Datalog and Answer Set Programming (part 3)

Riccardo Rosati

Knowledge Representation and Semantic Technologies
Maser of Science in Engineering in Computer Science
Sapienza Università di Roma
a.a. 2020/2021

http://www.diag.uniroma1.it/rosati/krst/

Positive Datalog with constraints

Constraints

A **constraint** is a new kind of rule of the form:

$:- \beta_1, \ldots, \beta_n$

where:

- n is a positive integer
- every β_i is an atom

That is, a constraint is a rule with an empty head.

Examples of constraints:

$:- p(X,Y), r(Y,Z).$
$:- p(a,b), q(b).$
$:- s(Y,X).$

A constraint is **ground** if it does not contain occurrences of variables.

Datalog programs with constraints

A **Datalog program with constraints** is a set of positive rules and constraints.

Semantics of constraints

A ground constraint :– $\beta_1, \ldots, \beta_n$ is **satisfied** in an interpretation I if at least one of its atoms β_i does **not** belong to I.

We naturally extend the notion of grounding of a Datalog program P to the presence of non-ground constraints in P:

The **grounding** of a **constraint** c with respect to a set of constants C, denoted as $ground(c, C)$, is the set of all ground constraints that can be obtained from c by replacing, for every variable x occurring in c, every occurrence of x with a constant from C.

The **grounding** of a (non-ground) positive Datalog **program** with constraints P, denoted as $ground(P)$, is the ground Datalog program obtained by the union of all the sets $ground(r, HU(P))$ such that r is a rule in P, and all the sets $ground(c, HU(P))$ such that c is a constraint in P.

Semantics of programs with constraints

An interpretation I is a **model** for a Datalog program with constraints P if every ground rule and every ground constraint in $ground(P)$ is satisfied in I.

(analogous definition of minimal model as in the case of positive Datalog)

Property: every positive Datalog program with constraints P has either no models (and hence no minimal models) or exactly one minimal model (and in the latter case we denote such a model by MM(P)).

Reasoning over programs with constraints

The techniques for reasoning with positive Datalog programs (naive or semi-naive evaluation) can be easily extended to the presence of contraints in the program.

Let P be a Datalog program with constraints. Then:

- Let P' be the program obtained from P eliminating all the constraints;
- Compute (using e.g. the naive or semi-naive evaluation) MM(P'), the minimal model of P';
- Check whether every constraint in P is satisfied in MM(P'):
  - if this is the case, then MM(P') is the minimal model of P;
  - otherwise P has no models (and hence no minimal models).

(Of course, every constraint in P is satisfied in MM(P') iff every ground constraint in $ground(P)$ is satisfied in MM(P'))

Example

Let P be the following program with constraints:

$$r(X,Y) :- p(X,Y).$$
$$r(X,Z) :- p(X,Y), r(Y,Z).$$
$$s(X,Y) :- r(Y,X).$$
$$:- p(X,X). \text{ [c1]}$$
$$:- r(X,Y), r(Y,X). \text{ [c2]}$$
$$p(a,b).$$
$$p(b,c).$$
$$p(c,d).$$

First, we notice that the program P' obtained from P eliminating the two constraints c1 and c2 corresponds to the previous program for which we have computed the minimal model through the semi-naive computation.

Example

So, the program P' has the following minimal model MM(P'):

$$\{\, p(a,b), p(b,c), p(c,d), r(a,b), r(b,c), r(c,d), r(a,c), r(b,d),$$
$$s(b,a), s(c,b), s(d,c), r(a,d), s(c,a), s(d,b), s(d,a) \,\}$$

Now we have to check whether the constraints c1 and c2 are satisfied in MM(P'). We have:

$ground(c1, HU(P))$ =

$$:\!-\; p(a,a).$$
$$:\!-\; p(b,b).$$
$$:\!-\; p(c,c).$$
$$:\!-\; p(d,d).$$

Example

$ground(c2, HU(P)) =$

$$
\begin{aligned}
&:- r(a,a), r(a,a).\\
&:- r(a,b), r(b,a).\\
&:- r(a,c), r(c,a).\\
&:- r(a,d), r(d,a).\\
&:- r(b,a), r(a,b).\\
&:- r(b,b), r(b,b).\\
&:- r(b,c), r(c,b).\\
&:- r(b,d), r(d,b).\\
&:- r(c,a), r(a,c).\\
&:- r(c,b), r(b,c).\\
&:- r(c,c), r(c,c).\\
&:- r(c,d), r(d,c).\\
&:- r(d,a), r(a,d).\\
&:- r(d,b), r(b,d).\\
&:- r(d,c), r(c,d).\\
&:- r(d,d), r(d,d).
\end{aligned}
$$

Example

It is easy to verify that every ground constraint in $ground(c1, HU(P))$ is satisfied in MM(P'), and that every ground constraint in $ground(c2, HU(P))$ is satisfied in MM(P').

Consequently, MM(P') is the minimal model of P.

Example

Now let P’’ be the program obtained from P adding the following constraint:

$$:- r(X,Y), r(Y,W), r(W,Z), s(Z,X). \quad \text{[c3]}$$

Among others, the following ground constraint belongs to $ground\big(c3, HU(P)\big)$:

$$:- r(a,b), r(b,c), r(c,d), s(d,a).$$

This ground constraint in NOT satisfied in MM(P’) (because all the atoms in the body of the constraints belong to MM(P’)), therefore c3 is non satisfied in MM(P’). Consequently, P’’ has no models (and hence no minimal models).