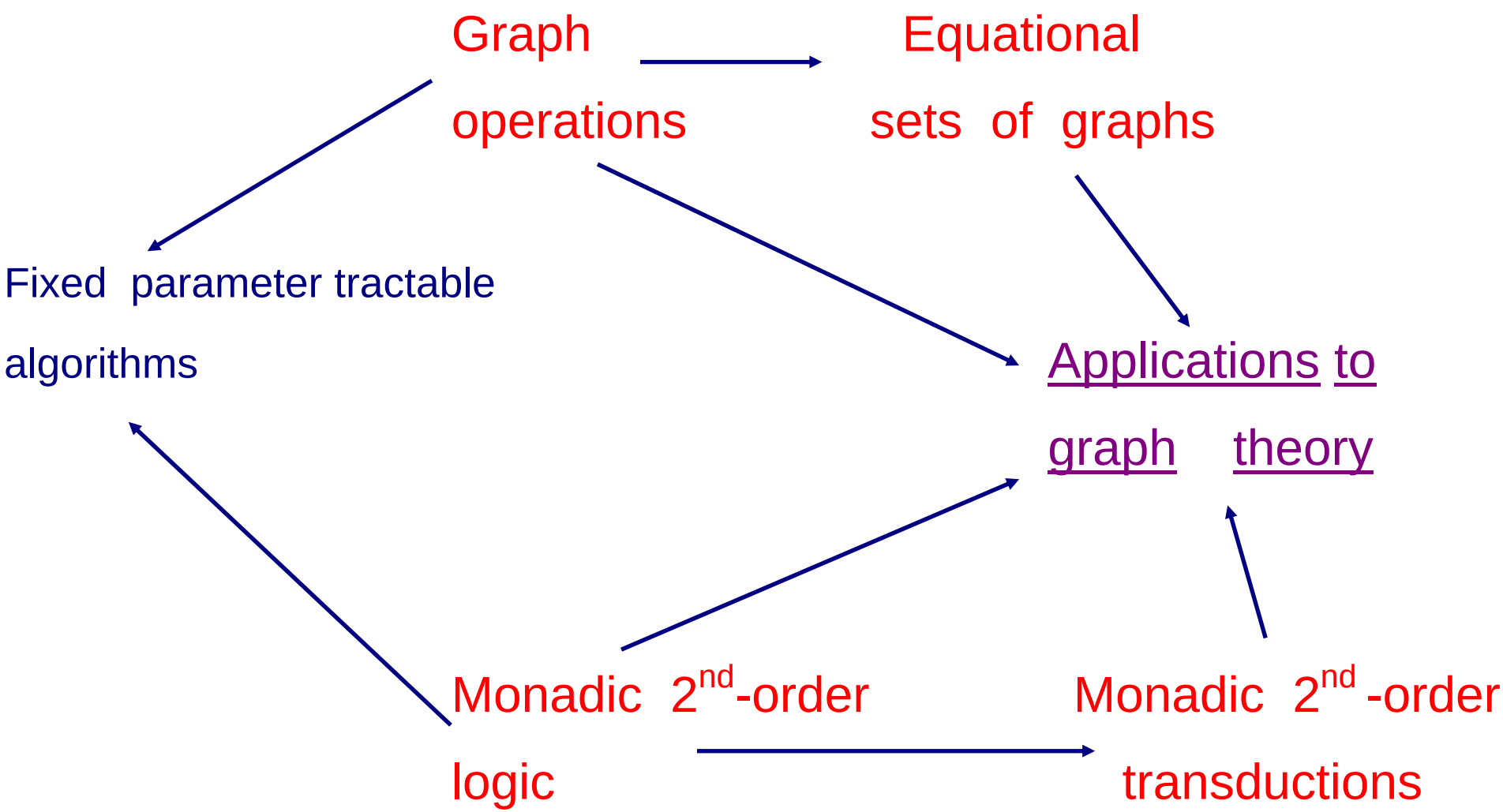
Logical Descriptions of Graph Hierarchies and of many other things

Bruno Courcelle

Université Bordeaux 1, LaBRI, & Institut Universitaire de France

Reference : Graph structure and monadic second-order logic,
Book to be published by Cambridge University Press, readable on :
http://www.labri.fr/perso/courcell/ActSci.html

A chart of main notions



Logic = Monadic Second-Order Logic (MS)

→ Formal expression of graph properties

→ And of graph transformations (MS transductions)

1) Descriptions and algorithmic constructions of sets of minimal excluded minors and minimal induced subgraphs
2) Logical characterizations of the graph hierarchies based on tree-width or clique-width (or rank-width)

   → short proofs of tree-width or clique-width (un)boundedness
3) A linear hierarchy of graph classes based on MS transductions

Monadic Second-Order Logic (quick review)

= First-order logic on power-set structures

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

MS (expressible) properties : transitive closure, properties of paths, connectivity, planarity (via Kuratowski, uses connectivity), *k*-colorability.

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

3-colorability :

$$\exists X,Y \,(\text{"}X,Y \text{ are disjoint"} \wedge \forall u,v \{ edg(u,v) \Rightarrow [(u \in X \Rightarrow v \notin X) \wedge (u \in Y \Rightarrow v \notin Y) \wedge (u \in V-(X \cup Y) \Rightarrow v \notin V-(X \cup Y)]\})$$

Non connectivity :

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

Transitive and reflexive closure : TC(R ; x, y) :

$\forall$ X { "X is R-closed" $\wedge$ x $\in$ X $\Rightarrow$ y $\in$ X }

where "X is R-closed" is defined by : $\forall u,v \, (u \in X \, \wedge \, R(u,v) \Rightarrow v \in X)$

R can be defined by a formula φ_R as in :

$\forall x,y \, (x \in Y \, \wedge \, y \in Y \Rightarrow$ TC("$u \in Y \, \wedge v \in Y \wedge edg(u,v)$" ; x, y)

expressing that G[Y] is connected (Y is free in φ_R).

Application : G contains (fixed) H as a minor where $V_H = \{1,\ldots,k\}$:

there exist pairwise disjoint vertex sets $X_1,\ldots, X_k$
in G such that each $G[X_i]$ is connected and, whenever
if i -- j in H, there is an edge between X_i and X_j in G.

Consequence : planarity is MS-expressible (no minor K_5 or $K_{3,3}$).

Edge set quantifications *increase* the expressive power

Incidence graph of G undirected, **Inc**(G) = ($V_G \cup E_G$, $inc_G(.,.)$)

$inc_G(v,e) \Leftrightarrow$ v is a vertex of edge e.

Monadic second-order formulas written with inc can use quantifications on sets of edges : they define MS_2 –expressible graph properties.

The existence of a *perfect matching* or of a *Hamiltonian circuit* or of a spanning tree of degree ≤ 3 is MS_2 -expressible but not MS-expressible.

Definition : A set graphs L is MS_1-definable if $L = \{ G \text{ finite } / \quad G \models \varphi \}$.

It is MS_2 –definable if $L = \{ G \text{ finite } / \quad \mathbf{Inc}(G) \models \varphi \}$.

(for a fixed MS *sentence* (formula without free variables) φ).

Two descriptions of graph properties

MS_1 : MS sentences interpreted on structure $(V_G , edg_G (.,.))$ (allows only vertex set quantifications)

FPT verification parameter clique-width

MS_2 : MS sentences interpreted on incidence graphs : $(V_G \cup E_G, inc_G(.,.))$ (allows vertex and edge set quantifications)

FPT verification parameter tree-width

MS logic is interesting with conditions like bounded tree-width or bounded clique-width (3-colorability is MS but NP-complete).

Algebraic view of graph decompositions

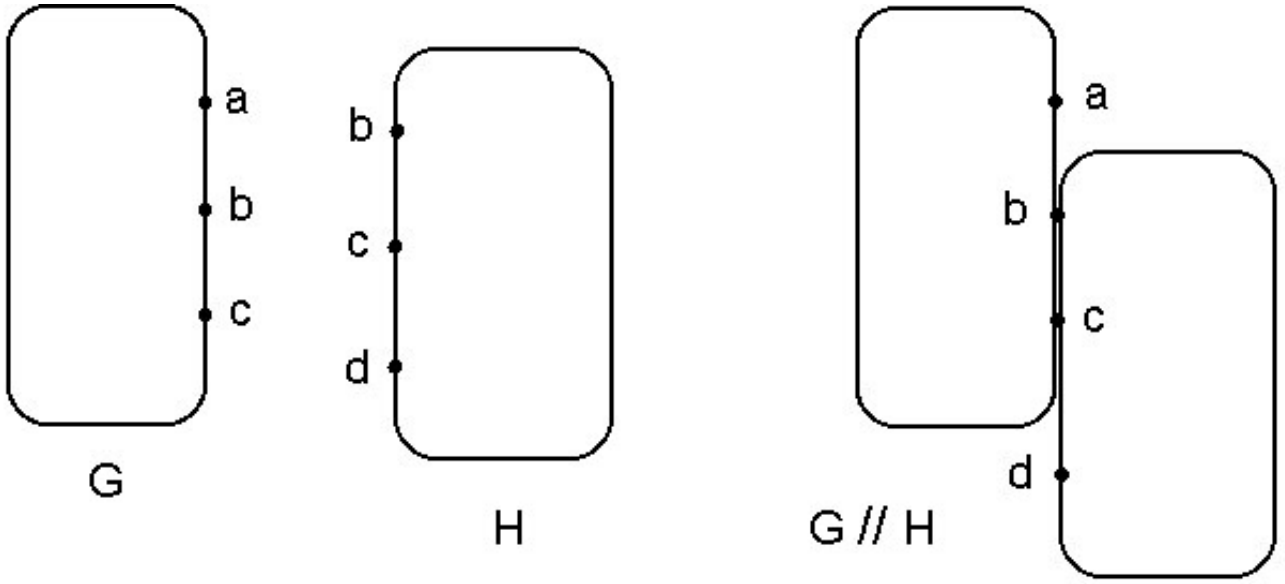
Graph operations characterizing tree-width

Graphs have distinguished vertices called *sources* (or terminals or boundary vertices) pointed to by labels from a finite set : {*a, b, c, ..., h*}.

Binary operation(s) : *Parallel composition*

G // H is the disjoint union of G and H ; 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 :

$Ren_{a \leftrightarrow b}(G)$ exchanges source labels *a* and *b*

(replaces *a* by *b* if *b* is not the label of a source)

Constant symbols denote *basic graphs* : the connected graphs with at most one edge.

An *algebra* of graphs the "tree-width" algebra

Proposition : A graph has tree-width $\leq$ k $\Leftrightarrow$ it is defined by a term that uses $\leq$ k+1 source labels.

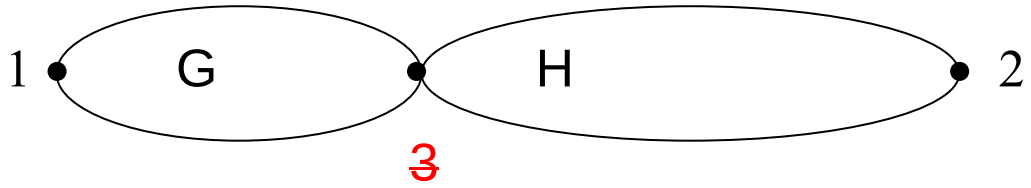
Example : Directed series-parallel graphs

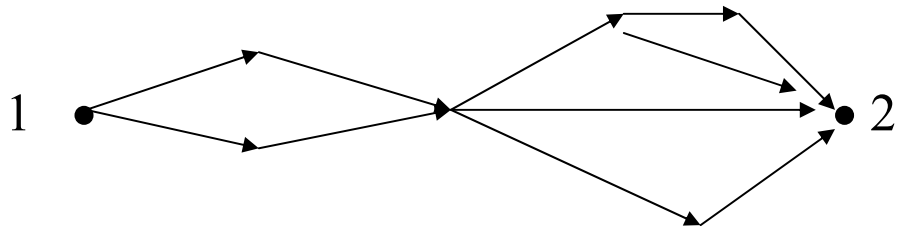
They are generated by the constant e = 1 ⟶ 2 ,

// (parallel-composition) and series-composition defined from other operations by :

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

Example :





The defining equation (S is the set of series-parall graphs) :

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

Graph operations defining clique-width

Graphs are simple, directed or not.

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

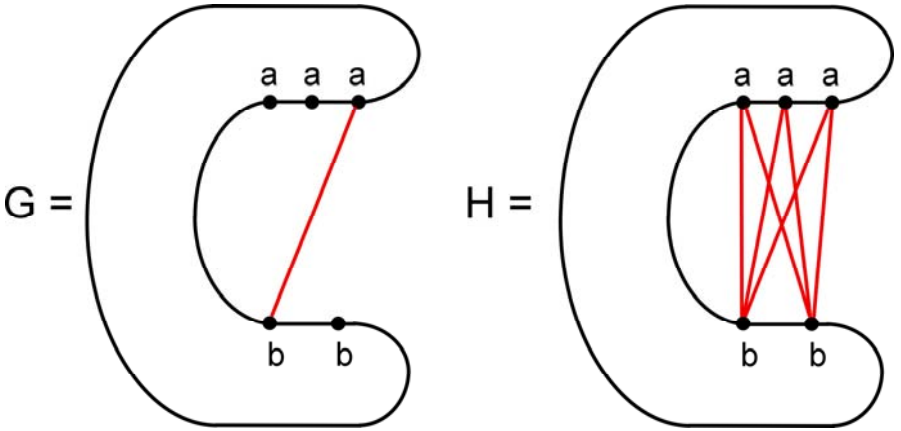
a label *p* may label several vertices, called the *p-ports*.

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

Unary operations: Edge addition denoted by $Add\text{-}edg_{a,b}$

$Add\text{-}edg_{a,b}$(G) is G augmented with directed or undirected edges from every *a*-port to every *b*-port.

The number of added edges depends on the argument graph.



H = $Add\text{-}edg_{a,b}$(G) ; only 5 new edges added

Vertex relabellings :

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

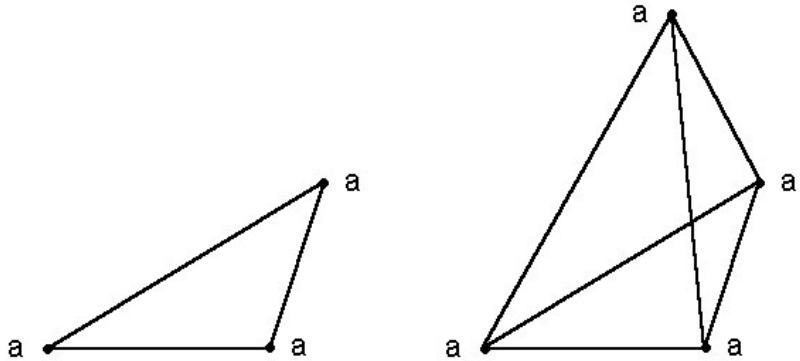
Basic graphs are those with a single vertex.

Another algebra of graphs : the "clique-width" algebra.

Definition : A graph has clique-width $\leq$ k $\Leftrightarrow$ it is defined by a term that uses $\leq$ k labels.

Example : Cliques have clique-width 2.

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



The problem of checking if G has clique-width $\leq$ k is NP-complete (Fellows *et al.*) (input is (G,k))

The equivalent notion of *rank-width* has good combinatorial and algorithmic properties; it has also an algebraic characterization with more complicated operations (compositions of clique-width operations).

Defined first for undirected graphs (Oum and Seymour), but extended to directed ones (Kanté).

From both algebras, we get :

1) linear notations for finite graphs,

2) finite descriptions of (certain) infinite sets of finite graphs and compact descriptions of (certain) finite sets of finite graphs, by means of Equation Systems (defining the equational sets of the corresponding algebras).

Examples of equational sets of graphs

In the "tree-width" algebra :

Series-parallel : $S = S // S \cup S \bullet S \cup e$

Biconnected outerplanar : (u = undirected edge)

$B = fg_1(fg_2(u // Q))$, $Q = u // Q \cup Q \bullet Q \cup u$

In the "clique-width" algebra :

Cographs : $C = C \oplus C \cup C \otimes C \cup \mathbf{1}$

$G \otimes H$ = *Relab*$_{2 \to 1}$(*Add-edg*$_{1,2}$(G $\oplus$ *Relab*$_{1 \to 2}$ (H))) (complete join)

Threshold graphs : $T = T \oplus \mathbf{1} \cup T \otimes \mathbf{1} \cup \mathbf{1}$

Equational sets in algebras in general

We consider systems of equations where S, T define sets of graphs or …

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

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

and :

f	is a binary operation,
g, k, m	are unary operations,
a, b	denote basic objects (e.g. graphs up to isomorphism).

An *equational set* is a component of the least solution of such an equation system.

This is well-defined in any algebra : Least Fixed Point Theorem.

For graphs, some facts do not hold as we could wish:

1) The set of all (finite) graphs is not equational (both algebras).

2) Neither are the sets of planar graphs and of square grids.

3) Parsing is sometimes NP-complete

(checking clique-width, cyclic band-width at most 2)

Theorem : (1) For each k, the set TWD(≤ k) of graphs of tree-width ≤ k is equational in the "tree-width"-algebra, and every "tree-width"-equational set has bounded ("boundable") tree-width.

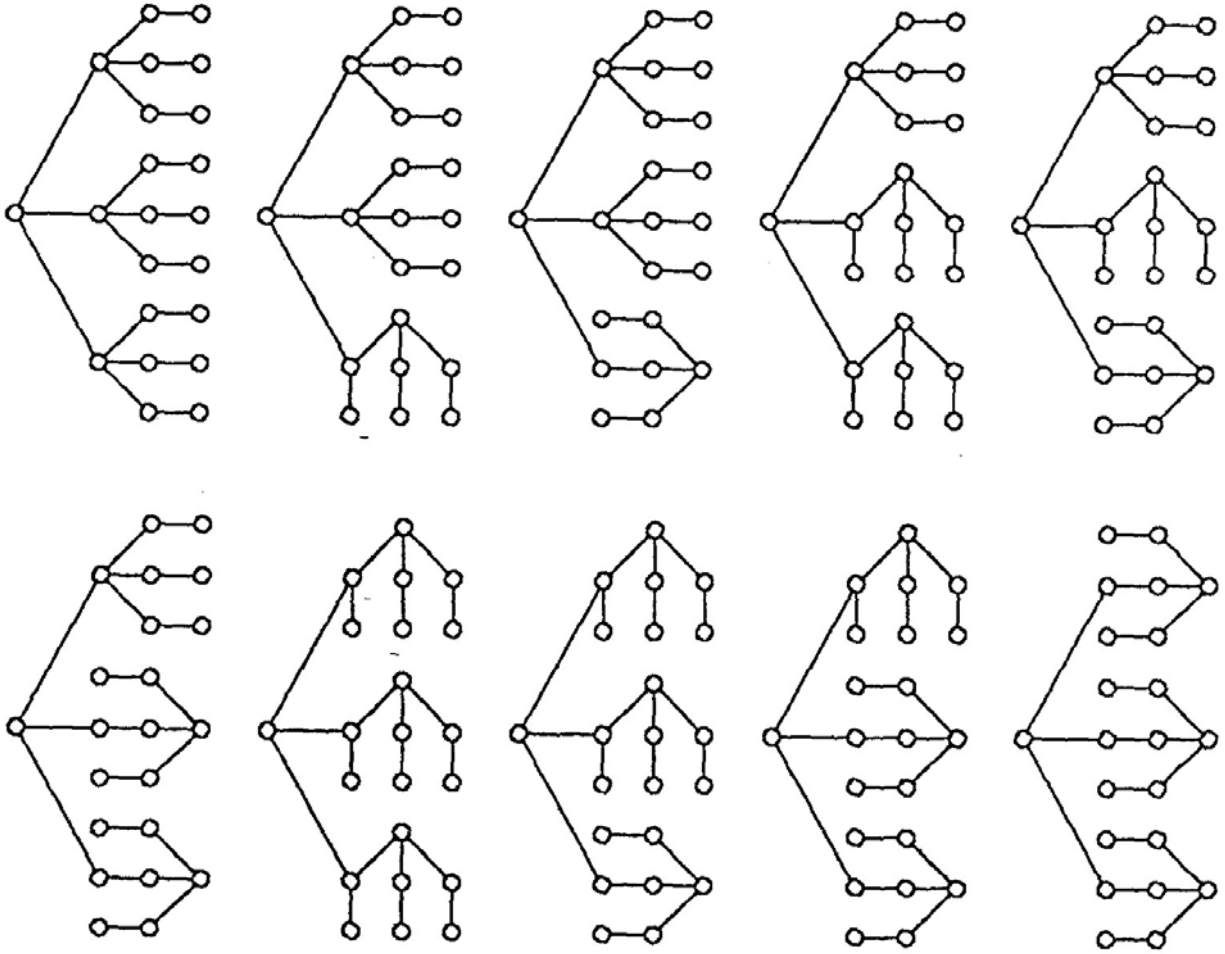
(2) Analogous facts for clique-width.

Filtering Theorems : (1) If L is "tree-width"-equational, and K is MS_2-definable, then $L \cap K$ is (effectively) "tree-width"-equational.

(2) If L is "clique-width"-equational, and K is MS_1-definable, then $L \cap K$ is (effectively) "clique-width"-equational.

Examples of compact descriptions of finite sets

Set T_2



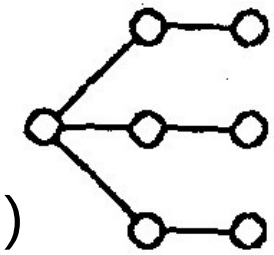
What do they come from ?

T_2 = the *trees* that are the minimal excluded minors for the class of graphs of path-width ≤ 2.

T_k is the corresponding set for path-width $\leq k$ where (Kajitani et al.) :

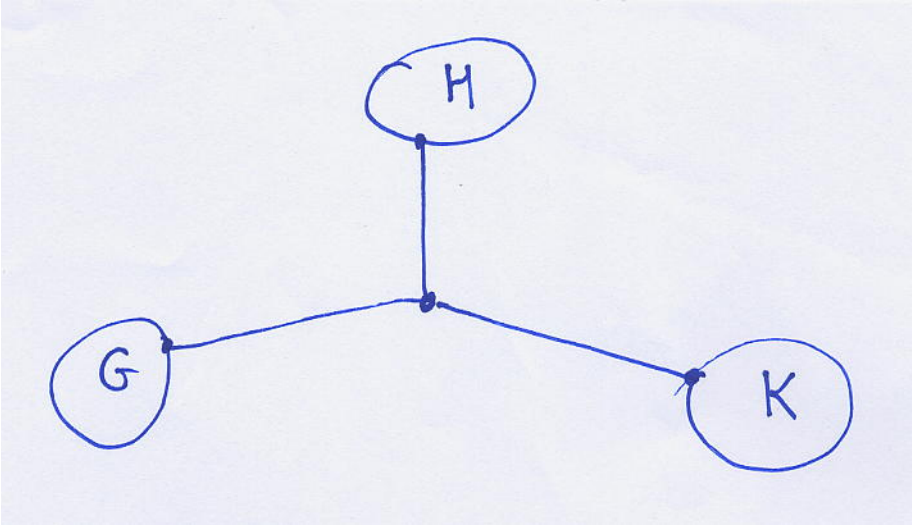
T_1 consists of

$T_{k+1} = S(T_{k+1}, T_{k+1}, T_{k+1})$



S(A,B,C) = set of star-compositions :

for all $G \in A$, $H \in B$, $K \in C$.



Each set T_k has more than $(k!)^2$ graphs, all with $(5/2).(3^k-1)$ vertices,

but has an equation system over the "tree-width"-algebra of size O(k)

The obstruction set of each minor-closed class is finite (Graph Minor Theorem) but in many (most ?) cases, very large and difficult to compute.

Graphs on the torus : *thousands* of graphs in the obstruction.

They are not random sets.

A list of 10 000 graphs produced by a computer is of little use.

Grammars should be able to enlighten the regularities.

Logical and equational description of obstructions

1) Let $\mathcal{C}$ be minor-closed and characterized by an MS_2 sentence φ (not constructed from the obstructions), then the obstruction set $\Omega(\mathcal{C})$ is characterized by the MS_2 sentence ψ saying that :

   $G \models \neg\varphi$ and for every vertex u, $G - u \models \varphi$ and

   for every edge e , $G - e \models \varphi$ and

   for every edge e , $G / e \models \varphi$ (G / e = contraction of e).

2) Although we know that ψ characterizes finitely many graphs, no algorithm can list them just from the input ψ.

3) For each k, one can construct (using ψ) the finite set $\Omega(\mathcal{C}) \cap TWD(\leq k)$.

4) From an upper bound to the tree-width of $\Omega(\mathcal{C})$, we can compute this set

5) At 3) one can construct an equation system for $\Omega(C) \cap$ TWD($\leq$ k).

But we have no guarantee it will be readable.

Applications (remain "theoretical" because computations are intractable) :

(AGK = Adler, Grohe, Kreutzer 2008)

For each k, n :

Graphs of path-width $\leq$ k	(Kabanets 1997, AGK)
Graphs of tree-width $\leq$ k	(AGK)
Graphs of tree-depth $\leq$ k	(below)
Graphs of n-depth tree-width $\leq$ k	(below)
Graphs embeddable on a surface	(AGK)

Apex graphs over a minor-closed class with known obstruction set (AGK)

Union of 2 minor-closed classes with known obstruction sets. (AGK)

Induced subgraph obstructions

Hereditary classes C (closed under induced subgraphs) may have infinite induced subgraph obstruction sets $\Sigma(C)$ showing some "regularities"

Examples	*bounds on the corresponding sets* Σ	
Chordal graphs	tree-width ≤ 2	"twd"-equational
Perfect graphs	clique-width ≤ 4	"cwd"-equational
Interval graphs	tree-width ≤ 3	"twd"-equational
Comparability graphs	clique-width $\leq ? \leq 10$	"cwd"-equational

→ Equation systems are able to capture their regularities.

Theorem : Let $\mathcal{C}$ be a hereditary class of graphs.

1) If $\mathcal{C}$ is MS_1-definable, then $\Sigma(\mathcal{C}) \cap CWD(\leq k)$ is "cwd"-equational.

2) If $\mathcal{C}$ is MS_2-definable, then $\Sigma(\mathcal{C}) \cap TWD(\leq k)$ is "twd"-equational.

As for minor-closure, from an MS_1 or MS_2 sentence that characterizes $\mathcal{C}$, one can build an MS_1 or MS_2 sentence that characterizes $\Sigma(\mathcal{C})$.

One uses then the Filtering Theorems : equational sets filtered by MS properties.

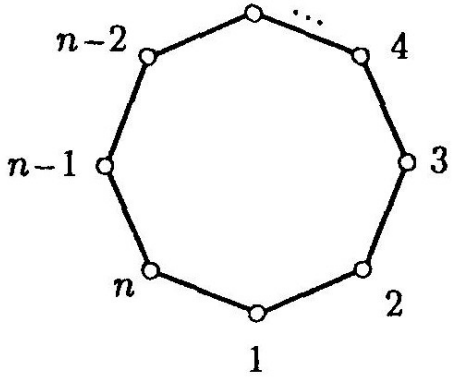
Application to interval graphs (the intersection graph of a set of intervals of integers)

B. & L. characterize them as “chordal with no asteroidal triple”:

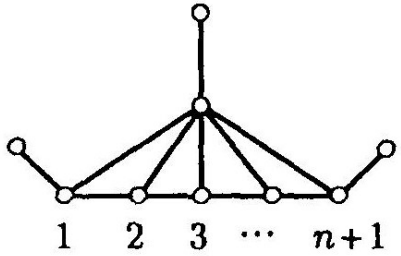
this is MS_1 expressible.

From this characterization, the obstructions have tree-width ≤ 5

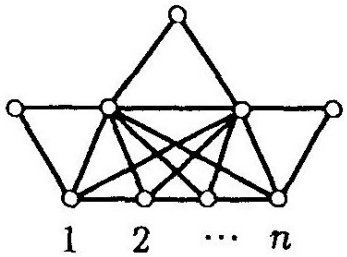
In principle, one could construct a “twd” - equation system defining these graphs, known from B.&L. and of tree- width ≤ 3



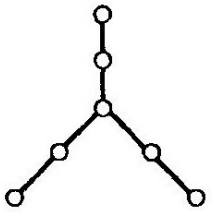
$G_n; n \geq 4$



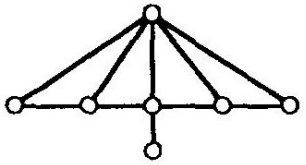
$K_n; n \geq 1$



$L_n; n \geq 1$



B_1



B_2

Other applications

An algorithm can construct the *finite* set $\Sigma(C)$ if C is a hereditary and MS_1 - definable class of cographs.

Finite because cographs are well-quasi-ordered for induced subgraph inclusion (Damaschke, 1990)

Cographs have clique-width 2, hence $\Sigma(C)$ has cwd ≤ 3.

Examples : 1) Threshold graphs ($\Sigma = \{ P_4, C_4, K_2 \oplus K_2 \}$)

2) Cographs with "modular decomposition tree" of height $\leq$ k.

3) "Semi-threshold" : $T = T \oplus T \cup T \otimes \mathbf{1} \cup \mathbf{1}$

Open problems :

(1) Find in such a way the set Σ(*Comparability Graphs*), an infinite set identified by Gallai that is "cwd"-equational.

(2) Treat related classes of partially ordered sets.

(3) Design systematic methods to construct "small" "twd"- or "cwd"-equation systems, (or equation systems of other types) to represent finite and infinite obstruction sets.

Tools : Monadic second-order logic + algebraic notions (equational and recognizable sets) + graph theoretic arguments.

Monadic second-order transductions

Transformations of graphs (more generally of relational structures) specified by MS_1 (or MS_2) formulas.

There are 2 representations for an input graph and 2 for the output, hence 4 types of graph transductions, denoted by :

$MS_{1,1}$ (or MS to simplify), $MS_{1,2,}$ $MS_{2,1}$ and $MS_{2,2}$

$MS_{i,o}$ means i = type of representation of input, o = type of repres. of output.

I will mainly compare MS-transductions, for graphs G handled as (V_G, edg_G) and $MS_{2,2}$-transductions, for graphs G represented by their incidence graphs = $(V_G \cup E_G , inc_G)$

Main Results (to be made more precise) :

(1) MS-transductions preserve bounded clique-width and "clique-width"-equational sets

(2) $MS_{2,2}$-transductions preserve bounded tree-width and "tree-width"-equational sets

Meaning : Robustness of the graph hierarchies based on clique-width and tree-width.

The word "transduction" comes from Formal Language Theory ; My aim is to extend FLT to graphs and other combinatorial objects.

Definitions

Σ = finite set of relation symbols (R) with fixed arities (ρ(R)).

STR(Σ): finite Σ-relational structures $S = < D_S , (R_S)_{R \in \Sigma} >$,
R_S relation on D_S of arity ρ(R)

An MS transduction is a partial function

τ : STR(Σ) X "data" → STR(Γ) specified by MS formulas.

Basic case : τ : STR(Σ) → STR(Γ) ; $T = \tau(S)$ is defined "inside" S by MS formulas.

Examples : The edge complement ; the transitive closure of a directed graph.

Next case : $T = \tau(S, \text{“data”})$; the “data” is a tuple $X_1, \ldots, X_p$ of subsets of the domain of S ; these sets are called the parameters. Parameters $X_1, \ldots, X_p$ are constrained to satisfy an MS property.

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

$(G,X,Y,Z) \longmapsto$ the *minor* of G having vertex set X, resulting from the contraction of the edges of Y and the deletion of the edges and vertices of Z. (It is of type $MS_{2,2}$).

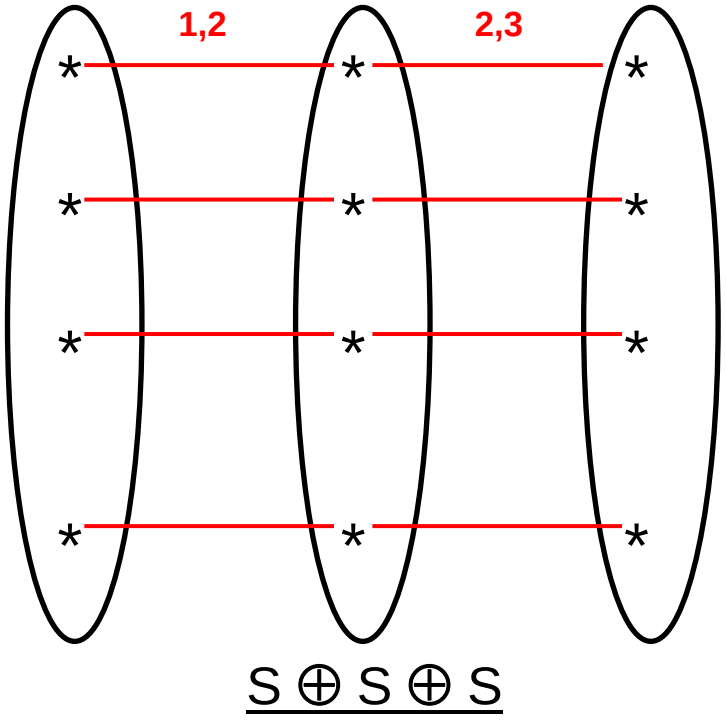
In the second example, no two vertices of X should be linked by a path of edges in Y.

$\tau(S)$: = the set of all $T = \tau(S, X_1, \ldots, X_p)$

for all “good” tuples of parameters.

General case : T is defined as above inside

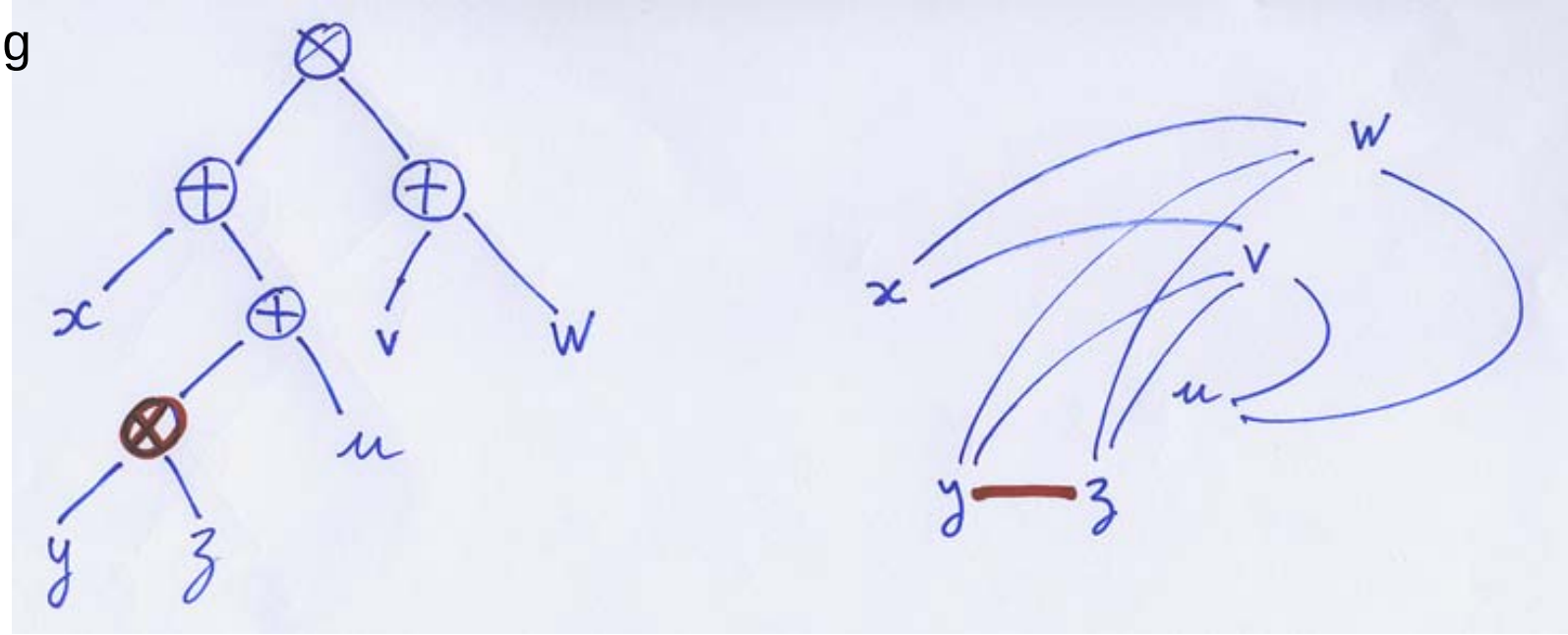
$S \oplus S \oplus \ldots \oplus S$: disjoint copies of S with "marked" equalities of copied elements



Composition Theorem : The composition of two MS transductions is an MS transduction.

Example 1 : From a term to a cograph

Terms are written with $\oplus$ (disjoint union), $\otimes$ (complete join) and constants x,y,z, … denoting vertices x,y,z ….

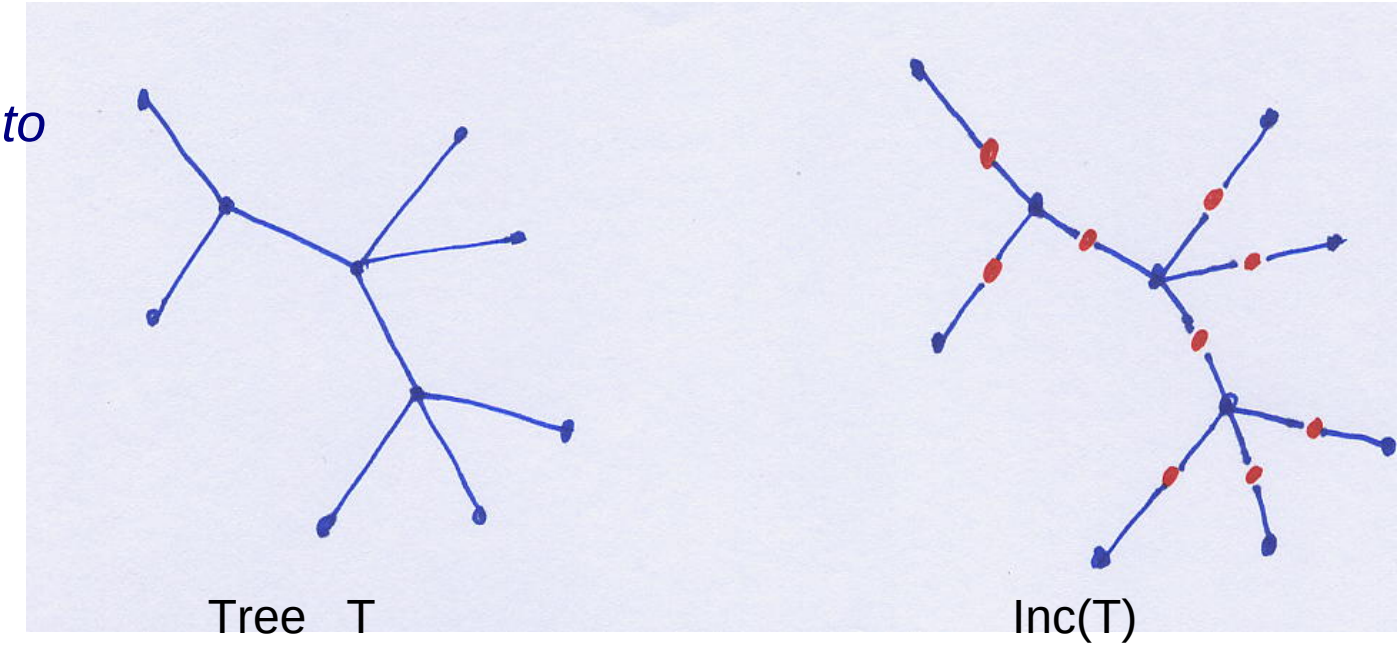


Vertices = {x,y,z,u,v,w } = occurrences of constants in the term.

Two vertices are adjacent if and only if their *least common ancestor* is labelled by $\otimes$ (like y and z , or u and w).

These conditions can be expressed by MS formulas on the labelled tree.

Example 2 : From a tree to its incidence graph (also a tree)



Tree T Inc(T)

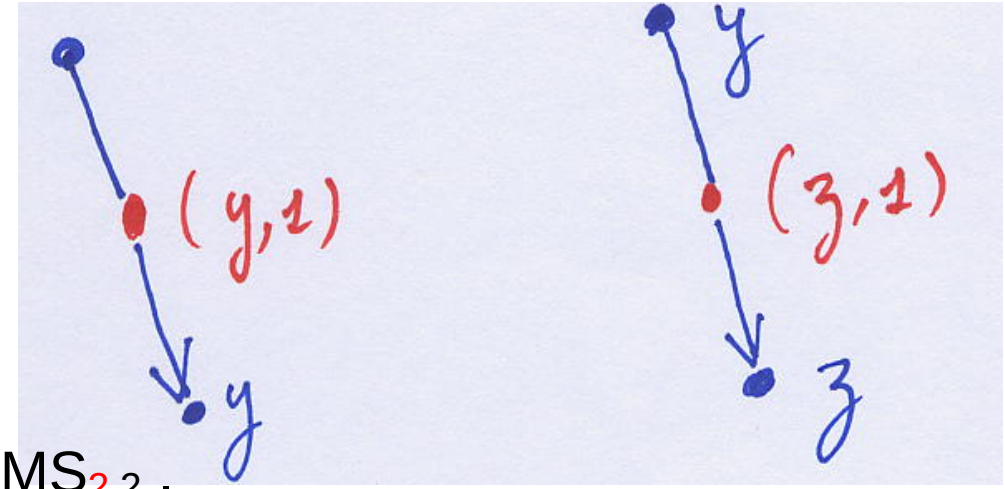
T = < N, edg> ; we use parameter { **r** } to make T rooted and directed

τ(T, { **r** }) = < N U (N - { **r** }) **x** { 1 } , inc(.,.) >

inc(x,y) is defined by :

x = (y,1) ∨ ∃ z [x = (z,1) ∧ edg(y,z)

∧ "y is on the path from **r** to z"]



From trees (or terms) to graphs :

$MS_{1,1} = MS_{2,1}$ and $MS_{1,2} = MS_{2,2}$.

$MS_{1,1}$ - transductions and $MS_{2,2}$ – transductions are incomparable

Why ? For expressing graph properties, MS_2 logic is more powerful than MS_1 logic (the "ordinary" MS logic).

For building graphs with $MS_{2,2}$ - transductions, we have *more* possibilities of using the input graph, but we want *more* for the output : to specify each edge as a copy of some vertex or some edge of the input graph.

Transitive closure is $MS_{1,1}$ but not $MS_{2,2}$

Edge subdivision is $MS_{2,2}$ but not $MS_{1,1}$

Proofs : Easy since, if S is transformed into T by an MS-transduction :

$$| D_T | \leq k. | D_S | \quad \text{for fixed } k$$

Robustness results : Preservation of widths

For every class of graphs C :

1) If C has tree-width $\leq$ k and τ is an $MS_{2,2}$ – transduction, then $\tau(C)$ has tree-width $\leq f_\tau(k)$

Follows from :

C has bounded tree-width $\Leftrightarrow$ $C \subseteq \tau$(Trees) for some $MS_{2,2}$ – transduction τ (the proof is constructive in both directions)

2) If C has clique-width $\leq$ k and τ is an $MS_{1,1}$ – transduction, then $\tau(C)$ has clique-width $\leq g_\tau(k)$.

Follows from :

C has bounded clique-width $\Leftrightarrow$ $C \subseteq \tau$(Trees) for some $MS_{1,1}$ – transduction τ (the proof is constructive)

Proof sketch for the logical characterization of bounded clique-width

1) A *k*-clique-width term is a rooted binary tree with each node labelled by one of the finitely many operations symbols using labels 1,…,k.

2) For each k, an MS-transduction can construct the defined graph from this labelled tree. (Extension of the proof given for cographs.)

Hence : If a graph class C has clique-width $\leq$ k , then $C \subseteq \tau_k(Trees)$

for some MS– transduction τ_k.

The converse uses technical tools from model theory (Fefermann-Vaught)

The proofs for tree-width are similar.

Gives easy proofs (but no good bounds) of facts like :

1) If C has bounded tree-width, its line graphs have bounded clique-width.

2) If C (directed graphs) has bounded tree-width or clique-width, the transitive closures of its graphs have bounded clique-width.

3) If C (directed graphs) has bounded clique-width, the transitive reductions of its graphs have bounded clique-width.

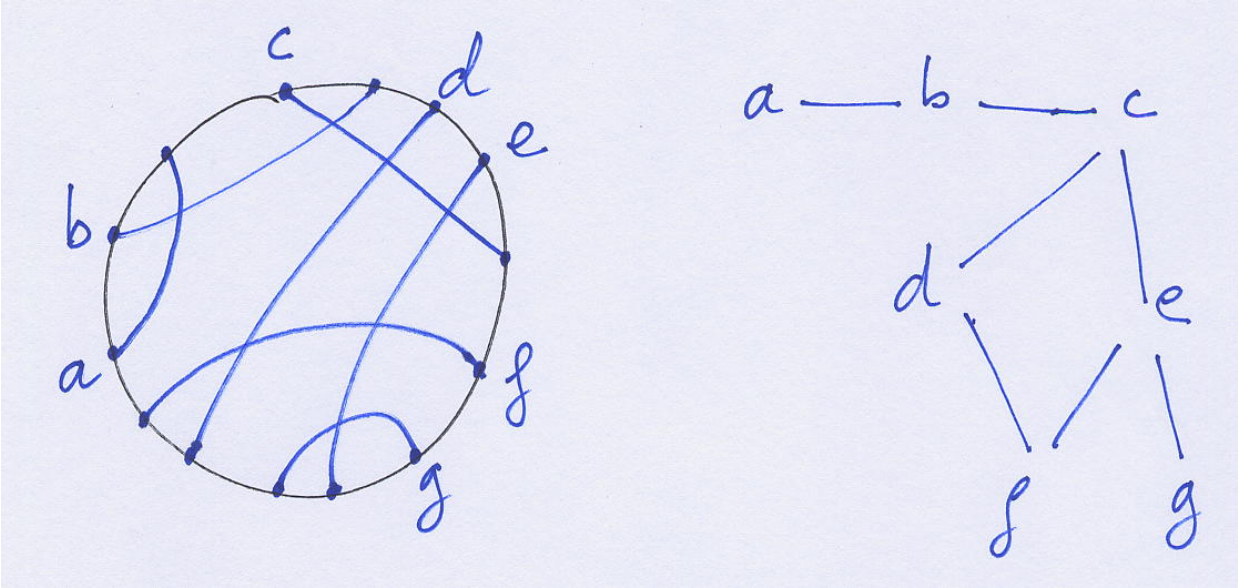
(Not trivial because clique-width is not monotone for subgraph inclusion).

4) The set of chordal graphs has *unbounded* clique-width

(because an MS transduction can define all graphs from chordal graphs, and graphs have *unbounded* clique-width).

5) k-leaf powers and similar "power" graphs of trees have bounded cwd

6) Circle graphs



Chord diagram Δ Circle graph $G(\Delta)$

Thm: Graphs Δ have bounded tree-width $\Leftrightarrow$ $G(\Delta)$ have bounded clique-width

1) $MS_{1,1}$ transduction from $G(\Delta)$ to Δ ;

2) Use "*split decomposition*" (Cunningham); $MS_{1,1}$ transduction from *prime* circle graphs to their unique chord diagrams.

Logical characterizations of equational sets

C is "tree-width" –equational $\Leftrightarrow$ $C = \tau(Trees)$ for some

$MS_{2,2}$ – transduction τ (For bounded tree-width we have $\subseteq$)

C is "clique-width" –equational $\Leftrightarrow$ $C = \tau(Trees)$ for some

$MS_{1,1}$ – transduction τ

Consequences : Closure of equational sets under the corresponding transductions.

(Extend robustness results for bounded widths).

Encoding powers of graph classes via MS transductions

An MS-transduction τ defines a graph H inside a graph G with help of parameters (sets of vertices or edges of G).

Say H is encoded in G : the encoding is represented by the parameters and τ is the decoding function.

The encoding powers of graph classes C and D can be compared as follows :

$C \leq D$ if $C \subseteq \tau(D)$ for some MS transduction τ

We get a *quasi-order on graph classes*.

We consider $MS_{2,2}$ - transductions : (formulas use edge set quantifications and must construct incidence graphs as outputs.)

For graph classes C and D we let :

$C \leq D$ if $C \subseteq \tau(D)$ for some $MS_{2,2}$ -transduction τ

$C \equiv D$ if $C \leq D$ and $D \leq C$

$C < D$ if $C \leq D$ and $C \not\equiv D$

$C <_c D$ if $C < D$ and there is no E with $C < E < D$

What is the structure of $<_c$ (the covering relation of $\leq$) *?*

With help of “Graph Minors 1 and 5 ” :

$\{\bullet\} < \text{Paths} <_c \text{Trees} <_c \text{Grids}$

These classes encode respectively :

finite sets,
sets of graphs of bounded path-width,
sets of graphs of bounded tree-width,
all sets of graphs .

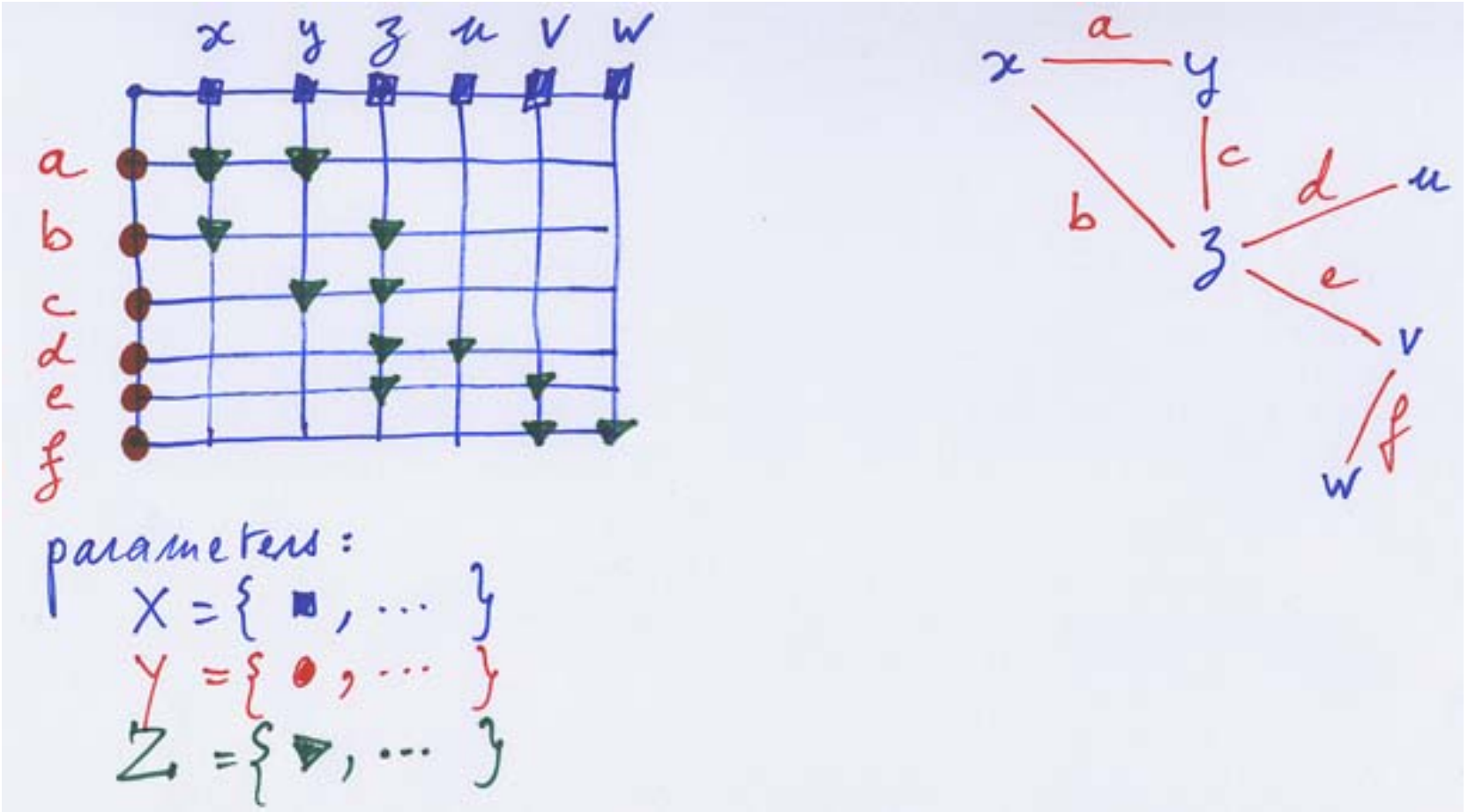
Proof : Trees $<_c$ Grids.

If a graph class C has bounded tree-width, it is $\leq$ Trees.

If C has unbounded tree-width, it contains all grids as minors,

hence : Grids $\leq C$ and Grids $\equiv C$, because Graphs $\leq$ Grids

Proof : All graphs $\leq$ Grids



A monadic second-order transduction using parameters X,Y,Z can transform all grids into all incidence graphs Inc(G).

More difficult : What is below Paths ?

Answer (A. Blumensath and B. C., Logic Colloquium 2008)

$\{\bullet\} <_c T_2 <_c \ldots T_n <_c T_{n+1} <_c \ldots <$ Paths $<_c$ Trees $<_c$ Square grids

where T_n is the class of rooted trees of height at most n (and unbounded degree).

Idea : T_n encodes the classes of graphs having tree-decompositions of height at most n and width at most k (for all k).

Definition : *n-depth tree-width of G* = $twd_n(G)$ = minimal width of a tree-decomposition of G of height at most n.

Related notion : *tree-depth* (Nesetril , Ossona de Mendez).

td(G) = minimal k such that each conn. comp. of G has a depth-first (normal) spanning tree of height at most k.

Some properties of these variants of tree-width :

1) $pwd(G) \leq n.(twd_n(G) + 1)$

2) If G is a minor of H : $twd_n(G) \leq twd_n(H)$, $td(G) \leq td(H)$

3) $td(G) \leq n$ implies $twd_n(G) \leq n$,

4) $twd_n(G) \leq k$ implies $td(G) \leq n.k$

Excluded Path Theorem

(cf. Excluded Tree and Grid Theorems of GM1 and GM5)

A class of graphs C excludes some path as a minor (equivalently, as a subgraph)

$\Leftrightarrow$ for some n, C has bounded n-depth tree-width

$\Leftrightarrow$ C has bounded tree depth.

We use n-depth tree-width rather than tree-depth to characterize the graph classes encoded by trees of each height

Logical properties of n-depth tree-width.

Proposition : For each n and k, there exists an $MS_{2,2}$-transduction that maps every graph of n-depth tree-width at most k *to all its strict* tree-decompositions of height at most n and width at most k

(*strict* = with certain connectivity properties ; every tree-decomposition can be made strict without increasing height and width).

Remark : The obstruction sets of graphs for n-depth tree-width $\leq$ k are computable from each pair n, k because we have monadic second-order characterizations of these classes and bounds on the tree-widths of the obstruction sets.

The same holds for the property " tree-depth $\leq$ k" .

In the hierarchy :

$$\{\bullet\} <_c T_2 <_c \ldots <_c T_n <_c \ldots < \text{Paths} <_c \text{Trees} <_c \text{Grids}$$

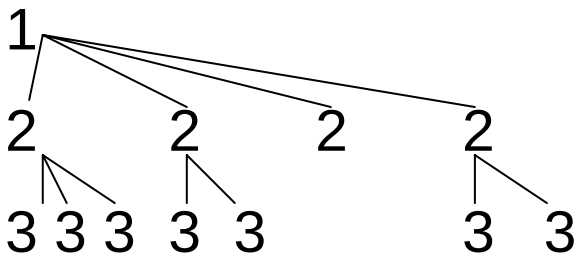
each level T_n encodes the sets of graphs of bounded n-depth tree-width.

Proofs to be done :

1) $T_n \leq$ Paths

Trees of height n can be encoded as sequences over [n] and decoded by MS-transductions.

1 2 333 2 33 2 2 33 encodes the tree :



2) $T_n < T_{n+1}$

One cannot define by an MS-transduction all trees of height n+1 from all trees of height n.

The (technical) proof uses analysis of MS definable relations on trees and some counting arguments.

Case n = 2.

Trees of height 2 correspond (via MS transductions) to sets (without relations).

If a k-copying MS-transduction with p parameters transforms sets into trees, these trees have less than $k.2^p$ internal nodes. We cannot get all trees of height 3 from sets by a single MS-transduction.

3) Hence, we cannot have $T_n \equiv$ Paths

"Dichotomy arguments" :

1) Let C be a set of bounded pathwidth (i.e., $C \leq$ Paths):

Either : it contains all paths as minors, then $C \equiv$ Paths

Or : (Excluded Path Thm) $twd_n(C)$ is bounded and $C \leq T_n$ for some n

2) Let C be a set of n-depth tree-width $\leq$ k ($C \leq T_n$):

Either : for all m, there is G in C s.t., for each n-depth tree-dec. U of width k of G , the tree U contains T(n,m) (T(n,m) = the m-ary complete tree of height n) and then $T_n \leq C$ (because n-depth tree-decompositions of width k are definable by MS transductions)

Or : for some m, every G in C has an n-depth tree-dec. U of width k, s.t. U does not contain T(n,m). By contracting some edges of U, one gets an (n-1)-depth tree-dec. of G of width m.(k+1), hence $C \leq T_{n-1}$.

Open question : What about the hierarchy based for

$MS_{1,1}$ – transduction ?

Theorem (B.C. & Oum, 2007) :

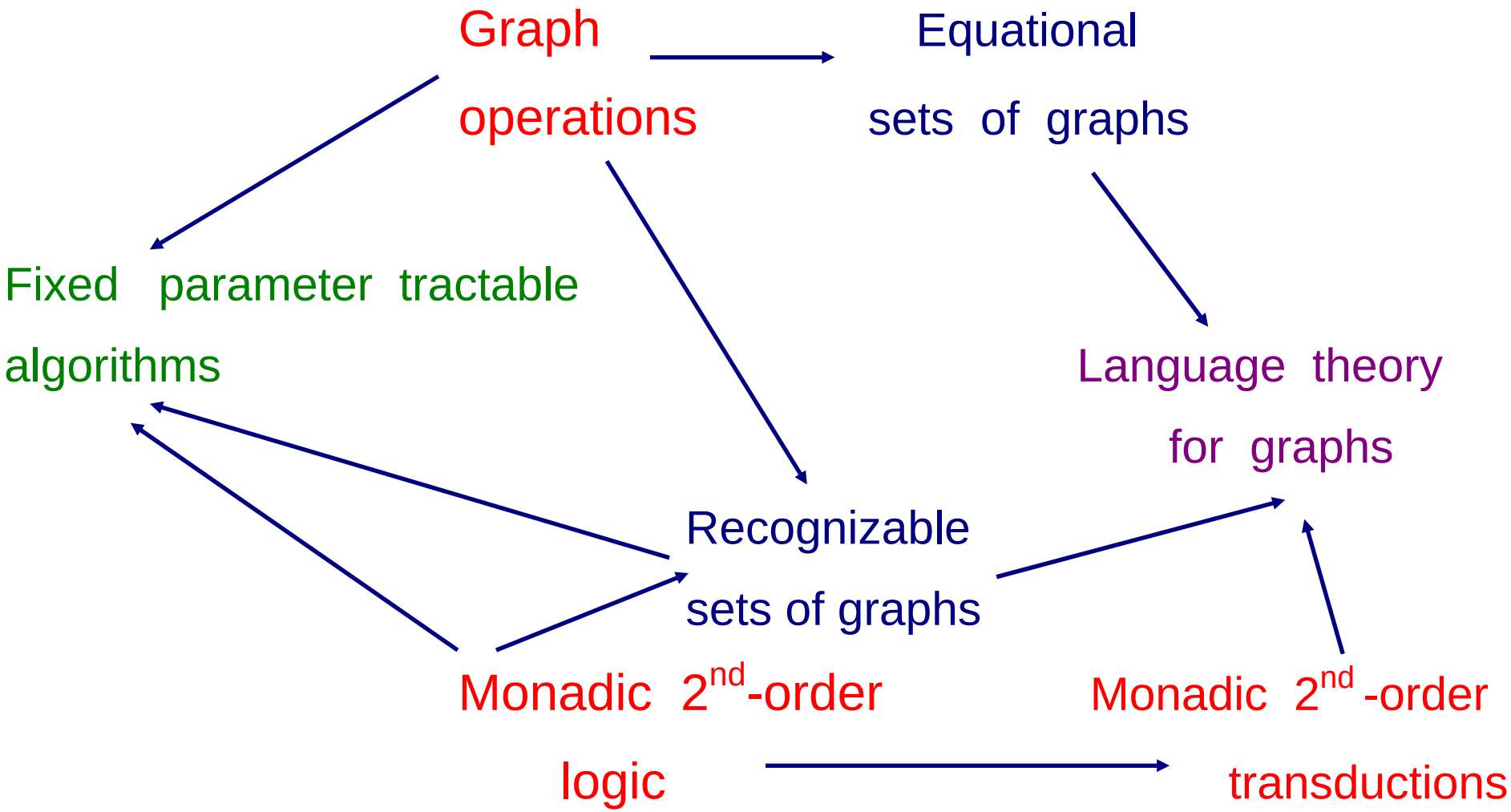
There exists an $MS_{1,1}$ - transduction (using *even cardinality set predicates*) that transforms every set of undirected graphs of unbounded rank-width into the set of all square grids.

(Uses *vertex-minors* instead of minors)

We need a result corresponding to GM1 about “linear rank-width” and excluding a forest as a *vertex-minor*.

We need also something like “n-depth rank-width” and constructions by MS tranductions of appropriate rank-decompositions.

Conclusion : The overview chart



Appendix : The fundamental property of MS transductions :

$$S \mid\longrightarrow \tau(S)$$

$$\tau^{\#}(\psi) \longleftarrow \mid \psi$$

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

$$S \mid= \tau^{\#}(\psi) \quad \text{if and only if} \quad \tau(S) \mid= \psi$$

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

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