

# FLP Semantics without Circular Justifications for General Logic Programs

Yi-Dong Shen <sup>1</sup>   Kewen Wang <sup>2</sup>

<sup>1</sup>Institute of Software, Chinese Academy of Sciences, Beijing, China  
<http://lcs.ios.ac.cn/~ydshen>

<sup>2</sup>Griffith University, Brisbane, Australia  
<http://www.ict.griffith.edu.au/~kewen>

AAAI 2012, Toronto, Canada

- Answer set programming (ASP) is a major logic programming paradigm for knowledge representation and reasoning.
- As summarized by [Lif10], there have been thirteen different definitions of answer sets in the literature.
- In this paper, we are devoted to a currently widely used definition of answer sets, called

## FLP semantics [FLP04]

Let  $\Pi$  be a normal logic program with rules of the form

$$A_0 \leftarrow A_1, \dots, A_m, \text{not } A_{m+1}, \dots, \text{not } A_n$$

where each  $A_i$  is an atom. Given an interpretation  $I$ , the FLP reduct of  $\Pi$  w.r.t.  $I$ , denoted  $f\Pi^I$ , consists of all rules in  $\text{ground}(\Pi)$  whose bodies are satisfied by  $I$ .  $I$  is an FLP answer set of  $\Pi$  if  $I$  is a minimal model of  $f\Pi^I$ .

- It was recently observed that when applied to more general logic programs, the FLP semantics may produce unintuitive answer sets with circular justifications caused by self-supporting loops [SY09, LPST10, She11, SW11].

## Circular justifications of the FLP semantics

Consider the following general logic program [BLM11]:

$$\begin{aligned}\Pi_1 : \quad & p(2) \leftarrow p(2) \wedge (\neg p(-1) \vee p(1)). & r_1 \\ & p(-1) \leftarrow \neg p(-1) \vee p(1) \vee p(2). & r_2 \\ & p(1) \leftarrow p(-1). & r_3\end{aligned}$$

Let  $I = \{p(-1), p(1)\}$  be an interpretation. The FLP reduct of  $\Pi_1$  w.r.t.  $I$  is  $f\Pi_1^I = \{r_2, r_3\}$ .  $I$  is a minimal model of  $f\Pi_1^I$  and thus is an answer set of  $\Pi_1$  under the FLP semantics. Observe that this FLP answer set has circular justifications caused by the following self-supporting loop:

$$p(1) \Leftarrow p(-1) \Leftarrow \neg p(-1) \vee p(1) \vee p(2) \Leftarrow p(1)$$

- In this paper, we address the circular justification problem for general logic programs by enhancing the FLP semantics with a level mapping formalism.
- In particular, we extend the Gelfond-Lifschitz three step definition of the standard answer set semantics from normal logic programs to general logic programs and define for general logic programs the first FLP semantics that is free of circular justifications. We call this FLP semantics the **well-justified FLP semantics**.
- This method naturally extends to general logic programs with additional constraints like aggregates, thus providing a unifying framework for defining the well-justified FLP semantics for different types of logic programs.

## Rules

A *rule* is of the form  $H \leftarrow B$ , where  $H$  and  $B$  are first-order formulas. Such a rule  $r$  expresses an *if-then* statement, saying that *if* the logic property  $B$  holds *then* infer  $H$ .

## General logic programs [BLM11]

A *general logic program*  $\Pi$  (also called a logic program with first-order formulas) consists of a finite set of rules.  $\Pi$  is a *normal logic program* if each rule  $r$  is of the form

$$A_0 \leftarrow A_1 \wedge \dots \wedge A_m \wedge \neg A_{m+1} \wedge \dots \wedge \neg A_n$$

where each  $A_i$  is an atom without equality and function symbols.

A *positive logic program* is a normal logic program without negative literals.

## Grounding

Let  $\mathcal{C}_\Pi \subseteq \mathcal{C}$  be a non-empty, finite set of constants including all constants occurring in  $\Pi$ . A *closed instance* of a rule is obtained by replacing every free variable in the rule with a constant in  $\mathcal{C}_\Pi$ . The *grounding* of  $\Pi$  w.r.t.  $\mathcal{C}_\Pi$ , denoted  $ground(\Pi)$ , is the set of closed instances of all rules in  $\Pi$ .

## Satisfaction

An interpretation  $I$  *satisfies* a closed instance  $r$  of a rule if it satisfies  $head(r)$  or it does not satisfy  $body(r)$ .  $I$  is a *model* of a logic program  $\Pi$  if it satisfies all rules in  $ground(\Pi)$ .

## Definition 1

Let  $\Pi$  be a general logic program and  $I$  an interpretation. The *FLP-reduct* of  $\Pi$  w.r.t.  $I$  is

$$f\Pi^I = \{r \in \text{ground}(\Pi) \mid I \text{ satisfies } \text{body}(r)\}.$$

$I$  is an *FLP answer set* of  $\Pi$  if  $I$  is a minimal model of  $f\Pi^I$ .

\*\* This FLP semantics for general logic programs is reformulated by [BLM11] in terms of a modified circumscription.

## Key points:

- Rules  $H \leftarrow B$  in a logic program differ essentially from implications  $B \supset H$  in classical logic because rules define a level mapping on their answer sets s.t. answers at upper levels are derived from answers at lower levels by applying the rules in the way that if the rule bodies hold then infer the heads.
- However, the FLP semantics is unable to capture such a level mapping. For  $I$  to be an answer set of  $\Pi$ , the FLP semantics only requires  $I$  to be a minimal model of  $f\Pi'$ . This amounts to treating all rules  $H \leftarrow B$  in  $f\Pi'$  as implications  $B \supset H$  in classical logic because  $I$  is a model of the rules  $H \leftarrow B$  in  $f\Pi'$  if and only if  $I$  is a model of the corresponding implications  $B \supset H$  in classical logic. Since classical implications define no level mapping on their models, some minimal models of  $f\Pi'$  may have no level mapping and thus some FLP answer sets may have circular justifications.



- Therefore, the key to overcome the circular justification problem is to enhance the FLP semantics with a level mapping formalism whereby the FLP reduct is treated as rules instead of classical implications.
- To achieve this, let us first look at how the standard answer set semantics builds a level mapping for its answer sets from normal logic programs.
- \*\* For an interpretation  $I$ , We denote  $I^-$  for  $\mathcal{H}_\Sigma \setminus I$ , and  $\neg I^-$  for  $\{\neg A \mid A \in I^-\}$ , where  $\mathcal{H}_\Sigma$  is the set of ground atoms of the signature  $\Sigma$ .

# Well-Justified FLP Answer Sets

Let  $\Pi$  be a normal logic program and  $I$  an interpretation.

Three step definition of the standard answer set semantics [GL88]

① Eliminate from  $ground(\Pi)$  all rules whose bodies contain a negative literal not satisfied by  $I$ .

② Eliminate all negative literals from the remaining rules.

\*\* All negative literals removed in the second step are in  $\neg I^-$ . The remaining part of  $ground(\Pi)$  after the two steps of transformation is called the *Gelfond-Lifschitz reduct*, denoted  $\Pi'$ .

③ Compute a fixpoint  $T_{\Pi'}^\alpha(\emptyset)$  from  $\Pi'$  via  $\langle T_{\Pi'}^i(\emptyset) \rangle_{i=0}^\infty$ , where  $T_{\Pi'}^0(\emptyset) = \emptyset$  and for  $i \geq 0$   $T_{\Pi'}^{i+1}(\emptyset) = T_{\Pi'}(T_{\Pi'}^i(\emptyset))$ .

\*\*  $T_P(S)$ , where  $P$  is a positive logic program and  $S$  a set of ground atoms, is the van Emden-Kowalski provability operator [vEK76] defined by  $T_P(S) = \{head(r) \mid r \in ground(P) \text{ such that } body(r) \text{ is satisfied by } S\}$ .

Then,  $I$  is an answer set of  $\Pi$  if  $I = T_{\Pi'}^\alpha(\emptyset)$ .

- Note that the derivation sequence  $\langle T_{\Pi'}^i(\emptyset) \rangle_{i=0}^{\infty}$  defines a level mapping on  $I$  such that  $A \in I$  is at level  $i > 0$  if  $A \in T_{\Pi'}^i(\emptyset)$  but  $A \notin T_{\Pi'}^{i-1}(\emptyset)$ .
- As a result, for any  $A \in I$  at level  $i$  there is a rule  $r \in \text{ground}(\Pi)$  whose head is  $A$  such that all negative literals in  $\text{neg}(r)$  are in  $\neg I^-$  and all positive literals in  $\text{pos}(r)$  are in  $T_{\Pi'}^{i-1}(\emptyset)$ .

- The above Gelfond-Lifschitz three step definition of answer sets for normal logic programs is not applicable to general logic programs, since rule heads and bodies of a general logic program are arbitrary first-order formulas.
- To deal with arbitrary first-order formulas in rule heads and bodies in a general logic program, we propose to extend the above Gelfond-Lifschitz three step definition of answer sets to general logic programs.

Let  $\Pi$  be a general logic program and  $I$  an interpretation.

## Extension of the three step definition of answer sets

- 1 We extend the first step to first-order formulas by eliminating from  $ground(\Pi)$  all rules whose bodies are not satisfied by  $I$ . This yields the FLP reduct  $f\Pi^I$ .
- 2 Instead of directly eliminating from  $f\Pi^I$  all negative literals in  $\neg I^-$ , we adapt the second step to first-order formulas by adding the negative literals in  $\neg I^-$  as constraints on  $f\Pi^I$ .
- 3 We extend the third step to first-order formulas by computing a fixpoint  $O^\alpha$  from  $f\Pi^I$  under the constraints  $\neg I^-$  via a sequence  $\langle O^i \rangle_{i=0}^\infty$ , where  $O^0 = \emptyset$  and for  $i > 0$  and any rule  $r$  in  $f\Pi^I$ , if  $body(r)$  is true in  $O^{i-1}$  under the constraints  $\neg I^-$ , i.e.  $O^{i-1} \cup \neg I^- \models body(r)$ , then  $head(r)$  is in  $O^i$ , where  $\models$  is the entailment relation.

Define  $\langle O^i \rangle_{i=0}^\infty$  using a one-step provability operator:

- We extend the van Emden-Kowalski operator  $T_P(S)$ , which is applicable only to a positive logic program  $P$  parameterized with a set  $S$  of ground atoms, to a new operator  $T_\Pi(O, N)$ , which is applicable to a general logic program  $\Pi$  parameterized with two first-order theories  $O$  and  $N$ .
- Intuitively, by applying  $T_\Pi(O, N)$  we infer all rule heads from  $ground(\Pi)$  whose rule bodies are true in  $O$  under the constraints  $N$ .

- Formally we define

$$T_\Pi(O, N) = \{head(r) \mid r \in ground(\Pi) \text{ s.t. } O \cup N \models body(r)\}.$$

- Therefore, we can apply the operator  $T_\Pi(O, N)$  to define  $\langle O^i \rangle_{i=0}^\infty$  by letting  $O^i = T_{f\Pi'}^i(\emptyset, \neg I^-)$  and  $O^\alpha = T_{f\Pi'}^\alpha(\emptyset, \neg I^-)$ .

# Well-Justified FLP Answer Sets

Properties of  $T_{f\Pi'}^i(\emptyset, \neg I^-)$ :

## Theorem

*Let  $I$  be a model of a general logic program  $\Pi$ . For any  $i \geq 0$ ,  $T_{\Pi}^i(\emptyset, \neg I^-) = T_{f\Pi'}^i(\emptyset, \neg I^-)$  and thus  $T_{\Pi}^{\alpha}(\emptyset, \neg I^-) = T_{f\Pi'}^{\alpha}(\emptyset, \neg I^-)$ .*

## Theorem

*Let  $I$  be a model of a normal logic program  $\Pi$ . For any  $i \geq 0$ ,  $T_{\Pi'}^i(\emptyset) = T_{\Pi}^i(\emptyset, \neg I^-)$  and thus  $T_{\Pi'}^{\alpha}(\emptyset) = T_{\Pi}^{\alpha}(\emptyset, \neg I^-)$ .*

## Corollary

*A model  $I$  of a normal logic program  $\Pi$  is an answer set under the standard answer set semantics if and only if  $I = T_{\Pi'}^{\alpha}(\emptyset)$  if and only if  $I = T_{\Pi}^{\alpha}(\emptyset, \neg I^-)$  if and only if  $I = T_{f\Pi'}^{\alpha}(\emptyset, \neg I^-)$ .*

We use  $T_{\Pi}^{\alpha}(\emptyset, \neg I^{-})$  to define answer sets:

## Definition of the well-justified FLP semantics

Let  $I$  be a model of a general logic program  $\Pi$ .  $I$  is a *well-justified FLP answer set* of  $\Pi$  if for each  $A \in I$ ,  $T_{f\Pi_I}^{\alpha}(\emptyset, \neg I^{-}) \cup \neg I^{-} \models A$ .

## Example

Consider the general logic program  $\Pi_1$  in the introduction.  $I = \{p(-1), p(1)\}$  is a model of  $\Pi_1$  and is also an FLP answer set.  $\neg I^{-}$  contains  $\neg p(2)$ .  $T_{f\Pi_1}^{\alpha}(\emptyset, \neg I^{-}) = \emptyset$ . Neither  $p(-1) \in I$  nor  $p(1) \in I$  can be proved true in  $T_{f\Pi_1}^{\alpha}(\emptyset, \neg I^{-})$  under the constraints  $\neg I^{-}$ , so  $I$  is not a well-justified FLP answer set of  $\Pi_1$ .



Properties of the well-justified FLP semantics:

## Theorem

*If  $I$  is a well-justified FLP answer set of a general logic program  $\Pi$ , then  $I$  is an FLP answer set of  $\Pi$ .*

## Level mapping

Well-justified FLP answer sets have a level mapping, which is built from the FLP reduct  $f\Pi^I$  via the sequence  $\langle T_{f\Pi^I}^i(\emptyset, \neg I^-) \rangle_{i=0}^{\infty}$ , where for each  $A \in I$ ,  $A$  is at level  $i > 0$  if

$T_{f\Pi^I}^i(\emptyset, \neg I^-) \cup \neg I^- \models A$  but  $T_{f\Pi^I}^{i-1}(\emptyset, \neg I^-) \cup \neg I^- \not\models A$ .





The enhancement of the level mapping makes the resulting FLP answer sets free of circular justifications.






The well-justified FLP semantics has been extended to general logic programs with aggregates, and to description logic programs (dl-programs).

- When this method is applied to normal logic programs with aggregates, the well-justified FLP semantics agrees with the conditional satisfaction based semantics defined by [SPT07].
- When applied to dl-programs, the semantics agrees with the strongly well-supported semantics defined by [She11].

- The FLP semantics is first introduced in [FLP04] for normal (and disjunctive) logic programs with aggregates. It is then extended to dl-programs and Hex programs [EIST05, EIL<sup>+</sup>08], modular logic programs [DTEFK09], disjunctive dl-programs [Luk10], and general logic programs with aggregates [BLM11]. [Tru10] and [FLL11] also extend the FLP semantics of [FLP04] to propositional formulas and first-order formulas, respectively. As illustrated in [BLM11], these two extensions do not agree with the FLP semantics of [BLM11] (see Definition 1) for general logic programs.
- We observe that the problem of circular justifications with the FLP semantics persists in all the above extensions.

- For normal logic programs with aggregates, [SY09] observe that answer sets under the FLP semantics of [FLP04] have circular justifications. They propose a default semantics by translating a propositional logic program to a default logic theory and show that the default semantics agrees with the conditional satisfaction based semantics of [SPT07].
- [LPST10] also indicate the circular justification (self-supportedness) problem with the FLP semantics of [FLP04]. They propose a computation based semantics for normal logic programs with aggregates, which proves to coincide with the conditional satisfaction based semantics.
- For dl-programs, [She11] observes the circular justification problem with the FLP semantics defined by [EIST05]. For disjunctive dl-programs, [SW11] notice the circular justification problem with the FLP semantics defined by [Luk10].

-  M. Bartholomew, J. Lee, and Y. Meng.  
First-order extension of the FLP stable model semantics via modified circumscription.  
In *IJCAI*, pages 724–730, 2011.
-  M. Dao-Tran, T. Eiter, M. Fink, and T. Krennwallner.  
Modular nonmonotonic logic programming revisited.  
In *ICLP*, pages 145–159, 2009.
-  T. Eiter, G. Ianni, T. Lukasiewicz, R. Schindlauer, and H. Tompits.  
Combining answer set programming with description logics for the semantic web.  
*Artificial Intelligence*, 172(12-13):1495–1539, 2008.
-  T. Eiter, G. Ianni, R. Schindlauer, and H. Tompits.  
A uniform integration of higher-order reasoning and external evaluations in answer-set programming.  
In *IJCAI*, pages 90–96, 2005.

-  P. Ferraris, J. Lee, and V. Lifschitz.  
Stable models and circumscription.  
*Artificial Intelligence*, 175(1):236–263, 2011.
-  W. Faber, N. Leone, and G. Pfeifer.  
Recursive aggregates in disjunctive logic programs: Semantics and complexity.  
In *Logics in Artificial Intelligence: European Workshop*, pages 200–212, 2004.
-  M. Gelfond and V. Lifschitz.  
The stable model semantics for logic programming.  
In *ICLP*, pages 1070–1080, 1988.
-  V. Lifschitz.  
Thirteen definitions of a stable model.  
In *Fields of Logic and Computation*, pages 488–503, 2010.
-  L. Liu, E. Pontelli, T.C. Son, and M. Truszczynski.

Logic programs with abstract constraint atoms: the role of computations.

*Artificial Intelligence*, 174(3-4):295–315, 2010.



T. Lukasiewicz.

A novel combination of answer set programming with description logics for the semantic web.

*IEEE Transactions on Knowledge and Data Engineering*, 22(11):1577–1592, 2010.



Y. D. Shen.

Well-supported semantics for description logic programs.

In *IJCAI*, pages 1081–1086, 2011.



T. C. Son, E. Pontelli, and P. H. Tu.

Answer sets for logic programs with arbitrary abstract constraint atoms.

*Journal of Artificial Intelligence Research*, 29:353–389, 2007.



Y. D. Shen and K. W. Wang.

Extending logic programs with description logic expressions for the semantic web.

In *International Semantic Web Conference*, pages 633–648, 2011.



Y. D. Shen and J. H. You.

A default approach to semantics of logic programs with constraint atoms.

In *LPNMR*, pages 277–289, 2009.



M. Truszczyński.

Reducts of propositional theories, satisfiability relations, and generalizations of semantics of logic programs.

*Artificial Intelligence*, 174(16-17):1285–1306, 2010.



M. H. van Emden and R. A. Kowalski.

The semantics of predicate logic as a programming language.

*J. ACM*, 23(4):733–742, 1976.