

Partial Stable Generated Models of Generalized Logic Programs with Constraints

Sibylle Schwarz
Universität Leipzig, Germany
email: schwarz@informatik.uni-leipzig.de
Graduiertenkolleg Wissensrepräsentation
Thesis advisor : Heinrich Herre

Motivation

Logic programs are an important knowledge representation tool. In many cases, definite logic programs are too restricted to formalize problems intuitively. Therefore definite logic programs have been extended in several ways by additional syntactic constructs such as negation and disjunction.

A very expressive program class are *generalized logic programs*. Their rules may contain any quantifier free formula in both their body and head. Definite, normal and disjunctive logic programs are special cases of generalized logic programs.

Constraint logic programming, defined in [4], extends definite logic programs by constraints: logical expressions that describe special properties of the problem domain. The combination of these two extensions defines the syntax of *generalized logic programs with constraints* (GLP_C).

Declarative semantics provides a mathematically precise definition of the meaning of a logic program.

While least Herbrand models generalize naturally to least constraint models of definite constraint logic programs, there is no general agreement about the declarative semantics of programs containing negation and disjunction.

The set of all *partial stable generated models* provides a declarative semantics of GLP_C -programs. They are *three-valued versions of stable generated models*, defined for generalized programs in [3] and extended to programs with constraints in [5].

Syntax

A **constraint domain** C is a pair (D_C, L_C) where

$L_C(X)$ is the constraint language. It contains true, false and =. Elements of $L_C(X)$ are called *constraints*.

D_C is a structure with non-empty carrier set $|D_C|$ and for every symbol from Σ_C there is a constant, function or relation representing it in D_C .

Rules are pairs of quantifier free first order formulas denoted by $r = (\varphi \Rightarrow \psi)$. $B(r) = \varphi$ is the body and $H(r) = \psi$ the head of the rule r .

C-rules $(c||r)$ are pairs of a constraint and a rule.

Generalized logic programs with constraints are sets of C-rules.

GLP_C denotes the set of all GLP_C -rules.

GLP_C^0 is the set of all GLP_C -rules without variables.

C-interpretations and C-models

C-base B_P^C set of all **generalized atoms** $p(d_1, \dots, d_n)$ with n -ary relation symbol p defined in program P and $d_i \in |D_C|$

partial C-interpretation $I : B_P^C \rightarrow V$, where V is the set of truth values {false, unknown, true} with the ordering false $<_t$ unknown $<_t$ true

$\mathbb{I}_{C,P}$ is the set of all partial C-interpretations of the GLP_C -program P .

I_{false} is the t -least interpretation that maps any generalized atom to false.

Intervals of partial C-interpretations $[I, J]_t = \{K \mid I \leq_t K \leq_t J\}$.

partial C-models of a program are partial C-interpretations M that satisfy $M(B(r)) \leq_t M(H(r))$ for every rule r in the ground instantiation of the program.

Generated C-models

Intended models of logic programs should be

grounded (can be created bottom-up from zero information by iterated application of program rules)

supported (every generalized atom that is true in the model is justified by a program rule whose body is satisfied in this model)

A C-model M of a logic program is *generated* by a relation R

if there is a chain $\{I_\alpha \mid 0 \leq \alpha < \kappa\}$ of C-interpretations with the following properties

- $I_0 = I_{\text{false}}$ and $I_\kappa = M$
- for successor ordinals $\alpha : R(I_{\alpha-1}, I_\alpha)$
- for limit ordinals $\beta : I_\beta = \sup\{I_\alpha \mid 0 \leq \alpha < \beta\}$

The **successor relation** $\text{Succ}_C^{(P,M)} \subseteq (\mathbb{I}_{C,P})^2$

- $$\text{Succ}_C^{(P,M)}(I, J) \text{ iff } \begin{array}{l} 1. J \in [I, M]_t \\ 2. \forall r \in \text{sel}([P]_C) : I(B(r)) \leq_t J(H(r)) \\ 3. \forall K \in [I, J]_t : \text{if } K \text{ satisfies 2. then } K = J \text{ (minimality)} \end{array}$$

always generates **grounded** interpretations.

$\text{Succ}_C^{(P,M)}$ is parameterized by the GLP_C -program P , a guessed C-interpretation M and a function $\text{sel} : 2^{\text{GLP}_C^0} \rightarrow 2^{\text{GLP}_C^0}$ that selects the set of firing rules from the ground instantiation of the program.

Partial stable generated C-models

We choose the rule selection function sel in a way that guarantees persistence of the bodies of all used rules and therefore **supportedness** of the generated model.

A three-valued C-interpretation M of a GLP_C -program P is a *partial stable generated C-model* of a GLP_C -program P if it is generated by, and a fixed point of, the successor relation $\text{Succ}_C^{(P,M)}$ using the rule selection function

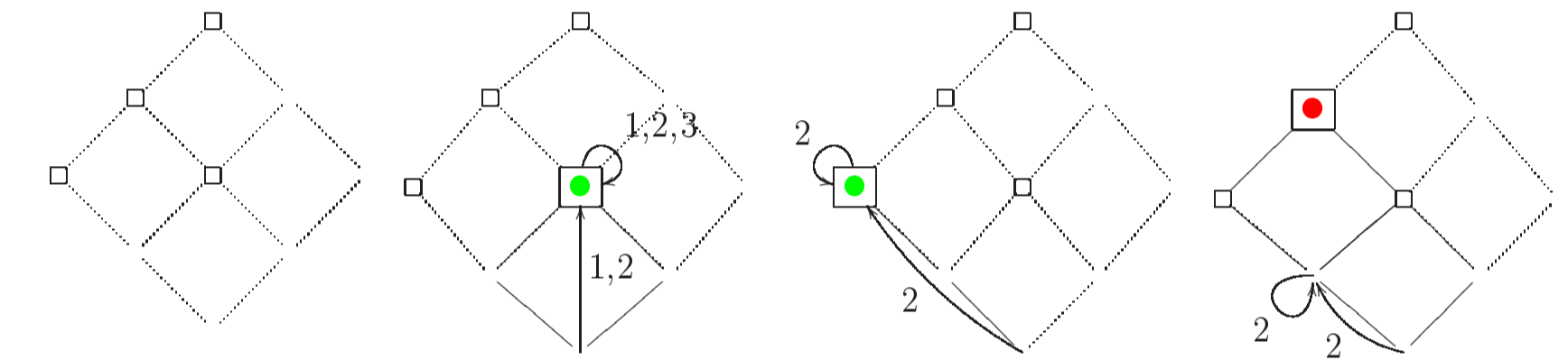
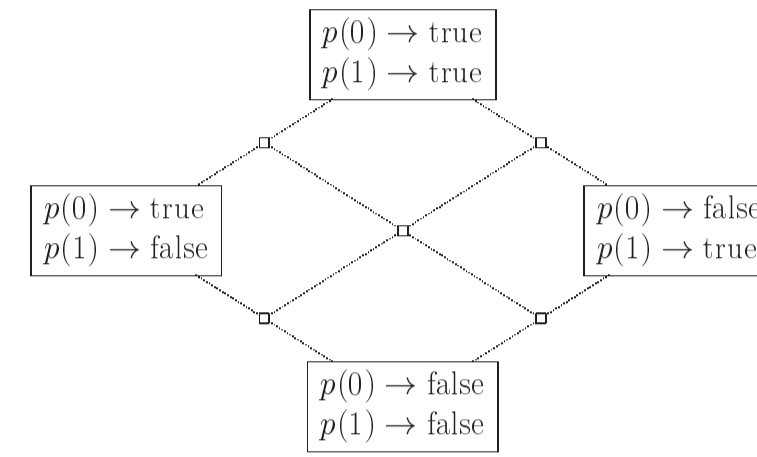
$$\text{sel} : \mathbb{I}_{C,P} \times \mathbb{I}_{C,P} \times 2^{\text{GLP}_C^0} \rightarrow 2^{\text{GLP}_C^0} \quad \text{sel}(I, M, P) = \{r \in [P]_C \mid [I, M]_t(B(r)) >_t \text{false}\}$$

The *partial stable model semantics* of a GLP_C -program is the set of all its partial stable generated C-models.

Example

$$P = \{ x \neq y, y = z \parallel \text{not } p(x) \vee p(y) \Rightarrow p(z) \\ x \neq 0, x \neq y \parallel p(x) \Rightarrow p(y) \}$$

$$[P]_{\{0,1\}} = \{ \text{not } p(0) \vee p(1) \Rightarrow p(1) \quad (1) \\ \text{not } p(1) \vee p(0) \Rightarrow p(0) \quad (2) \\ p(1) \Rightarrow p(0) \quad (3) \}$$



$$\begin{array}{lll} (p(0) \rightarrow \text{unknown}, & (p(0) \rightarrow \text{true}, & (p(0) \rightarrow \text{true}, \\ p(1) \rightarrow \text{unknown}) & p(1) \rightarrow \text{false}) & p(1) \rightarrow \text{unknown}) \end{array}$$

all models of P stable generated stable generated not stable generated

Questions

The most important task is to find an *operational semantics* corresponding to the presented declarative semantics.

Problems related to this question:

- more abstract representation of C-interpretations
- lifting the successor relation $\text{Succ}_C^{(P,M)}$ to these abstract C-interpretations
- proof theoretical characterization of partial stable generated C-models
- comparison to other declarative semantics of restricted classes of constraint logic programs ([2], [1])
- investigation of other interesting rule selection functions sel

References

- [1] J. Dix and F. Stolzenburg. A framework to incorporate non-monotonic reasoning into constraint logic programming. *JLP*, 37(1-3), 1998.
- [2] F. Fages and R. Gori. A hierarchy of semantics for normal constraint logic programs. In *ALP'96, LNCS 1139*, 1996.
- [3] H. Herre and G. Wagner. Stable models are generated by a stable chain. *JLP*, 30(2), 1997.
- [4] J. Jaffar, M. Maher, K. Marriott, and P. Stuckey. The semantics of constraint logic programs. *JLP*, 37, 1998.
- [5] S. Schwarz. Stable generated models of generalized constraint logic programs. In *Proc. WFLP 2001*, 2001.