Aggregates in Recursion: Issues and Solutions

Wolfgang Faber

Alpen-Adria-Universität Klagenfurt

RuleML Webinar, 2018-05-25

Outline

Motivation: Aggregates

- Aggregates facilitate problem representation
- Standard in database query languages
  For example:
  SELECT company.name FROM company,employee
  WHERE company.id = employee.cid AND
  COUNT(employee.id) < 10

Motivation: Aggregates and Logic Programming

- In this talk: Combination of Logic Programming and Aggregates
- For example:
  SELECT company.name FROM company,employee
  WHERE company.id = employee.cid AND
  COUNT(employee.id) < 10

  using Logic Programming (here ASP/Datalog):

  $$result(N) \text{ :- } company(N, Y), \#count\{X : employee(X, Y)\} < 10.$$

- Does this look innocent?

Recursion

- A main feature of Logic Programming is recursion
- Example:

$$tc(A, B) \coloneq p(A, B).$$
$$tc(A, B) \coloneq p(A, X), tc(X, B).$$

  defines the transitive closure of p
- Supported also in SQL
  - Recent (since SQL-99)
  - Not particularly well-known
  - Common Table Expressions (CTE)

Recursion and Aggregates

- Combination of recursion and aggregates?
  - Explicitly forbidden in SQL!
- Meaning of

$$p(a) \coloneq \#count\{X : p(X)\} > 0.$$

- Meaning of

$$p(a) \coloneq \#count\{X : p(X)\} < 1.$$

- Meaning of

$$p(1) \coloneq \#avg\{X : p(X)\}! = 1.$$
$$p(-1) \coloneq \#avg\{X : p(X)\}! = 1.$$

ASP Semantics

- Herbrand interpretations
- Reduct for interpretation I:
  1. Delete rules whose negative body is true in I.
  2. Delete negative body from all other rules.
- Interpretations I which are minimal models of the reduct for I are answer sets.

- [Gelfond, Lifschitz 1988] (nondisjunctive)
- [Przymusinski 1991] (disjunctive)
- [Gelfond, Lifschitz 1991] (disjunctive, 2 kinds of negation)

ASP Semantics

- Herbrand interpretations
- Reduct for interpretation I:
  1. Delete rules whose negative body is true in I.
  2. Delete negative body from all other rules.
- Interpretations I which are minimal models of the reduct for I are answer sets.

- [Gelfond, Lifschitz 1988] (nondisjunctive)
- [Przymusinski 1991] (disjunctive)
- [Gelfond, Lifschitz 1991] (disjunctive, 2 kinds of negation)

Aggregates

- Aggregate Functions: Functions over ground term (multi)sets
- (Multi)sets specified as $\{A, B : Conj\}$ or $\{\langle c, d : Conj\rangle, \ldots\}$
- Evaluate *Conj* w.r.t. an interpretation
- Aggregate Atoms: Aggregate Function plus comparison

Important: Aggregate atoms depend on truth values of a set of standard atoms!

Aggregates

- Aggregate Functions: Functions over ground term (multi)sets
- (Multi)sets specified as $\{A, B : Conj\}$ or $\{\langle c, d : Conj\rangle, \ldots\}$
- Evaluate *Conj* w.r.t. an interpretation
- Aggregate Atoms: Aggregate Function plus comparison

Important: Aggregate atoms depend on truth values of a set of standard atoms!

Aggregates

- Aggregate Functions: Functions over ground term (multi)sets
- (Multi)sets specified as $\{A, B : Conj\}$ or $\{\langle c, d : Conj\rangle, \ldots\}$
- Evaluate *Conj* w.r.t. an interpretation
- Aggregate Atoms: Aggregate Function plus comparison

Important: Aggregate atoms depend on truth values of a set of standard atoms!

Aggregates

- Aggregate Functions: Functions over ground term (multi)sets
- (Multi)sets specified as $\{A, B : Conj\}$ or $\{\langle c, d : Conj\rangle, \ldots\}$
- Evaluate *Conj* w.r.t. an interpretation
- Aggregate Atoms: Aggregate Function plus comparison

Important: Aggregate atoms depend on truth values of a set of standard atoms!

Outline

Aggregate Stratification

Definition

A program $\mathcal{P}$ is *stratified on an aggregate atom A* if there exists a level mapping $|| \; ||$ from its predicates to ordinals, such that for each rule and for each of its head atoms *a* the following holds:

1. For each predicate *b* of standard body literals: $||b|| \leq ||a||$,
2. for each predicate *b* inside an aggregate body atom: $||b|| < ||a||$, and
3. for each predicate *b* in the head: $||b|| = ||a||$.

Aggregate Stratification

- Note: Stratification is relative to a program
- Unstratified aggregate atoms occur recursively

Example

$$a \text{:-}\ \#count\{\langle t : b\rangle\} > 0.$$

is stratified on the aggregate atom.

$$\begin{array}{l} a \text{:-}\ \#count\{\langle t : b\rangle\} > 0. \\ b \text{:-}\ a. \end{array}$$

is not stratified on the aggregate atom.

Answer Sets for Aggregate-stratified Programs

- Basic Idea: Treat aggregate atoms like negative literals
- Reduct:
  1. Delete rules containing unsatisfied aggregates and negative literals
  2. Delete aggregates and negative literals from all other rules
- [Kemp, Stuckey 1991], [Gelfond 2002], [Dell'Armi, F., Ielpa, Leone, Pfeifer 2003]
- Many programs are aggregate stratified
- Use of aggregates often yields computational advantages
- But: Not all programs are aggregate stratified

Answer Sets for Aggregate-stratified Programs

- Basic Idea: Treat aggregate atoms like negative literals
- Reduct:
  1. Delete rules containing unsatisfied aggregates and negative literals
  2. Delete aggregates and negative literals from all other rules

- [Kemp, Stuckey 1991], [Gelfond 2002], [Dell'Armi, F., Ielpa, Leone, Pfeifer 2003]
- Many programs are aggregate stratified
- Use of aggregates often yields computational advantages
- But: Not all programs are aggregate stratified

Answer Sets for Aggregate-stratified Programs

- Basic Idea: Treat aggregate atoms like negative literals
- Reduct:
  1. Delete rules containing unsatisfied aggregates and negative literals
  2. Delete aggregates and negative literals from all other rules

- [Kemp, Stuckey 1991], [Gelfond 2002], [Dell'Armi, F., Ielpa, Leone, Pfeifer 2003]
- Many programs are aggregate stratified
- Use of aggregates often yields computational advantages
- But: Not all programs are aggregate stratified

Outline

Unstratification

- What happens when we consider unstratified aggregates?
- Can we just keep the simple semantic definition?

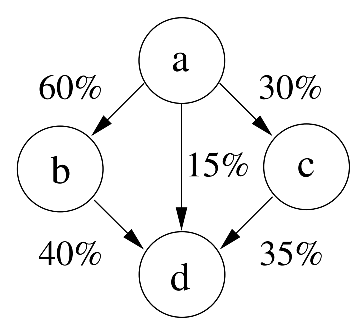
Company Control

Input: Set of companies and shares companies hold of other companies.
Output: Companies controlled (direct or indirect shares > 50%) by other companies
Encoding from the literature:

$$\begin{aligned}
&\mathit{controlsStk}(C1, C1, C2, P) \text{ :- } \mathit{ownsStk}(C1, C2, P).\\
&\mathit{controlsStk}(C1, C2, C3, P) \text{ :- } \mathit{controls}(C1, C2), \mathit{ownsStk}(C2, C3, P).\\
&\mathit{controls}(C1, C3) \text{ :- } \mathit{company}(C1), \mathit{company}(C3),\\
&\qquad\qquad \#\mathit{sum}\{P, C2 : \mathit{controlsStk}(C1, C2, C3, P)\} > 50.
\end{aligned}$$

Company Control, Instance 1

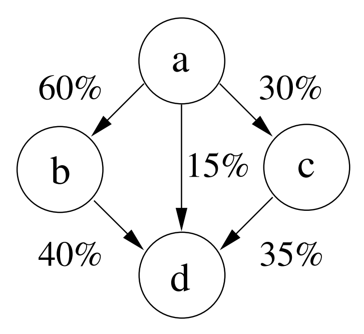
Example (Company Control, Instance 1)



{controlsStk(a,a,b,60),
controlsStk(a,a,c,30),
controlsStk(a,a,d,15),
controlsStk(b,b,d,40),
controlsStk(c,c,d,35),
controlsStk(a,b,d,40),
controls(a,b),
controls(a,d)}

Company Control, Instance 1

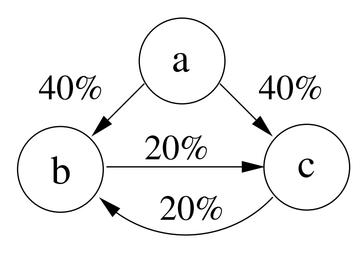
Example (Company Control, Instance 1)



{controlsStk(a,a,b,60),
controlsStk(a,a,c,30),
controlsStk(a,a,d,15),
controlsStk(b,b,d,40),
controlsStk(c,c,d,35),
controlsStk(a,b,d,40),
controls(a,b),
controls(a,d)}

Company Control, Instance 2

Example (Company Control, Instance 2)

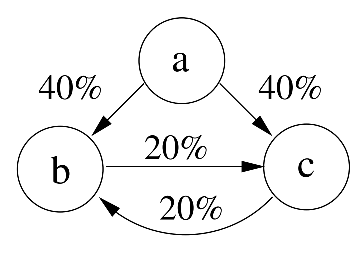


{controlsStk(a,a,b,40),
controlsStk(a,a,c,40),
controlsStk(b,b,c,20),
controlsStk(c,c,b,20)}

But also:
{controlsStk(a,b,c,20),
controlsStk(a,c,b,20),
controls(a,b),
controls(a,c)}

Company Control, Instance 2

Example (Company Control, Instance 2)

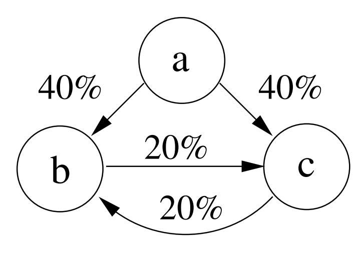


{controlsStk(a,a,b,40),
controlsStk(a,a,c,40),
controlsStk(b,b,c,20),
controlsStk(c,c,b,20)}

But also:
{controlsStk(a,b,c,20),
controlsStk(a,c,b,20),
controls(a,b),
controls(a,c)}

Company Control, Instance 2

Example (Company Control, Instance 2)



{controlsStk(a,a,b,40),
controlsStk(a,a,c,40),
controlsStk(b,b,c,20),
controlsStk(c,c,b,20)}

But also:

{controlsStk(a,b,c,20),
controlsStk(a,c,b,20),
controls(a,b),
controls(a,c)}

Essence

Example

$a \coloneq \#count\{\langle t : a\rangle\} < 1.$
No answer sets.
$a \coloneq \#count\{\langle t : a\rangle\} > 0.$
Answer sets: $\emptyset, \{a\}$?

$\#count\{\langle t : a\rangle\} < 1$ behaves like *not a*
$\#count\{\langle t : a\rangle\} > 0$ behaves like *a*
$\Rightarrow$ aggregates should not be treated like negative literals, but also not like positive literals

Essence

Example

$a \text{:-} \#count\{\langle t : a \rangle\} < 1.$
No answer sets.
$a \text{:-} \#count\{\langle t : a \rangle\} > 0.$
Answer sets: $\emptyset$, $\{a\}$?

$\#count\{\langle t : a \rangle\} < 1$ behaves like *not a*
$\#count\{\langle t : a \rangle\} > 0$ behaves like *a*
$\Rightarrow$ aggregates should not be treated like negative literals, but also not like positive literals

Monotonicity and Antimonotonicity

- Monotone Literals:
  truth for interpretation I implies truth for all $J \supseteq I$
- Antimonotone Literals:
  truth for interpretation J implies truth for all $I \subseteq J$
- Nonmonotone Literals:
  neither monotone nor antimonotone

Monotonicity: Examples

- $\#count\{\ldots\} \geq 1$ is monotone
- $\#count\{\ldots\} < 1$ is antimonotone
- $\#avg\{\ldots\} < 3$ is nonmonotone
- Positive standard literals are monotone
- Negative standard literals are antimonotone

FLP Semantics: Novel Reduct Definition

Definition of reduct according to [F.,Leone,Pfeifer 2004, F., Leone, Pfeifer 2011]:

- Delete rules with a false body literal.
- That's it!

Answer Set: Subset-minimal model of the reduct

Theorem

For aggregate-free programs, answer sets under this definition coincide with the ones defined in [Gelfond, Lifschitz 1991].

(Mostly) equivalent semantics defined in [Ferraris 2005], [Ferraris 2011].

FLP Semantics: Novel Reduct Definition

Definition of reduct according to [F.,Leone,Pfeifer 2004, F., Leone, Pfeifer 2011]:

- Delete rules with a false body literal.
- That's it!

Answer Set: Subset-minimal model of the reduct

Theorem

For aggregate-free programs, answer sets under this definition coincide with the ones defined in [Gelfond, Lifschitz 1991].

(Mostly) equivalent semantics defined in [Ferraris 2005], [Ferraris 2011].

FLP Semantics: Novel Reduct Definition

Definition of reduct according to [F.,Leone,Pfeifer 2004, F., Leone, Pfeifer 2011]:

- Delete rules with a false body literal.
- That's it!

Answer Set: Subset-minimal model of the reduct

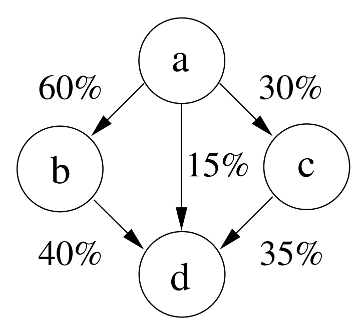
Theorem

For aggregate-free programs, answer sets under this definition coincide with the ones defined in [Gelfond, Lifschitz 1991].

(Mostly) equivalent semantics defined in [Ferraris 2005], [Ferraris 2011].

Company Control, Instance 1

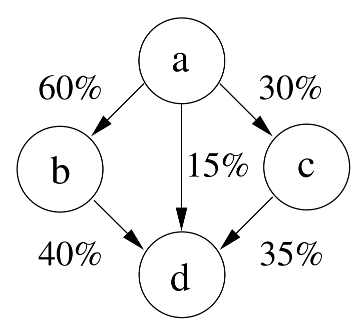
Example (Company Control, Instance 1)



{controlsStk(a,a,b,60),
controlsStk(a,a,c,30),
controlsStk(a,a,d,15),
controlsStk(b,b,d,40),
controlsStk(c,c,d,35),
controlsStk(a,b,d,40),
controls(a,b),
controls(a,d)}

Company Control, Instance 1

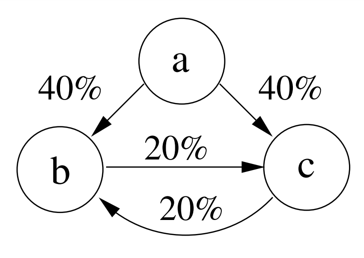
Example (Company Control, Instance 1)



{controlsStk(a,a,b,60),
controlsStk(a,a,c,30),
controlsStk(a,a,d,15),
controlsStk(b,b,d,40),
controlsStk(c,c,d,35),
controlsStk(a,b,d,40),
controls(a,b),
controls(a,d)}

Company Control, Instance 2

Example (Company Control, Instance 2)

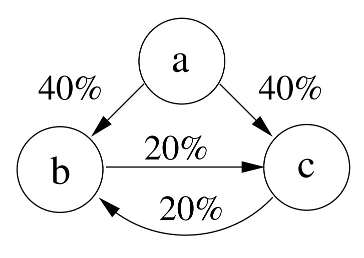


{controlsStk(a,a,b,40),
controlsStk(a,a,c,40),
controlsStk(b,b,c,20),
controlsStk(c,c,b,20)}

Only this answer set.

Company Control, Instance 2

Example (Company Control, Instance 2)



{controlsStk(a,a,b,40),
controlsStk(a,a,c,40),
controlsStk(b,b,c,20),
controlsStk(c,c,b,20)}

Only this answer set.

Small examples

Example

$a \mathrel{:-} \#count\{\langle t : a\rangle\} < 1.$

No answer sets.

$a \mathrel{:-} \#count\{\langle t : a\rangle\} > 0.$

Answer sets: only $\emptyset$

PSP Semantics

- Alternative semantics for unstratified aggregates:
  - [Pelov 2004], [Son, Pontelli 2007], [Shen, Wang 2012]
  - Definitions use different operator-based techniques
- Evaluate aggregates for a pair of interpretations
- $(I, J) \models A$ iff $K \models A$ for all $I \subseteq K \subseteq J$.
- PSP answer sets are those I that are fixpoints of $K_\Pi^I \Uparrow \emptyset$
- Operator $K_\Pi^I(X)$ collects heads of rules for which $(X, I) \models A$ for all body atoms

Small examples: PSP

Example

$a \coloneq \#count\{\langle t : a\rangle\} < 1.$
No answer sets.
$K_{\Pi}^{\emptyset}(\emptyset) = \{a\}$
$K_{\Pi}^{\emptyset}(\{a\}) = \{a\}$
$K_{\Pi}^{\{a\}}(\emptyset) = \emptyset$

$a \coloneq \#count\{\langle t : a\rangle\} > 0.$
Answer sets: only $\emptyset$
$K_{\Pi}^{\emptyset}(\emptyset) = \emptyset$
$K_{\Pi}^{\{a\}}(\emptyset) = \emptyset$

Outline

FLP and PSP

- Small examples could suggest that they coincide.
- But this is not true in general.

Example

$$
\begin{array}{l}
p(1) \coloneq \#avg\{X : p(X)\} \geq 0. \\
p(1) \coloneq p(-1). \\
p(-1) \coloneq p(1).
\end{array}
$$

FLP answer sets: $\{p(1), p(-1)\}$
PSP answer sets: none

In general: each PSP answer set is also FLP, but not necessarily vice versa.
Classes on which the semantics coincide?

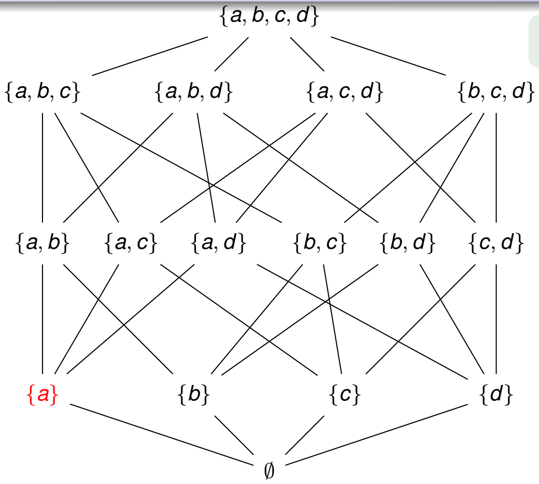
Aggregate-stratified Programs

Easy observation:

Theorem

FLP and PSP semantics coincide on aggregate-stratified programs.

Monotone, antimonotone, convex



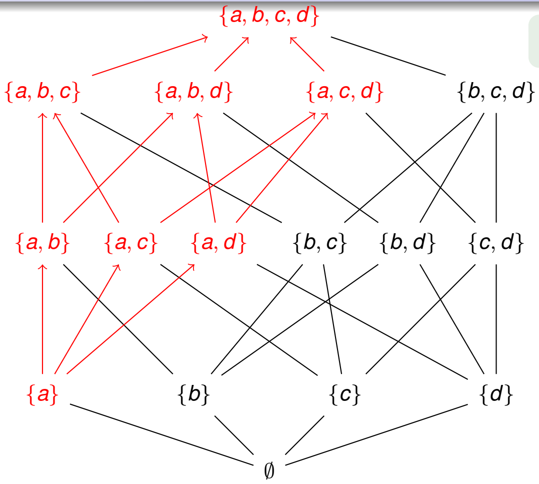
$$1 \leq \#count\{a, b, c, d\}$$

S is monotone if

$I \models S \wedge J \models S \implies$

$K \models S \quad \forall K \in \ \uparrow I \cap \downarrow J$

Monotone, antimonotone, convex



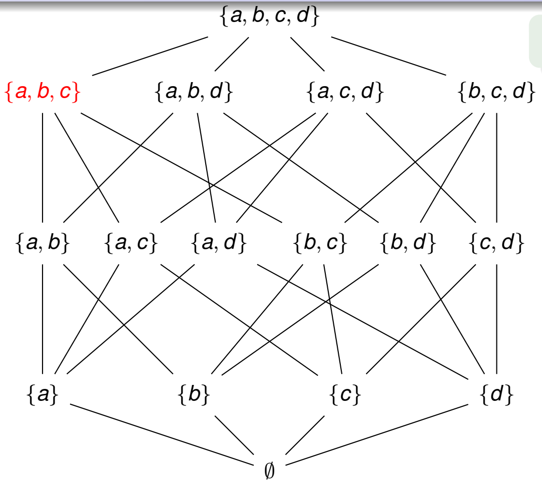
$$1 \leq \#count\{a, b, c, d\}$$

S is monotone if

$I \models S \wedge J \models S \implies$

$K \models S \quad \forall K \in \; \uparrow I \cap \downarrow J$

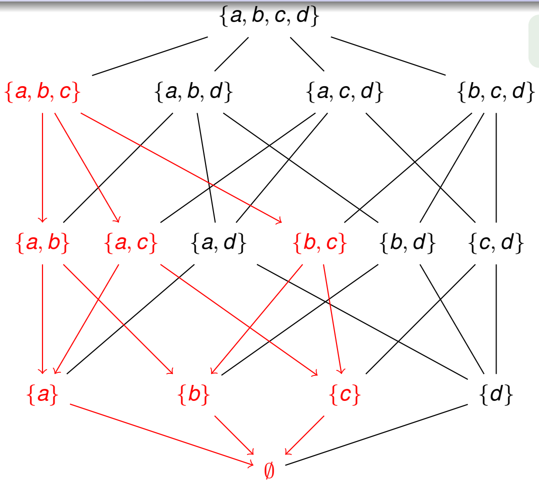
Monotone, antimonotone, convex



$$\#count\{a,b,c,d\} \leq 3$$

S is antimonotone if
$I \models S \wedge J \models S \implies$
$K \models S \quad \forall K \in \uparrow I \cap \downarrow J$

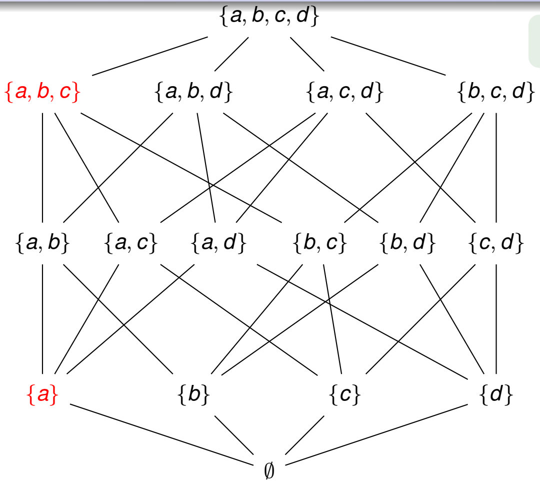
Monotone, antimonotone, convex



$$\#count\{a, b, c, d\} \leq 3$$

S is antimonotone if
$I \models S \wedge J \models S \implies$
$K \models S \quad \forall K \in {\uparrow} I \cap {\downarrow} J$

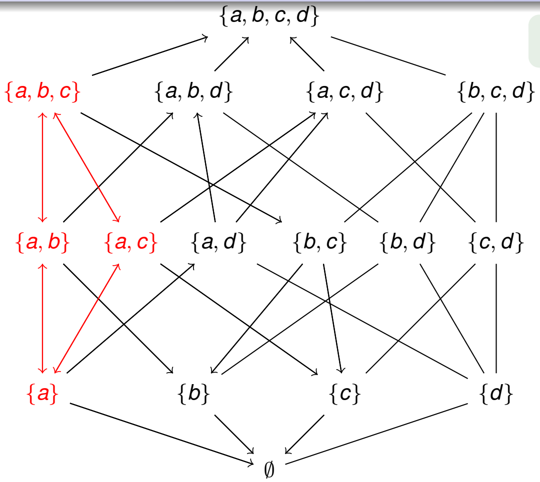
Monotone, antimonotone, convex



$$1 \leq \#count\{a, b, c, d\} \leq 3$$

S is convex if
$I \models S \wedge J \models S \implies$
$K \models S \quad \forall K \in \ \uparrow I \cap \downarrow J$

Monotone, antimonotone, convex



$$1 \leq \#count\{a, b, c, d\} \leq 3$$

S is convex if
$I \models S \wedge J \models S \implies$
$K \models S \quad \forall K \in \ \uparrow I \cap \downarrow J$

Convex Programs

Implicit in [Liu, Truszczyński 2006]:

Theorem

FLP and PSP semantics coincide on programs containing only convex aggregates.

Corollary

FLP and PSP semantics coincide on programs containing only monotone and antimonotone aggregates.

Outline

Problem

Cautious reasoning over variable-free programs and polynomial-time computable aggregate functions.

Input: A ground program $\mathcal{P}$ and a ground standard atom A.
Output: Is A true in all FLP answer sets of $\mathcal{P}$?

Similar results for related problems (answer set existence, brave reasoning).

Problem

Cautious reasoning over variable-free programs and polynomial-time computable aggregate functions.

Input: A ground program $\mathcal{P}$ and a ground standard atom A.
Output: Is A true in all FLP answer sets of $\mathcal{P}$?

Similar results for related problems (answer set existence, brave reasoning).

Complexity of Cautious Reasoning

	$\{\}$	$\{\mathtt{not}\}$	$\{\lor\}$	$\{\mathtt{not}, \lor\}$
$\{\mathbf{M}\}$	P	co-NP	co-NP	Π_2^P
$\{\mathbf{S}\}$	P	co-NP	Π_2^P	Π_2^P
$\{\mathbf{C}\}$	co-NP	co-NP	Π_2^P	Π_2^P
$\{\mathbf{N}\}$	Π_2^P	Π_2^P	Π_2^P	Π_2^P

$\{\mathbf{M}\}$: monotone aggregates $\{\mathbf{S}\}$: stratified aggregates $\{\mathbf{C}\}$: convex aggregates $\{\mathbf{N}\}$: non-convex aggregates

Complexity: FLP and PSP

- Results for PSP: analogous to FLP
- Slight difference for non-convex aggregates [Alviano, F. 2013]:
  - One non-convex aggregate is sufficient to express any problem in Π_2^P with FLP
  - Arbitrarily many are needed to do the same with PSP

Only Aggregates?

All of the results presented here also apply to other extensions of ASP that have constructs that evaluate truth on sets of basic atoms, for example:

- Abstract Constraint Atoms (U, V) [Marek, Remmel 2004]
- HEX programs [Eiter et al. 2005]
- Nested Expressions [Lifschitz et al. 1999]
- Generalized Quantifiers [Lindström 1966]
- Cardinality and Weight Constraints [Simons 2000]

Summary

- Aggregates in ASP
- FLP and PSP Semantics
- Properties: Coincidence and Complexity

More

- Beyond FLP and PSP? [Alviano, F. 2015]

Example

$$a \mathrel{:-} \#count\{a, b\} != 1.$$
$$b \mathrel{:-} \#count\{a, b\} != 1.$$

No FLP, no PSP answer sets! Unintuitive?

- Reduce programs with non-convex aggregates to programs with monotone aggregates in a compact way. [Alviano, F., Gebser 2015]
- System support: results in [Alviano, F., Gebser 2015] allow for non-convex aggregates in gringo $\geq$ 4.5.