The Complexity of the List Homomorphism Problem for Graphs

L. Egri, [1] A. Krokhin, [2] B. Larose, [3] P. Tesson [4]

[1] School of Computer Science
McGill University, Montréal

[2] School of Engineering and Computing Sciences
Durham University, UK

[3] Department of Mathematics and Statistics
Concordia University, Montréal

[4] Département d'informatique et de génie logiciel
Université Laval, Québec

Workshop on Graph Homomorphisms, Fields Institute,
July 2011

Overview

We completely classify the computational complexity of the list **H**-colouring problem for graphs:

- in combinatorial and algebraic terms;
- descriptive complexity equivalents are given as well via Datalog and its fragments;
- for every graph **H**, the problem is either
  - NP-complete,
  - NL-complete,
  - L-complete or
  - first-order definable.

Overview, continued

- Motivation: our algebraic characterisations match general complexity conjectures on constraint satisfaction problems;
- Metaproblem: the procedure to identify in which class a graph belongs is efficient.

Preliminaries

Definition

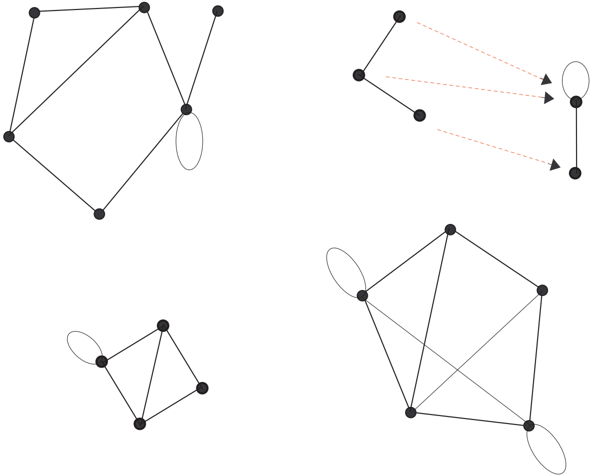
A *graph* is a structure $\mathbf{H} = \langle H; \theta \rangle$ with a single binary relation θ which is symmetric: $(a, b) \in \theta$ iff $(b, a) \in \theta$.

Remark: Our graphs may have loops on certain vertices.

Definition

A *graph homomorphism* is an edge-preserving map between two graphs. Formally, f is a homomorphism $f : \mathbf{G} \to \mathbf{H}$ if $(f(u), f(v))$ is an edge of $\mathbf{H}$ for every edge (u, v) of $\mathbf{G}$.

Pictures of graphs and homomorphisms

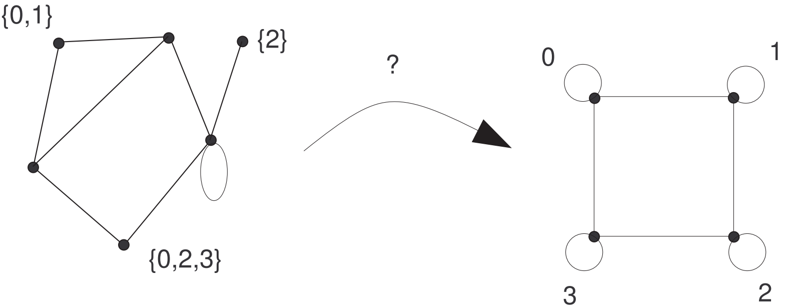


List Homomorphism Problems

Given a graph **H**, the *list homomorphism problem for* **H** is:
$CSP(\mathbf{H} + lists)$

- Input: a graph **G**, and for each vertex v of **G** a list L_v of vertices of **H**;
- Question: is there a homomorphism $f : \mathbf{G} \to \mathbf{H}$, such that $f(v) \in L_v$ for all $v \in \mathbf{G}$?

List Homomorphism Problems, cont'd



Motivation and Background

- Our main motivation is a series of general conjectures that predict the (descriptive) complexity of Constraint Satisfaction Problems based on the properties of their associated algebra;
- Since part of the proof of our results relies on the algebraic and descriptive complexity approach, we give a brief overview of these.

CSPs (homomorphism form)

Definition

Let $\mathbf{G} = \langle G; \rho_1, \cdots, \rho_s \rangle$ and $\mathbf{H} = \langle H; \theta_1, \cdots, \theta_s \rangle$ be similar relational structures. A *homomorphism* from $\mathbf{G}$ to $\mathbf{H}$ is a relation preserving map $f : G \to H$, i.e. such that $f(\rho_i) \subseteq \theta_i$ for each $1 \leq i \leq s$.

Definition ($CSP(\mathbf{H})$)

Let $\mathbf{H}$ be a structure.

$CSP(\mathbf{H}) = \{\mathbf{G} : \mathbf{G} \to \mathbf{H}\}$, i.e.

- Input: a structure $\mathbf{G}$ similar to $\mathbf{H}$;
- Question: is there a homomorphism from $\mathbf{G}$ to $\mathbf{H}$?

CSP Classification Problems

Two main classification problems about problems $CSP(\mathbf{H})$:

1. Classify $CSP(\mathbf{H})$ w.r.t. *computational complexity*, i.e., w.r.t. membership in a given complexity class (e.g. P, NL, L), modulo assumptions like P $\neq$ NP
2. Classify $CSP(\mathbf{H})$ w.r.t. *descriptive complexity*, i.e., w.r.t. definability of $CSP(\mathbf{H})$ in a given logic (FO, Datalog and its fragments - linear, symmetric)

In addition, there is a meta-problem:

- Determine the complexity of deciding whether $CSP(\mathbf{H})$ has given (computational or descriptive) complexity.

Datalog

- A *Datalog Program* consists of *rules*, and takes as input a relational structure.
- a typical Datalog rule might look like this one:

$$\theta_1(x, y) \leftarrow \theta_2(w, u, x), \theta_3(x), R_1(x, y, z), R_2(x, w)$$

- the relations R_1 and R_2 are basic relations from the input structures (EDBs);
- the relations θ_i are auxiliary relations (IDBs);
- the rule stipulates that if the condition on the righthand side (the *body* of the rule) holds, then the condition of the left (the *head*) should also hold.

2-colouring

- Let $\mathbf{H} = \langle \{0,1\}; E = \{(0,1),(1,0)\}\rangle$ be the complete graph on 2 vertices. Clearly $CSP(\mathbf{H})$ is just the 2-colouring problem.
- We describe a Datalog program that accepts precisely those graphs that *cannot* be 2-coloured.
- It uses a single binary auxiliary relation (IDB) we'll denote *OddPath*.

A Datalog program for 2-colouring

A Datalog program recursively computes the auxiliary relations (IDBs).
Intuition: locally derive new constraints, trying to get a contradiction (to certify that there's no solution).

$$\begin{aligned} OddPath(x,y) &\leftarrow E(x,y) \\ OddPath(x,y) &\leftarrow OddPath(x,z), E(z,u), E(u,y) \\ \gamma &\leftarrow OddPath(x,x) \end{aligned}$$

The 0-ary relation γ is the *goal predicate* of the program: it "lights up" precisely if the input structure admits NO homomorphism to the target structure $\mathbf{H}$.

Fragments

A Datalog program is *linear* if each rule contains at most one occurrence of an IDB in the body, i.e. if each rule looks like this

$$\theta_1(x, y) \leftarrow \theta_2(w, u, x), R_1(x, y, z), R_2(x, w)$$

where the θ_i's are the only IDBs in it.

A linear Datalog program is *symmetric* if it is invariant under symmetry of rules, i.e. if the program contains the above rule, then it must also contain its *symmetric*:

$$\theta_2(w, u, x) \leftarrow \theta_1(x, y), R_1(x, y, z), R_2(x, w).$$

Definability in Datalog (and fragments)

We say that $\neg CSP(\mathbf{H})$ is *definable in (linear, symmetric) Datalog* if there exists a (linear, symmetric) Datalog program that accepts precisely those structures that do not admit a homomorphism to $\mathbf{H}$.

Facts:

- $\neg CSP(\mathbf{H})$ definable in Datalog $\Rightarrow$ $CSP(\mathbf{H}) \in$ P;
- $\neg CSP(\mathbf{H})$ definable in lin. Dat. $\Rightarrow$ $CSP(\mathbf{H}) \in$ NL;
- $\neg CSP(\mathbf{H})$ definable in sym. Dat. $\Rightarrow$ $CSP(\mathbf{H}) \in$ L.

The converse of the last two statements holds for all CSPs known to belong to NL and L.

Definability in Datalog, cont'd

The 3 fragments constitute a strict hierarchy, and there are CSPs in P not expressible in Datalog:

- LinEq(mod 2) belongs to P, but not definable in Datalog;
- Horn 3-Sat is def in Datalog, but not in lin. Datalog;
- Directed *st*-Conn is in lin., but not sym. Datalog.

Example: 2-col is in Symmetric Datalog

The program we described to solve 2-colouring can be symmetrised, i.e. we can safely add the symmetric of every rule without changing the outcome;

$$\begin{array}{rcl} OddPath(x,y) & \leftarrow & E(x,y) \\ OddPath(x,y) & \leftarrow & OddPath(x,z), E(z,u), E(u,y) \\ OddPath(x,z) & \leftarrow & OddPath(x,y), E(z,u), E(u,y) \\ \gamma & \leftarrow & OddPath(x,x) \end{array}$$

Hence, 2-colouring is solvable in Logspace.

Polymorphisms

- A *polymorphism* of $\mathbf{H}$ is a homomorphism $f : \mathbf{H}^n \to \mathbf{H}$; we denote by $Pol(\mathbf{H})$ the set of all polymorphisms of $\mathbf{H}$.
- If $\mathbf{H}$ is a graph: an *edge-preserving* mapping, i.e.

$$\begin{array}{cccc} a_1 & a_2 & \dots & a_n \\ | & | & \dots & | \\ b_1 & b_2 & \dots & b_n \end{array} \quad \Rightarrow \quad \begin{array}{c} f(a_1, a_2, \dots, a_n) \\ | \\ f(b_1, b_2, \dots, b_n) \end{array}$$

- For $\mathbf{H}$ + *lists*: the above + *conservativity*, i.e. $\forall x_1, \dots, x_n \quad f(x_1, x_2, \dots, x_n) \in \{x_1, x_2, \dots, x_n\}$.

The algebra $\mathbb{A}(\mathbf{H})$

Definition

Let $\mathbf{H}$ be a relational structure. The algebra associated to $\mathbf{H}$ is defined as $\mathbb{A}(\mathbf{H}) = \langle H; Pol(\mathbf{H})\rangle$.

Fact (Bulatov, Jeavons, Krokhin '05 + L, Tesson '09)

The (computational and descriptive) complexity of $CSP(\mathbf{H})$ is completely determined by the properties of $\mathbb{A}(\mathbf{H})$.

BJK+LT prove that specific properties of $\mathbb{A}(\mathbf{H})$ that are necessary (and conjecture that they are sufficient) for:

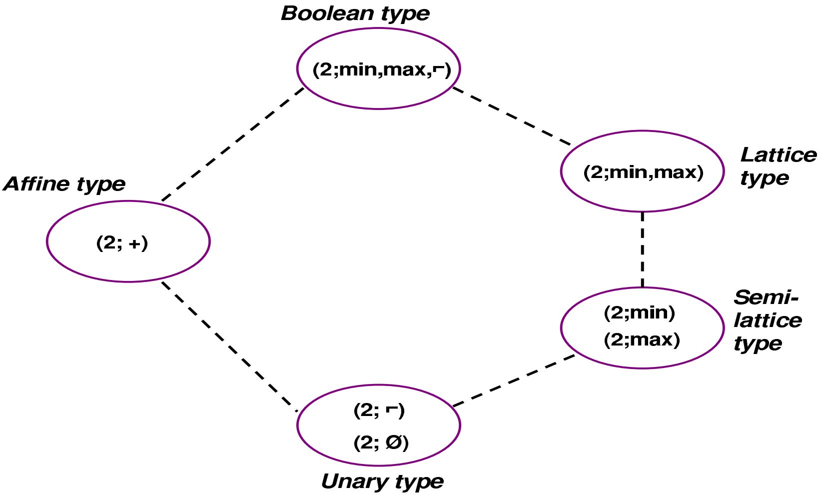
- $CSP(\mathbf{H})$ to be in P,
- $CSP(\mathbf{H})$ to be in NL & definable in Linear Datalog,
- $CSP(\mathbf{H})$ to be in L & definable in Symmetric Datalog.

The Five Types (in Conservative Algebras)

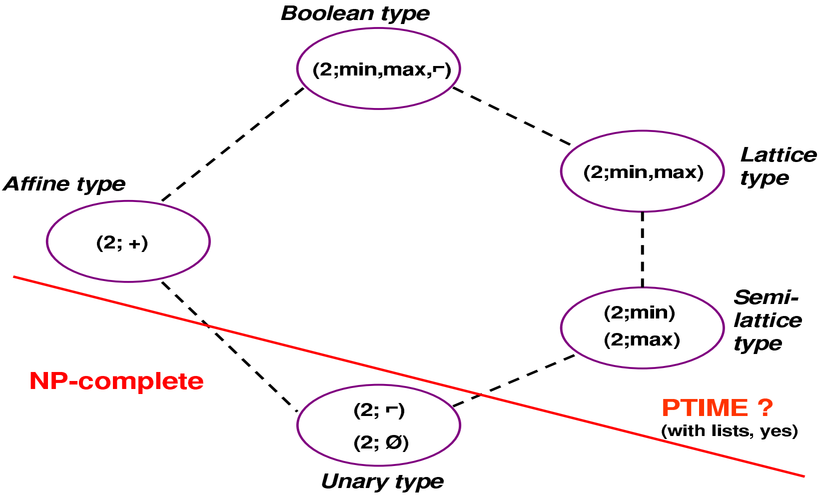
Consider the problem $CSP(\mathbf{H} + lists)$; its associated algebra is conservative, denote it by $\mathbb{A}$.
Let $X = \{0,1\}$ be an arbitrary 2-element subset of H.
The set X can be assigned (in $\mathbb{A}$) one of *five types*:
By Post'41, there exist only *five possibilities* for the set $\{f(x_1,\ldots,x_n,0,1) \mid f = g_{|\{0,1\}}, g \in Pol(\mathbf{H} + lists)\}$:

1. essentially unary op's $s(x_1,\ldots,x_n) = t(x_i)$ *unary*
2. all linear Boolean op's $\sum a_i x_i + a_0$ $(mod\ 2)$ *affine*
3. all possible Boolean operations *Boolean*
4. all monotone Boolean operations *lattice*
5. all op's of the form $\min(x_1,\ldots,x_n)$ and 0,1 *semilattice*

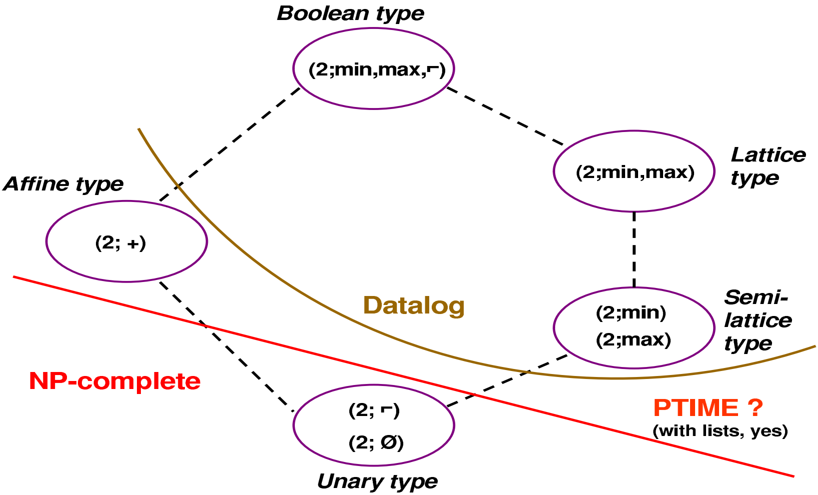
Ordering of Types



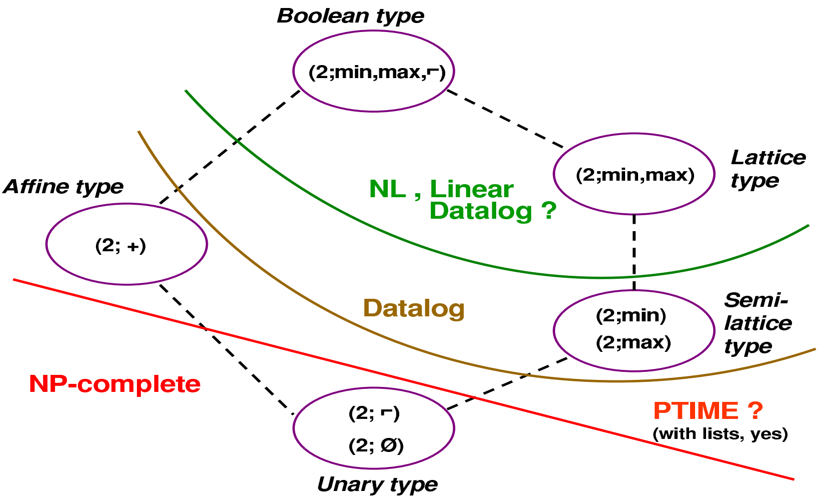
Types and Conjectures



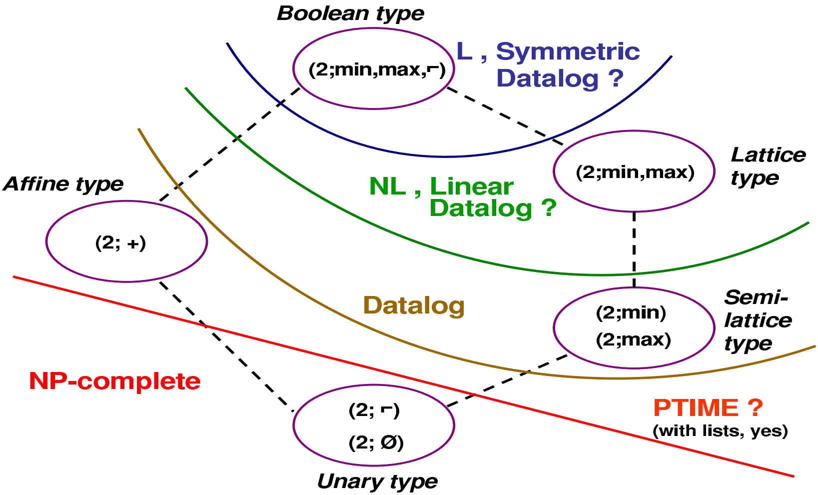
Types and Conjectures



Types and Conjectures



Types and Conjectures

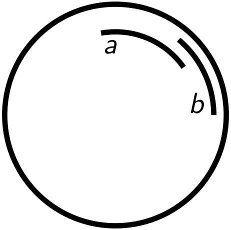


Some Classification Results

- Schaefer '78 – each Boolean $CSP(\mathbf{H})$ (aka generalised $\textsc{Sat}$) is in P or NP-complete;
- Allender et al. '09 + L, Tesson '09 –
  each Boolean $CSP(\mathbf{H})$ is in AC^0 or else complete for one of the following classes: NP, P, NL, $\oplus$L, L.
  Also classification wrt definability in FO and (fragments of) Datalog;
- Bulatov '03 – algebraic characterisation of list CSPs in P (= omitting the unary type);
- Barto, Kozik '09 – algebraic characterisation of CSPs definable in Datalog (= omitting the unary and affine types).

Bi-arc graphs

- (Feder, Hell, Huang) a graph $\mathbf{H}$ is *bi-arc* iff $\mathbf{H} \times \mathbf{K}_2$ is the complement of a *circular arc graph*:
- bi-arc means: vertices are arcs, and vertices are adjacent if the corresponding arcs intersect.
- Ex: odd cycles and the 6-cycle are NOT bi-arc graphs.



Statement of Main Result

Full fine-grained classification of $CSP(\mathbf{H} + lists)$ for graphs.

Theorem

Let $\mathbf{H}$ *be a graph. Then the following holds.*

- *If* $\mathbf{H}$ *is not bi-arc then* $CSP(\mathbf{H} + lists)$ *is* NP*-complete and* $\neg CSP(\mathbf{H} + lists)$ *is not definable in Datalog;*
- *if* $\mathbf{H}$ *is bi-arc, but not in class* $\mathcal{L}$ *(def later), then* $CSP(\mathbf{H} + lists)$ *is* NL*-complete. Also,* $\neg CSP(\mathbf{H} + lists)$ *is definable in linear, but not in symmetric, Datalog;*
- *If* $\mathbf{H}$ *is in* $\mathcal{L}$ *then* $CSP(\mathbf{H} + lists)$ *is in* L *and* $\neg CSP(\mathbf{H} + lists)$ *is definable in Symmetric Datalog.*
- *Everything matches the algebraic conjectures.*

Proof: Step 1

Definition

A 3-ary operation $M : H^3 \to H$ is a *majority* operation if it satisfies $M(x,x,y) = M(x,y,x) = M(y,x,x) = x$ for all $x \in H$.

Our starting point is the following dichotomy result:

Theorem (Feder, Hell, Huang, 1999)

Let $\mathbf{H}$ *be a graph. Then t.f.a.e.:*

1. $\mathbf{H} +$ *lists admits a majority operation;*
2. $\mathbf{H}$ *is a bi-arc graph.*

If this condition is satisfied then $CSP(\mathbf{H} + \textit{lists})$ *is in* P*, otherwise it is* NP*-complete.*

Proof: Step 2

From FHH: every bi-arc graph admits a (conservative) majority operation.

Theorem (Dalmau, Krokhin, 2008)

If $\mathbf{H}$ *is invariant under a majority operation then* $\neg CSP(\mathbf{H})$ *is expressible in linear Datalog; in particular* $CSP(\mathbf{H})$ *is in* NL.

Hence list-homomorphism problem for bi-arc graphs is in NL and expressible in linear Datalog.

Proof: Step 3

What now ?

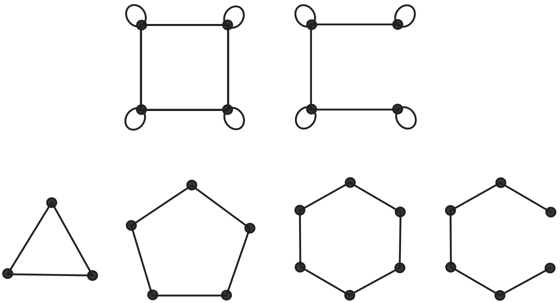
- from the above, the algebra associated to any bi-arc graph omits the unary, affine and semilattice types;
- if the algebra *admits* the lattice type, then the CSP is NL-complete;
- we sieve to find all these graphs and see what remains.

The class $\mathcal{L}$ by forbidden subgraphs

Definition (Version 1)

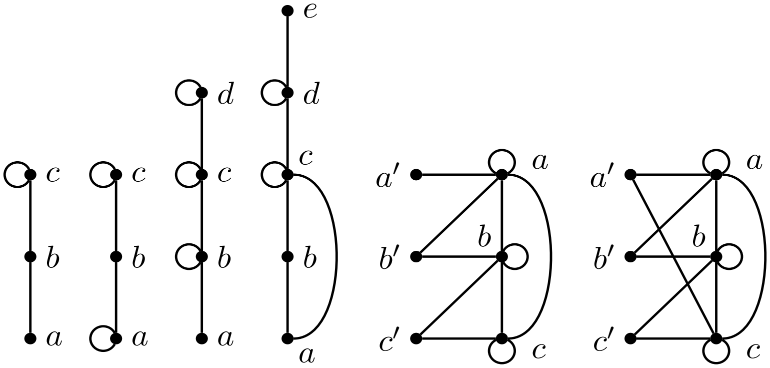
H is in $\mathcal{L}$ if it avoids as induced subgraph every of the following 12 forbidden graphs:

The 2 reflexive and 4 irreflexive bad guys ...



The class $\mathcal{L}$ by forbidden subgraphs, cont'd

... and the 6 mixed bad guys:

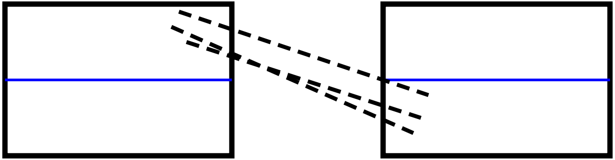


Proof: Step 4

- Hopefully we have found *all* bad guys, i.e. no graph in $\mathcal{L}$ admits the lattice type;
- if this holds, conjectures predict the CSP is in symm Datalog;
- unfortunately, the graphs in $\mathcal{L}$ are defined by a negative condition, which is useless to prove this;
- We're in luck: this family of graphs admits a very nice inductive definition !

The class $\mathcal{L}$ by inductive definition

- first we consider only irreflexive graphs:
- define the *special sum* of two bipartite graphs $\mathbf{H}_1$ and $\mathbf{H}_2$ as follows: connect every vertex of one colour class of $\mathbf{H}_1$ to every vertex of one colour class of $\mathbf{H}_2$:



$\mathbf{H}_1$ $\mathbf{H}_2$

The class $\mathcal{L}$ by inductive definition, cont'd

Lemma

Let $\mathbf{H}$ *be an irreflexive graph. Tfae:*

- $\mathbf{H}$ *is obtained from one-element graphs using disjoint union and special sum;*
- $\mathbf{H}$ *is bipartite, and avoids the 6-cycle and 5-path;*
- $\mathbf{H} \in \mathcal{L}$.

The class $\mathcal{L}$ by inductive definition, cont'd

A connected graph H is *basic* if it is an irreflexive graph in $\mathcal{L}$ or is obtained from one by turning one colour class into a reflexive clique.

The graph $H_1 \oslash H_2$ is obtained from the disjoint union of the two graphs by connecting every loop in H_1 to every vertex in H_2.

Lemma

The class $\mathcal{L}$ is the smallest class $\mathcal{C}$ of graphs such that:

1. *$\mathcal{C}$ contains the basic graphs;*
2. *$\mathcal{C}$ is closed under disjoint union;*
3. *if H_1 is a basic graph and $H_2 \in \mathcal{C}$ then $H_1 \oslash H_2 \in \mathcal{C}$.*

The class $\mathcal{L}$

Theorem

Let $\mathbf{H}$ *be a graph, and let* $\mathbb{A}$ *be the algebra associated to* $\mathbf{H} + \textit{lists}$*. Then t.f.a.e.:*

1. $\mathbf{H} \in \mathcal{L}$;
2. $\mathcal{V}(\mathbb{A})$ *admits only the Boolean type;*
3. $\mathcal{V}(\mathbb{A})$ *is 4-permutable;*
4. $\neg CSP(\mathbf{H} + \textit{lists})$ *is expressible in symmetric Datalog.*

If these conditions hold then $CSP(\mathbf{H} + \textit{lists})$ *is in* L*; otherwise it is* NL*-complete (and* $\neg CSP(\mathbf{H} + \textit{lists})$ *is expressible in linear Datalog) or it is* NP*-complete.*

FO-definable Problems $CSP(\mathbf{H} + lists)$

Theorem (L, Tesson' 09)

Every problem $CSP(\mathbf{H})$ is either FO-definable or else L*-hard under FO-reductions.*

Theorem

For a graph $\mathbf{H}$, $CSP(\mathbf{H} + lists)$ is FO-definable iff the following holds:

- *the loops in $\mathbf{H}$ form a clique,*
- *the non-loops in $\mathbf{H}$ form an independent set,*
- *the non-loops can be ordered $v_1 \ldots v_n$ so that $N(v_i) \subseteq N(v_{i+1})$ for all $i = 1 \ldots n-1$.*

Meta-Problem

Theorem

Given a graph $\mathbf{H}$, *it can be decided in polynomial-time what computational and descriptive complexity* $CSP(\mathbf{H} + \textit{lists})$ *has.*

- $\mathbf{H}$ is bi-arc iff $\overline{\mathbf{H} \times \mathbf{K}_2}$ is circular arc. Circular arc graphs can be recognised in poly-time (McConnell '03)
- The class $\mathcal{L}$ is defined by a finite number of forbidden induced subgraphs, hence poly-time recognition.
- Structures with FO-definable CSPs can be recognised in poly-time (L, Loten, Tardif '06).

Sieve: an example

An illustration: Why the 5-path is bad:

- the 5-path is a bi-arc graph, so admits a majority operation and hence $\mathcal{V}(\mathbb{A})$ omits types 1, 2 and 5;
- we produce (by pp-definability) a 2-element subalgebra with monotone terms;
- hence this divisor is of type 4.