Linear Datalog and n-permutability implies symmetric Datalog

Alexandr Kazda

Department of Mathematics
Vanderbilt University
Nashville

July 21 2015

Datalog

- A way to formalize consistency checking.
- Datalog program consists of predicates (relations on A) and rules for obtaining new statements about possible values of variables from X.
- The program P run on the instance I will start with the set of constraints as its "axioms" and will derive new statements using rules.
- If P derives the goal predicate, the instance is unsatisfiable. Otherwise, the program stops when it can not derive any more statements.
- We say that P decides CSP($\mathbb{A}$) if P derives the goal on all unsatisfiable instances of CSP($\mathbb{A}$) (and nowhere else).

Example: Deciding 2-colorability by Datalog

- P will have two predicates: E (from $\mathbb{A}$) and S.
- S corresponds to the relation $\{(0,0),(1,1)\}$ on A.
- Rules of P:

$$
\begin{aligned}
S(x,x) &\Leftarrow \\
S(x,y) &\Leftarrow S(x,t) \wedge E(t,u) \wedge E(u,y) \\
S(x,y) &\Leftarrow S(x,t) \wedge E(u,t) \wedge E(u,y) \\
S(x,y) &\Leftarrow S(x,t) \wedge E(t,u) \wedge E(y,u) \\
S(x,y) &\Leftarrow S(x,t) \wedge E(u,t) \wedge E(y,u) \\
G &\Leftarrow S(x,y) \wedge E(x,y) \\
G &\Leftarrow S(x,y) \wedge E(y,x)
\end{aligned}
$$

- P reaches the goal predicate if and only if the instance contains a cycle of odd length.

Identifying a 7-cycle

- Instance of 2-colorability has the constraints

$$E(1,\ 2),\ E(2,\ 3),\ E(3,\ 4),\ E(4,\ 5),\ E(5,\ 6),\ E(6,\ 7),\ E(7,\ 1).$$

- A proof of G by $P(I)$:

$$\begin{aligned}
S(1,1) &\Leftarrow \\
S(1,3) &\Leftarrow S(1,1) \wedge E(1,2) \wedge E(2,3) \\
S(1,5) &\Leftarrow S(1,3) \wedge E(3,4) \wedge E(4,5) \\
S(1,7) &\Leftarrow S(1,5) \wedge E(5,6) \wedge E(6,7) \\
G &\Leftarrow S(1,7) \wedge E(7,1)
\end{aligned}$$

Linear and Symmetric Datalog

- Predicates on left hand side of some rule: IDBs
- Linear Datalog: At most one IDB on the right hand side of any rule.
- Symmetric Datalog: At most one IDB on the right hand side of any rule. If there is an IDB on both sides of a rule, we can switch the IDBs.
- Example: Rule

$$S(x, y) \Leftarrow S(x, t) \wedge E(t, u) \wedge E(u, y)$$

  gives

$$S(x, t) \Leftarrow S(x, y) \wedge E(t, u) \wedge E(u, y).$$

- Our 2-colorability program is in symmetric Datalog!

Problems decidable by Datalog

Theorem (L. Barto, M. Kozik)

Let $\mathbb{A}$ be a finite relational structure that contains the relation $\{(a)\}$ for each $a \in A$. TFAE:

- $\mathrm{CSP}(\mathbb{A})$ *is decidable by a Datalog program,*
- *it is not possible to rewrite systems of linear equations over some finite field as* $\mathrm{CSP}(\mathbb{A})$ *instances.*
- *The algebra of polymorphisms of $\mathbb{A}$ generates a meet semidistribtive variety.*

How about linear and symmetric Datalog?

n-permutability

- A relational structure $\mathbb{A}$ is n-permutable if there exist operations $p_0(x, y, z), \ldots, p_n(x, y, z) : A^3 \to A$ that preserve all the relations of $\mathbb{A}$ and satisfy for all $x, y, z \in A$ the set of equalities:

$$\begin{aligned} x &= p_0(x, y, z) \\ p_i(x, x, y) &= p_{i+1}(x, y, y) \\ p_n(x, y, z) &= z. \end{aligned}$$

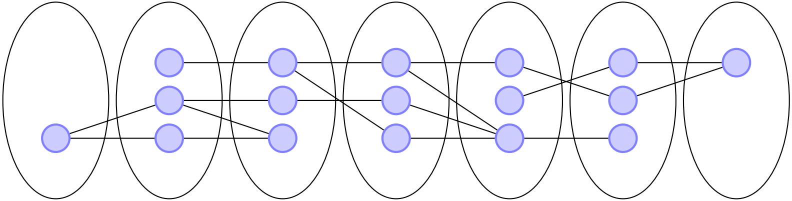
- In order for symmetric Datalog to decide CSP($\mathbb{A}$), there must exist $n \in \mathbb{N}$ such that $\mathbb{A}$ is n-permutable (B. Larose, P. Tesson, L. Egri).

Solving path instances is enough

Theorem

If there is a linear *Datalog program that decides* CSP($\mathbb{A}$), *and* $\mathbb{A}$ *is n-permutable for some n, then there is a* symmetric *Datalog program that decides* CSP($\mathbb{A}$).

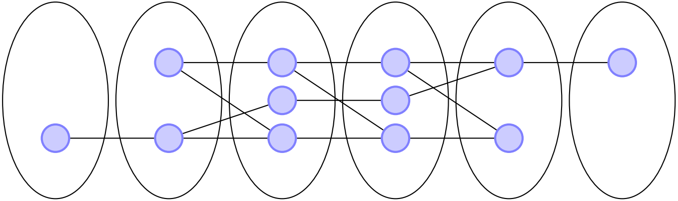
- $\mathbb{A}$ has bounded pathwidth duality, ...
- ... therefore we only need to decide path CSP instances.



- How to use symmetric Datalog to decide path CSP instances?

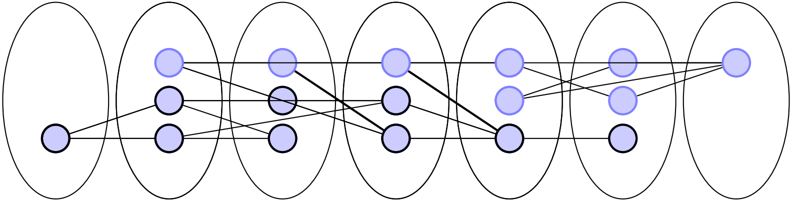
Using n-permutability

- If $\mathbb{A}$ is 4-permutable, then the following can't be an unsatisfiable instance of CSP($\mathbb{A}$):



- Applying Hagemann-Mitschke terms gives us a solution.

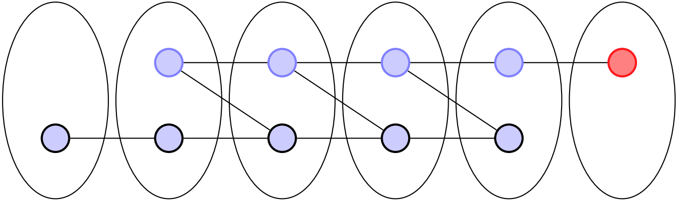
Glued potatoes



- We label black the vertices that can be reached from the starting potato.
- We want lots of backwards edges: Edges from blue to black vertices.
- We use some trickery to glue together potatoes without blue to black edges.
- Nature of the trickery: We can run a smaller symmetric Datalog program inside our original program.

Subdirect subinstance

- Additional trickery gives us a long part of the path instance where everything is subdirect.
- Then it is a matter of pigeonhole principle to find something like this:



Characterizing symmetric Datalog

Conjecture

CSP($\mathbb{A}$) *is decidable by a linear Datalog program if and only if the algebra of polymorphisms of* $\mathbb{A}$ *generates a semidistributive variety.*

Conjecture

CSP($\mathbb{A}$) *is decidable by a symmetric Datalog program if and only if the algebra of polymorphisms of* $\mathbb{A}$ *generates a semidistributive and n-permutable variety for some* $n \in \mathbb{N}$.

Theorem (AK)

Let $\mathbb{A}$ *be n-permutable and let there exist a linear Datalog program that decides* CSP($\mathbb{A}$)*. Then there exists a symmetric Datalog program that decides* CSP($\mathbb{A}$).

Our result plus the first conjecture would give the second conjecture.

Thank you for your attention.