has a decidable MS satisfiability problem, it is the image of a set of trees under an MS transduction, equivalently, has bounded clique-width.

Theorem (B.C., S. Oum 2004): If a set of graphs has a decidable C_2MS satisfiability problem, it has bounded clique-width.

MS = (Basic) MS logic without edge quantifications, MS_2 = MS logic with edge quantifications
C_2MS = MS logic with even cardinality set predicates. A set *C* has a decidable *L* satisfiability problem if one can decide for every formula in *L* whether it is satisfied by some graph in *C*

Proof of Seese's Theorem :

A) If a set of graphs C has unbounded tree-width, the set of its minors includes all k x k-grids (Robertson, Seymour)

B) If a set of graphs contains all kxk-grids, its MS_2 satisfiability problem is undecidable

C) If C has decidable MS_2 satisfiability problem, so has Minors(C), because $C \longrightarrow$ Minors(C) is an MS_2 transduction.

Hence, if C has unbounded tree-width and a decidable MS_2 satisfiability problem, we have a contradiction for the decidability of the MS_2 satisfiability problem of Minors(C).

Proof of Courcelle-Oum's Theorem :

D) Equivalence between the cases of all (directed and undirected) graphs and bipartite undirected graphs.

A') If a set of bipartite graphs C has unbounded clique-width, the set of its vertex-minors contains all "S_k" graphs

C') If C has decidable C_2MS satisfiability problem, so has Vertex-Minors(C), because $C \longrightarrow$ Vertex-Minors(C) is a C_2MS transduction.

E) An MS transduction transforms S_k into the kxk-grid.

Hence A' + B + C' + E gives the result for bipartite undirected graphs. Result with D.

Definitions and facts

Local complementation of G at vertex v

$G * v$ = G with edge complementation of $G[n_G(v)]$,

the subgraph induced by the neighbours of v

Local equivalence ($\approx_{loc}$) = transitive closure of local complementation

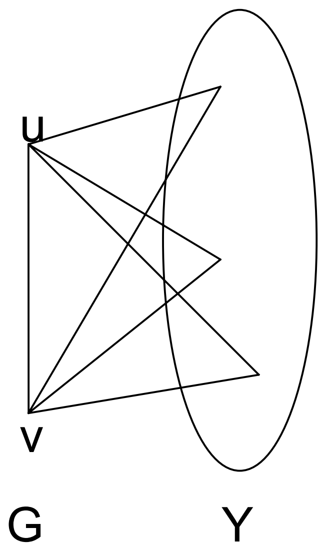
(at all vertices)

Vertex-minor relation :

$H \leq_{VM} G \;:\Leftrightarrow\; H$ is an induced subgraph of some $G' \approx_{loc} G$.

Proposition (Courcelle and Oum 2007) : The mapping that associates with G its locally equivalent graphs is a C_2MS transduction.

Why is the even cardinality set predicate necessary ?



Consider $G * X$ for $X \subseteq Y$:

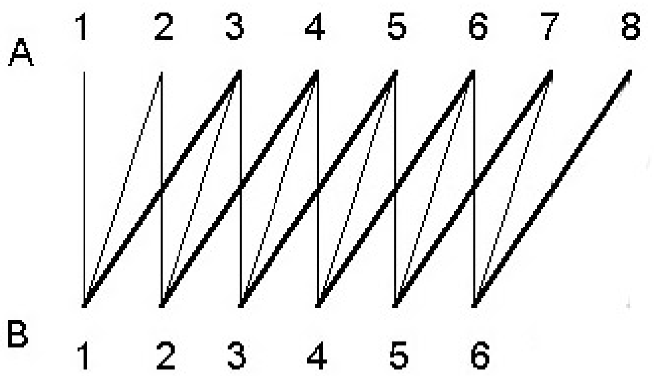
u is linked to v in $G * X$

$\Leftrightarrow$ Card(X) is even

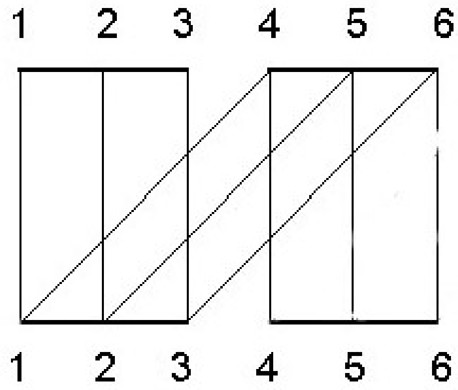
Definition of S_k : bipartite : $A = \{1,\ldots,(k+1)(k-1)\}$, $B = \{1,\ldots,k(k-1)\}$ for $j \in A$, $i \in B$:

$edg(i,j) \Leftrightarrow i \leq j \leq i+k-1$

From S_k to $Grid_{kxk}$ by an MS transduction



S_3



(folded) $Grid_{3x4}$

1) One can define the orderings of A and B :

$$x, y \text{ are consecutive} \Leftrightarrow Card(n_G(x)\,\Delta\, n_G(y)) = 2$$

2) One can identify the edges from $i \in B$ to $i \in A$, and

from $i \in B$ to $i+k-1 \in A$ (thick edges on the left drawing)

3) One can create edges (e.g. from $1 \in A$ to $2 \in A$, from $2 \in A$ to $3 \in A$ etc…and similarly for B, and from $1 \in B$ to $4 \in A$, etc…) and delete others (from $4 \in B$ to $6 \in A$ etc…), and vertices like 7,8 in A, to get a grid containing $Grid_{kxk}$

Corollary : If a set of directed acyclic graphs having Hamiltonian directed paths has a decidable MS satisfiability problem, then :

it has bounded clique-width,

it is the image of a set of trees under an MS transduction.

Proof : Since on these graphs a linear order is MS definable, MS and C_2MS are equivalent.

The previously known techniques for similar results (in particular for line graphs or interval graphs, B.C. 2004) do not work in this case.