Posets, graphs and algebras: a case study for the fine-grained complexity of CSP's

Part 1: Preliminaries on Complexity and CSP's

Benoit Larose [1] [2]

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

[2]Department of Mathematics
Champlain Regional College, St-Lambert

SSAOS, Slovakia, September 2009

Overview: some history

- In the mid 90's, an important connection was made (Feder & Vardi, Jeavons) relating
    - the complexity of Constraint Satisfaction Problems
    - the nature of the operations that preserve the constraint relations
- early 21st century: the connection between algebra and complexity is made clearer by Bulatov, Jeavons, Krokhin
- this has led in the last few years to major advances in our understanding of CSP's

Overview: what's a CSP ?

- A Constraint Satisfaction Problem consists of:
  - a finite set of variables,
  - a set of constraints on these variables,
  - a set of possible values for the variables;

  the problem is to decide whether we can assign values to the variables so as to satisfy all the constraints.
- typical examples from "real life" are:
  - Sudoku and crossword puzzles,
  - database queries, scheduling problems, etc.
- not so real-life examples are:
  - graph colouring,
  - graph reachability,
  - 3-SAT, Horn SAT, 2-SAT, etc.

Overview: Dichotomy Conjectures

- **Dichotomy Conjecture** (Feder & Vardi, 1994): every (fixed target) CSP is either solvable in poly-time, or is NP-complete;
- in 2000, BJK refined the conjecture in algebraic terms: to each CSP, one associates an algebra $\mathbb{A}$; the identities satisfied by the algebra should control the tractability of the CSP:
- **Algebraic Dichotomy Conjecture**: if the variety generated by $\mathbb{A}$ omits type 1, then the CSP is tractable, otherwise it is NP-complete.

Overview: some evidence

- the conjecture has been verified in many special cases, in particular:
    - 2 elements (Schaefer, 1978)
    - 3 elements (Bulatov, 2002)
    - list homomorphism problems (Bulatov, 2003)
    - various other special cases (graphs, etc.)

Overview: refining the Boolean case

- In 2005, Allender, Bauland, Immerman, Schnoor, Vollmer obtain a complete classification of the complexity of Boolean CSP's;
- all Boolean CSP's satisfy one of the following conditions:
  - in AC^0;
  - $\mathcal{L}$-complete;
  - $\mathcal{NL}$-complete;
  - $\oplus\mathcal{L}$-complete;
  - $\mathcal{P}$-complete;
  - $\mathcal{NP}$-complete.
- the above are all standard complexity classes.

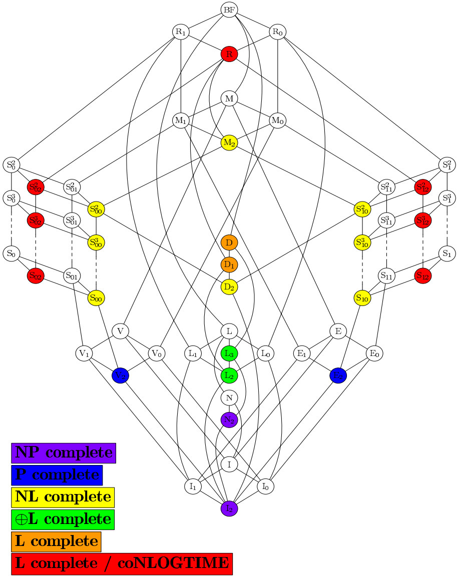


Figure 1: Graph of all closed classes of Boolean functions

Overview: Fine-grained Complexity Conjectures

- Remarkably, the classification of the complexity of Boolean CSP's lines up perfectly with
  - the typeset of the variety of the associated algebra
  - the (non-)expressibility of the CSP in various logics
- It appears that the algebra not only predicts which CSP's are easy or hard, but in fact the "exact" complexity of the CSP;
- precise conjectures have been formulated that relate the complexity (both descriptive and algorithmic) with the nature of the identities of the associated algebra

Outline of the talks

- Part 1: Preliminaries on Complexity and CSP's
- Part 2a: Preliminaries on Algebra and
  Statement of the Conjectures
- Part 2b: Some Evidence: General Results
- Part 3: More Evidence: Graphs and Posets

Homework ?!?

- take a glance at last year's talks by
  - A. Krokhin (Algebraic approach to CSP's)
  - R. Willard (Computational complexity)

Relational Structures

- a k-ary *relation* on a set H is a subset of H^k, i.e. a set of k-tuples;
- a *relational structure*

$$\mathbf{H} = \langle H; \theta_1, \theta_2, \ldots, \theta_r \rangle$$

  consists of a non-empty set H (its *universe*) and some relations θ_i on H.
- for instance, (di)graphs are relational structures $\mathbf{H} = \langle H; \theta \rangle$ with a single binary relation (the edges.)

Homomorphisms

- Let $\mathbf{G} = \langle G; \rho_1, \rho_2, \ldots, \rho_s \rangle$ and $\mathbf{H} = \langle H; \theta_1, \theta_2, \ldots, \theta_r \rangle$ be relational structures.
- $\mathbf{G}$ and $\mathbf{H}$ have the same *signature* if $r = s$ and arity of $\rho_i =$ arity of θ_i for all $1 \leq i \leq r$.
- a map $f : G \to H$ is a *homomorphism* if $f(\rho_i) \subseteq \theta_i$ for all i;
- if $\mathbf{G}$ and $\mathbf{H}$ are graphs, a homomorphism is just an edge-preserving map, i.e.
  (u, v) is an edge of $\mathbf{G} \implies (f(u), f(v))$ is an edge of $\mathbf{H}$.

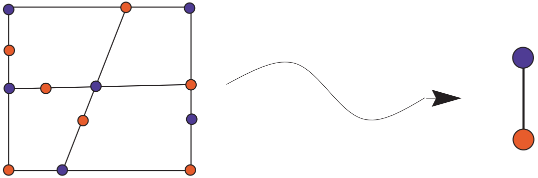
$CSP(\mathbf{H})$

- fix a "target" structure $\mathbf{H}$;
- the problem $CSP(\mathbf{H})$:
  1. Instance: a structure $\mathbf{G}$;
  2. Question: is there a homomorphism $\mathbf{G} \to \mathbf{H}$?

- i.e. $CSP(\mathbf{H})$ is the set of structures that admit a homomorphism to $\mathbf{H}$;
- $\neg CSP(\mathbf{H})$ = structures that do NOT admit a homomorphism to $\mathbf{H}$.

An Example: 2-Colouring

- let $\mathbf{G}$ be a graph;
- it is easy to see that
  proper 2-colouring of $\mathbf{G}$ = homomorphism $\mathbf{G} \rightarrow$ edge;



- In particular: 2-COL is the problem $CSP(\mathbf{H})$ where
  $\mathbf{H} = \langle\{0,1\}; E\rangle$ with $E = \{(0,1),(1,0)\}$.

Cores

If $\mathbf{H}$ and $\mathbf{H}_0$ admit homomorphisms to one another, then

$$CSP(\mathbf{H}_0) = CSP(\mathbf{H}).$$

Hence we may always assume $\mathbf{H}$ is a **core**,
i.e.
$\mathbf{H}$ has no proper retracts,
i.e.
every homomorphism from $\mathbf{H}$ to $\mathbf{H}$ is onto,
i.e.
of all structures equivalent to $\mathbf{H}$, $\mathbf{H}$ has smallest universe.

Motivation

- It is well-known that $CSP(\mathbf{H})$ is:
  - poly-time solvable if $\mathbf{H}$ is the complete graph on 2 vertices;
  - NP-complete, if $\mathbf{H}$ is the complete graph on 3 vertices (or more);
  - various other complexities for other targets $\mathbf{H}$;
- The Main Question: Given any finite structure $\mathbf{H}$, can we determine what the complexity of $CSP(\mathbf{H})$ is ?

Outline of this section:

- We describe 5 important complexity classes
- for each class we describe a problem that somehow captures its essence (*complete* problems);
- we give a CSP form of each problem:
  these will be used as running examples.

Reductions, hardness, completeness

Reductions

All reductions are *first-order reductions*
(unless otherwise specified)

- A problem P is *hard* for the complexity class $\mathcal{C}$ if every problem in $\mathcal{C}$ reduces to P;
- the problem P is *$\mathcal{C}$-complete* if it is hard for $\mathcal{C}$ and belongs to the class $\mathcal{C}$.

The class $\mathcal{NP}$

- $\mathcal{NP}$ is the class of problems recognised by a polynomial time bounded non-deterministic Turing machine
- equivalently: $\mathcal{NP}$ is the class of polynomially verifiable problems
- for any structure $\mathbf{H}$ the problem $CSP(\mathbf{H})$ is in $\mathcal{NP}$: given a solution to $CSP(\mathbf{H})$, one may verify it in polynomial time.

A complete problem for $\mathcal{NP}$

(positive) NOT ALL EQUAL 3-SAT

- Input: Sets $S_1, \dots, S_m$ with at most three elements;
- Question: can one colour the elements so that no set gets only one colour ?

A complete problem for $\mathcal{NP}$, CSP form

CSP form of positive NOT ALL EQUAL 3-SAT

CSP(**H**), where **H** is the structure $\mathbf{H} = \langle \{0,1\}; \theta \rangle$ where

$$\theta = \{0,1\}^3 \setminus \{(0,0,0),(1,1,1)\}.$$

The class $\mathcal{P}$

- $\mathcal{P}$ is the class of problems recognised by a polynomial time bounded (deterministic) Turing machine

A complete problem for $\mathcal{P}$

HORN-3-SAT

- Input: A conjunction of Horn 3-clauses
- Question: is there a satisfying assignment ?

A complete problem for $\mathcal{P}$, CSP form

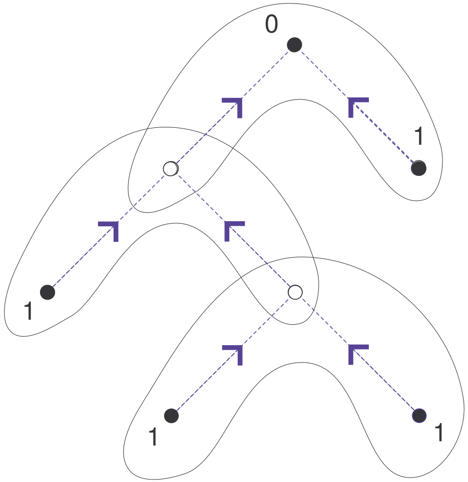
CSP form of HORN-3-SAT

$CSP(\mathbf{H})$, where $\mathbf{H}$ is the structure $\mathbf{H} = \langle\{0,1\};\{0\},\{1\},\rho\rangle$ where

$$\begin{aligned}\rho &= \{(x,y,z) : (y \wedge z) \rightarrow x\}\\ &= \{0,1\}^3 \setminus \{(0,1,1)\}\end{aligned}$$

A complete problem for $\mathcal{P}$, cont'd

An unsatisfiable instance:



The class $\mathcal{NL}$

- $\mathcal{NL}$ is the class of problems recognised by a logarithmic space bounded non-deterministic Turing machine

A complete problem for $\mathcal{NL}$

Directed Reachability

- Input: a directed graph and two specified nodes s and t;
- Question: is there a directed path from s to t ?

A complete problem for $\mathcal{NL}$, CSP form

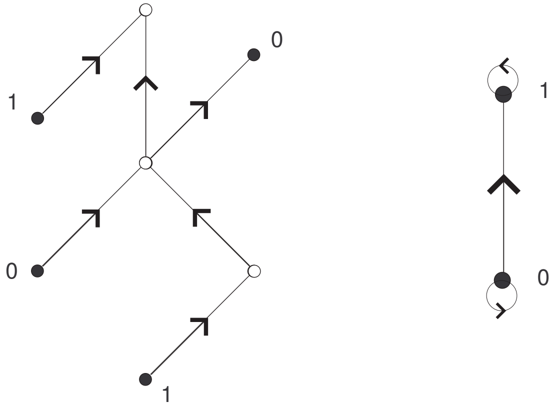
CSP form of *Directed Reachability*

$CSP(\mathbf{H})$, where $\mathbf{H}$ is the structure $\mathbf{H} = \langle\{0,1\};\{0\},\{1\},\leq\rangle$

- Note: this is actually *Un*reachability, but $\mathcal{NL}$ is closed under complementation (Immerman 1988; Szelepcsényi 1987)

A complete problem for $\mathcal{NL}$, cont'd

An unsatisfiable instance (and target): there exists a directed path from a node coloured 1 to a node coloured 0.



The class $\mathcal{L}$

- $\mathcal{L}$ is the class of problems recognised by a logarithmic space bounded (deterministic) Turing machine

A complete problem for $\mathcal{L}$

Undirected Reachability

- Input: an undirected graph and specified nodes s and t;
- Question: is there a path from s to t ?

- The fact that this problem is in $\mathcal{L}$ follows from a deep result of Reingold (2005)

A complete problem for $\mathcal{L}$, CSP form

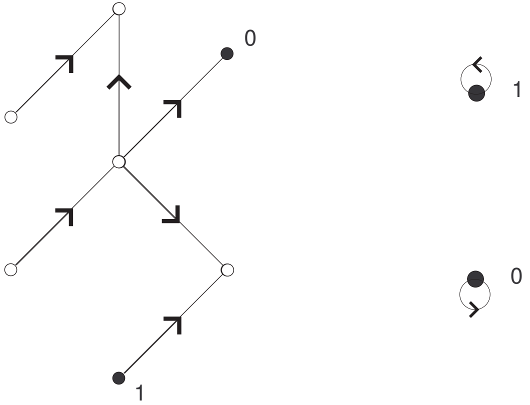
CSP form of *Undirected Reachability*

$CSP(\mathbf{H})$, where $\mathbf{H}$ is the structure $\mathbf{H} = \langle \{0,1\}; \{0\}, \{1\}, = \rangle$

- Note: the CSP actually encodes *Un*reachability.

A complete problem for $\mathcal{L}$, cont'd

An unsatisfiable instance (and target): there exists an undirected path from a node coloured 1 to a node coloured 0.



The classes $mod_p\mathcal{L}$

Let $p \geq 2$ be a prime.

- A language L is in $mod_p\mathcal{L}$ if there exists a logarithmic space-bounded non-deterministic Turing machine M such that $w \in L$ precisely if the number of accepting paths on input w is $0 \mod p$.
- If $p = 2$, $mod_2\mathcal{L}$ is denoted $\oplus\mathcal{L}$ and is called *parity* $\mathcal{L}$.

A complete problem for $mod_p\mathcal{L}$

Linear equations mod p

- Input: a system of linear equations mod p;
- Question: is there a solution ?

Some complete problems for $mod_p\mathcal{L}$, CSP form

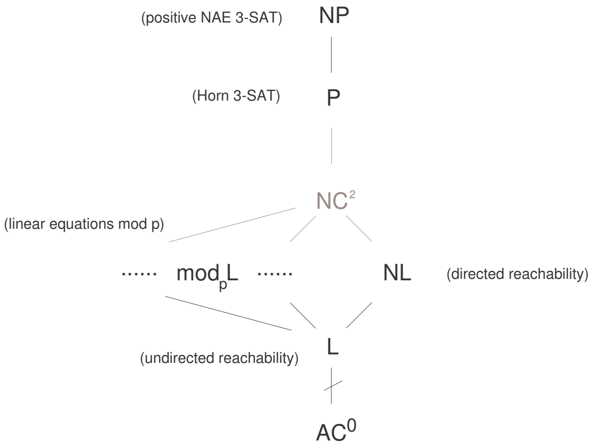
- let $\mathbb{A} = \langle A; +, 0\rangle$ be a finite Abelian group and let b be any non-zero element of $\mathbb{A}$ such that $pb = 0$ for some prime p;
- the following problem is $mod_p\mathcal{L}$-complete

$mod_p\mathcal{L}$-complete CSP form

$CSP(\mathbf{H})$, where $\mathbf{H}$ is the structure $\langle A; \mu, \{b\}, \{0\}\rangle$ with

$$\mu = \{(x, y, z) : x + y = z\}.$$

Containments of these complexity classes



Motivation

Descriptive Complexity

- given a set of structures $\mathcal{S}$, is there a sentence in some "nice" logic that describes precisely the members of $\mathcal{S}$?
- The nicer the logic, the easier it is to recognise the structures.

Motivation, continued

- For instance: if **H** is the 2-element directed edge $0 \longrightarrow 1$, then

$$\neg CSP(\mathbf{H}) = \{\mathbf{G} : \exists a, b, c\, (a \longrightarrow b) \wedge (b \longrightarrow c)\};$$

- It follows that $CSP(\mathbf{H})$ is describable in first-order logic, and hence has very low complexity (FO-definable, in AC^0)

Why Datalog ?

- Datalog is a well-studied database query language;
- it turns out that a large number of natural CSP's are describable via Datalog (viewed as a "nice" logic)
- this property provides simple poly-time algorithms for the CSP;
- in fact: we get a uniform *template* of algorithms (which allows proofs of tractability)

Outline of this section:

- We define the notion of *Datalog Program*, a means of describing certain sets of structures;
- we define 2 *fragments* of Datalog, i.e. special restricted versions;
- we describe, for each fragment,
  a CSP which is definable in it;
- each of these CSP's somehow captures the essence of each fragment
- We provide upper bounds on the complexity of CSP's describable in Datalog and fragments.

Datalog

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

$$I(x, y) \leftarrow J(w, u, x), K(x), \theta_1(x, y, z), \theta_2(x, w)$$

- the relations θ_1 and θ_2 are basic relations of the input structures (EDB's);
- the relations I, J, K are auxiliary relations used by the program (IDB's);
- 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.

An example

Recall:

HORN-3-SAT

$CSP(\mathbf{H})$ where $\mathbf{H} = \langle \{0,1\}; \{0\}, \{1\}, \rho \rangle$ with

$$\rho = \{(x,y,z) : (y \wedge z) \to x\}$$

- Here is a *Datalog program* that accepts precisely those structures that are NOT in $CSP(\mathbf{H})$, i.e. that do not admit a homomorphism to $\mathbf{H}$:

A Datalog program for HORN-3-SAT

A Datalog program

$$
\begin{aligned}
W(x) &\leftarrow 1(x) \\
W(x) &\leftarrow W(y), W(z), \rho(x, y, z) \\
G &\leftarrow W(x), 0(x)
\end{aligned}
$$

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

Definition (Definability in Datalog)

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

Theorem

If $\neg CSP(\mathbf{H})$ is definable in Datalog then $CSP(\mathbf{H})$ is in $\mathcal{P}$.

- Idea: IDB's have bounded arity, so the program can do only polynomially many steps before stabilising

A first fragment: Linear Datalog

Definition (Linear Datalog)

A Datalog program is said to be *linear* if each rule contains at most one occurrence of an IDB in the body.

In other words, each rule looks like this

$$I(x,y) \leftarrow J(w,u,x), \theta_1(x,y,z), \theta_2(x,w)$$

where I and J are the only IDB's, or like this

$$I(x,y) \leftarrow \theta_1(x,y,z), \theta_2(x,w).$$

A non-linear Datalog program

Our program for HORN-3-SAT is *not* linear, since the IDB W occurs twice in the body of the second rule:

A non-linear program

$$\begin{array}{rcl} W(x) & \leftarrow & 1(x) \\ W(x) & \leftarrow & W(y), W(z), \rho(x, y, z) \\ G & \leftarrow & W(x), 0(x) \end{array}$$

A linear Datalog program for *Directed Reachability*

A linear Datalog program

$$
\begin{aligned}
W(x) &\leftarrow 1(x) \\
W(y) &\leftarrow W(x), \theta_{\leq}(x, y) \\
G &\leftarrow W(x), 0(x)
\end{aligned}
$$

Expressibility in Linear Datalog

Theorem

If $\neg CSP(\mathbf{H})$ is definable in Linear Datalog then $CSP(\mathbf{H})$ is in $\mathcal{NL}$.

- Idea: the program rejects if and only if there is a *derivation path* that ends in the goal predicate: this amounts to directed reachability

Another fragment: Symmetric Datalog

Definition (Symmetric Datalog)

A Datalog program is said to be *symmetric* if (i) it is linear and (ii) it is invariant under symmetry of rules.

In other words, if the program contains the rule

$$I(x, y) \leftarrow J(w, u, x), \theta_1(x, y, z), \theta_2(x, w)$$

then it must also contain its *symmetric*:

$$J(w, u, x) \leftarrow I(x, y), \theta_1(x, y, z), \theta_2(x, w).$$

A non-symmetric (linear) Datalog program

Our program for *Directed Reachability* is *not* symmetric:

A non-symmetric linear program

$$\begin{aligned} W(x) &\leftarrow 1(x) \\ W(y) &\leftarrow W(x), \theta_{\leq}(x, y) \\ G &\leftarrow W(x), 0(x) \end{aligned}$$

A symmetric Datalog program for *Undirected Reachability*

A symmetric Datalog program

$$\begin{array}{rcl} W(x) & \leftarrow & 1(x) \\ W(y) & \leftarrow & W(x), \theta_=(x,y) \\ W(x) & \leftarrow & W(y), \theta_=(x,y) \\ G & \leftarrow & W(x), 0(x) \end{array}$$

Expressibility in Symmetric Datalog

Theorem (Egri, BL, Tesson, 2007)

If $\neg CSP(\mathbf{H})$ is definable in Symmetric Datalog then $CSP(\mathbf{H})$ is in $\mathcal{L}$.

- Idea: The program rejects if and only if there is a derivation path that ends in the goal predicate: since the rules are symmetric this amounts to undirected reachability

Non-expressibility Results

The problems we described above which are complete for $mod_p\mathcal{L}$, $\mathcal{P}$ and $\mathcal{NL}$ also have "extremal" properties with respect to expressibility in fragments of Datalog:

Non-expressibility Results, cont'd

Theorem (Feder, Vardi, 1993)

Let $\mu = \{(x, y, z) : x + y = z\}$, *let* $b \neq 0$, *and let* $\mathbf{H} = \langle A; \mu, \{b\}\rangle$. *Then* $\neg CSP(\mathbf{H})$ *is not expressible in Datalog.*

Theorem (Cook, Sethi, 1976)

HORN-3-SAT is not expressible in Linear Datalog.

Theorem (Egri, BL, Tesson, 2007)

Directed Reachability *is not expressible in Symmetric Datalog.*

Recap of Talk 1

$CSP(\mathbf{H})$	*complete*	*expressible in*	*NOT expressible in*
NAE SAT	$\mathcal{N}P$	-	Datalog
linear equations	$mod_p\mathcal{L}$	??	Datalog
Horn SAT	$\mathcal{P}$	Datalog	Lin. Datalog
Directed Reach.	$\mathcal{N}L$	Lin. Datalog	Symm. Datalog
Undir. Reach.	$\mathcal{L}$	Symm. Datalog	FO

Outline of Talk 2

- Talk 2, Part a:
  - to every structure $\mathbf{H}$ we associate an idempotent algebra $\mathbb{A}(\mathbf{H})$;
  - the identities satisfied by $\mathbb{A}(\mathbf{H})$ impose lower bounds on the complexity of $CSP(\mathbf{H})$;
  - we conjecture that the typeset of the variety generated by $\mathbb{A}(\mathbf{H})$ determines the complexity of $CSP(\mathbf{H})$.
- Talk 2, Part b:
  - We present some general results that support these conjectures.