Operator-based semantics for logic programs: recent advances

Jesse Heyninck

November 26, 2024

Open Universiteit, the Netherlands
University of Cape Town, South Africa

Why Operator-Based Semantics?

- Powerful framework for defining and studying semantics rule-based languages

  If you have an idea of how to eavluate rule bodies in a three-valued setting, the framework does the rest of you.
- Single unifying framework that allows to derive most common logic programming semantics (and the semantics of default logic, autoepistemic logic, abstract argumentation, ...)
- Gives a natural justification of the well-founded and stable model semantics.
- Applied in grounders such as gringo.
- Useful for explanations.

Goal of this talk

- Give an overview of the operator-based view on semantics of logic programming [VGRS91, Fit06, DMT00].
- Given an overview of some recent developments (by Bart, Ofer and myself) in this area:
    - Non-deterministic operators,
    - Disjunctive logic programs,
    - Choice logic programs.

Goals and Structure

Syntax of Logic Programs

Semantics of Positive Programs

Semantics of Normal Logic Programs

Non-Deterministic Operators

Stable Semantics

- Stable Semantics for Deterministic Operators
- Stable Non-Deterministic Operators

Approximation Fixpoint Theory

Round up

Syntax of Logic Programs

Syntax of Logic Programs

Set of atoms $\mathcal{A} = \{a, b, c, p, q, r, a_1, a_2, \ldots\}$

$$a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m$$

- Program is a set of rules.
- Rule is positive if $m = 0$.
- Program is positive if all the rules are positive.

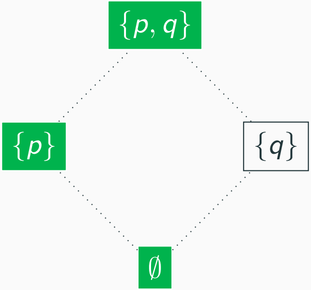
Semantics of Positive Programs

What are the semantics of logic programs?

$$p \leftarrow q.$$

Classical models? $\emptyset, \{p\}, \{p, q\}$.

Notice: a formula follows from every classical model if it follows from the minimal model $\emptyset$.



What are the semantics of logic programs?

$$p \leftarrow q., \quad q \leftarrow .$$

What are the semantics of logic programs?

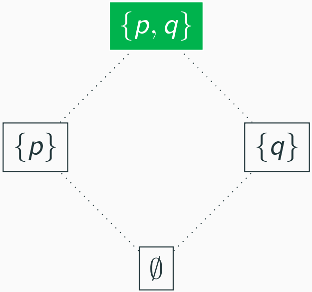
$$p \leftarrow q., \quad q \leftarrow .$$

Classical models? $\emptyset, \{p\}, \{p, q\}$.

What are the semantics of logic programs?

$$p \leftarrow q., \quad q \leftarrow .$$

Classical models? $\emptyset, \{p\}, \{p, q\}$.



$T_{\mathcal{P}}$-operator

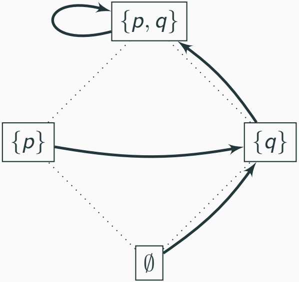
$$T_{\mathcal{P}} : \wp(\mathcal{A}_{\mathcal{P}}) \mapsto \wp(\mathcal{A}_{\mathcal{P}})$$

$$T_{\mathcal{P}}(x) = \{a \mid a \leftarrow b_1, \ldots, b_n \in \mathcal{P} \text{ and } b_1, \ldots, b_n \in x\}$$

Example

$$\mathcal{P} = \{p \leftarrow q., \quad q \leftarrow .\}$$

x	$\emptyset$	$\{p\}$	$\{q\}$	$\{p, q\}$
$T_{\mathcal{P}}(x)$	$\{q\}$	$\{q\}$	$\{p, q\}$	$\{p, q\}$



Models and Fixpoints

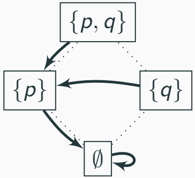
Definition
x is a pre-fixpoint of $T_{\mathcal{P}}$ if $T_{\mathcal{P}}(x) \subseteq x$.

Intuition: everything I can derive from x using $\mathcal{P}$ is in x.

Models of $\mathcal{P}$.

Example

$$\mathcal{P} = \{p \leftarrow q.\}$$



Models and Fixpoints

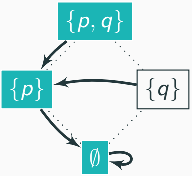
Definition
x is a pre-fixpoint of $T_{\mathcal{P}}$ if $T_{\mathcal{P}}(x) \subseteq x$.

Intuition: everything I can derive from x using $\mathcal{P}$ is in x.

Models of $\mathcal{P}$.

Example

$$\mathcal{P} = \{p \leftarrow q.\}$$



Models and Fixpoints

Definition
x is a pre-fixpoint of $T_{\mathcal{P}}$ if $T_{\mathcal{P}}(x) \subseteq x$.

Intuition: everything I can derive from x using $\mathcal{P}$ is in x.

Models of $\mathcal{P}$.

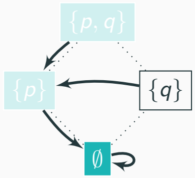
Definition
x is a fixpoint of $T_{\mathcal{P}}$ if $T_{\mathcal{P}}(x) = x$.

Intuition: same, plus everything in x is derivable.

Supported models of $\mathcal{P}$.

Example

$$\mathcal{P} = \{p \leftarrow q.\}$$



Models and Fixpoints

Definition
x is a pre-fixpoint of $T_{\mathcal{P}}$ if $T_{\mathcal{P}}(x) \subseteq x$.

Intuition: everything I can derive from x using $\mathcal{P}$ is in x.

Models of $\mathcal{P}$.

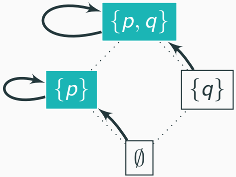
Definition
x is a fixpoint of $T_{\mathcal{P}}$ if $T_{\mathcal{P}}(x) = x$.

Intuition: same, plus everything in x is derivable.

Supported models of $\mathcal{P}$.

Example

$$\mathcal{P} = \{p \leftarrow ., \quad q \leftarrow q.\}$$



Models and Fixpoints

- For positive programs, $T_{\mathcal{P}}$ has a unique least fixpoint x.
- It is also the least pre-fixpoint.
- We can compute it by iterating $T_{\mathcal{P}}$ starting from $\emptyset$:
  $T_{\mathcal{P}}(\dots T_{\mathcal{P}}(\emptyset)\dots) = \bigcup_{i \geq 0} T^i_{\mathcal{P}}(\emptyset)$
  And this is possible in polynomial time.

Models and Fixpoints

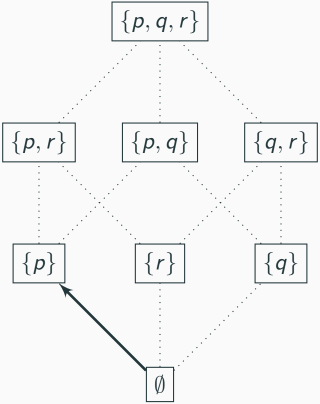
- For positive programs, $T_{\mathcal{P}}$ has a unique least fixpoint x.
  - Any fixpoint y of $T_{\mathcal{P}}$ will be a superset: $x \subseteq y$.
- It is also the least pre-fixpoint.
  - Any pre-fixpoint y of $T_{\mathcal{P}}$ will be a superset: $x \subseteq y$.
  - $\mathcal{P}$ has a least model,
    i.e. any model of $\mathcal{P}$ includes x.
  - If something follows from every model of $\mathcal{P}$, it follows from x.
- We can compute it by iterating $T_{\mathcal{P}}$ starting from $\emptyset$:
  $T_{\mathcal{P}}(\ldots T_{\mathcal{P}}(\emptyset)\ldots)$
  And this is possible in polynomial time.

Models and Fixpoints

- For positive programs, $T_{\mathcal{P}}$ has a unique least fixpoint x.
  - Any fixpoint y of $T_{\mathcal{P}}$ will be a superset: $x \subseteq y$.
- It is also the least pre-fixpoint.
  - Any pre-fixpoint y of $T_{\mathcal{P}}$ will be a superset: $x \subseteq y$.
  - $\mathcal{P}$ has a least model,
    i.e. any model of $\mathcal{P}$ includes x.
  - If something follows from every model of $\mathcal{P}$, it follows from x.
- We can compute it by iterating $T_{\mathcal{P}}$ starting from $\emptyset$:
  $T_{\mathcal{P}}(\ldots T_{\mathcal{P}}(\emptyset)\ldots)$
  And this is possible in polynomial time.
  Underlying result: A $\subseteq$-monotonic operator over a complete lattice admits a least fixpoint.

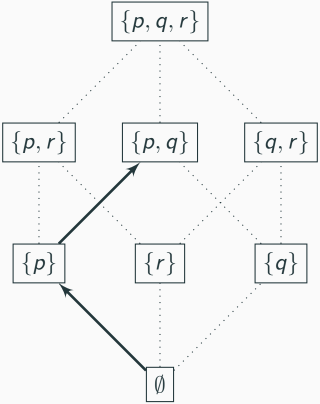
Least fixpoint computation

$$\mathcal{P} = \{p \leftarrow . \quad q \leftarrow p. \quad r \leftarrow p, q.\}$$



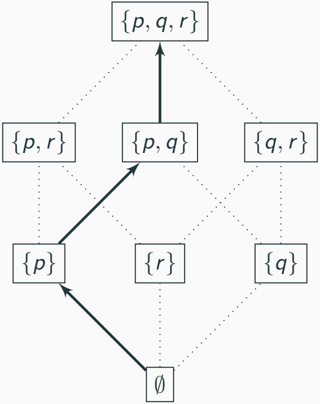
Least fixpoint computation

$$\mathcal{P} = \{p \leftarrow . \quad q \leftarrow p. \quad r \leftarrow p, q.\}$$



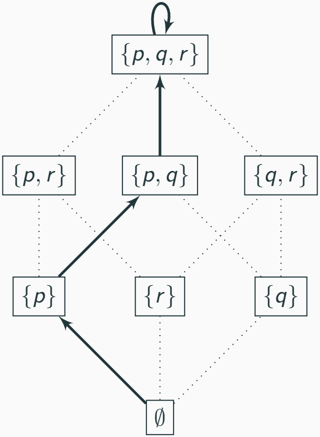
Least fixpoint computation

$$\mathcal{P} = \{p \leftarrow . \quad q \leftarrow p. \quad r \leftarrow p, q.\}$$



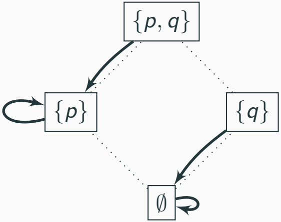
Least fixpoint computation

$$\mathcal{P} = \{p \leftarrow . \quad q \leftarrow p. \quad r \leftarrow p, q.\}$$



Least fixpoint $\neq$ unique fixpoint

Example ($\mathcal{P} = \{p \leftarrow p.\}$)



Semantics of Normal Logic Programs

Enters Negation

$$p \leftarrow \neg q$$

How to extend our operator?

Enters Negation

$$p \leftarrow \neg q$$

How to extend our operator?

Easy:

$$\begin{aligned} T_{\mathcal{P}}(x) = \{a \mid \quad & a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \text{ and} \\ & b_1, \ldots, b_n \in x, \text{ and } c_1, \ldots, c_m \notin x\} \end{aligned}$$

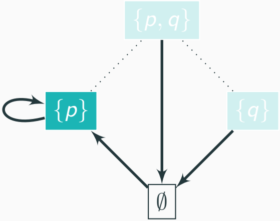
Enters Negation

$$p \leftarrow \neg q$$

How to extend our operator?

Easy:

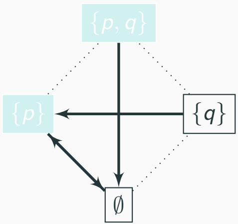
$$T_{\mathcal{P}}(x) = \{a \mid \quad a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \text{ and } b_1, \ldots, b_n \in x, \text{ and } c_1, \ldots, c_m \notin x\}$$



Great, thanks for your attention

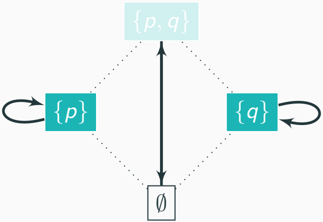
Great, thanks for your attention

$\mathcal{P} = \{p \leftarrow \neg p\}$



No unique fixpoint

$\mathcal{P} = \{p \leftarrow \neg q; q \leftarrow \neg p\}$



Problems with negation

- There might not be a fixpoint.
- There might be multiple minimal fixpoints.
- We don't know how to find fixpoints.

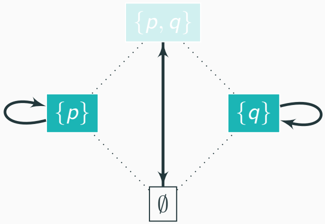
Problems with negation

- There might not be a fixpoint.
- There might be multiple minimal fixpoints.
- We don't know how to find fixpoints.
- Anyone sees what went wrong with our operator?

Problems with negation

- There might not be a fixpoint.
- There might be multiple minimal fixpoints.
- We don't know how to find fixpoints.
- Anyone sees what went wrong with our operator?

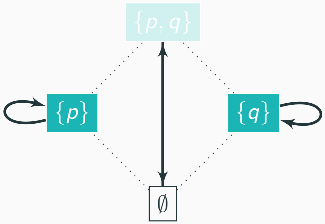
$\Rightarrow$ It is not a monotonic operator



Problems with negation

- There might not be a fixpoint.
- There might be multiple minimal fixpoints.
- We don't know how to find fixpoints.
- Anyone sees what went wrong with our operator?

$\Rightarrow$ It is not a monotonic operator



Solution: approximations of operators that are $\leq_i$-monotonic.

Approximations

- Pairs of sets of atoms (x, y).
  - x contains all atoms that are definitely true.
  - y contains all atoms that are possibly true.

Approximations

- Pairs of sets of atoms (x, y).
  - x contains all atoms that are definitely true.
  - y contains all atoms that are possibly true.
- How to compare such pairs of sets?
  - $(x_1, y_1) \leq_t (x_2, y_2)$ if $x_1 \subseteq x_2$ and $y_1 \subseteq y_2$.
  - $(x_1, y_1) \leq_i (x_2, y_2)$ if $x_1 \subseteq x_2$ and $y_2 \subseteq y_1$.

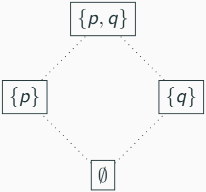
Approximations

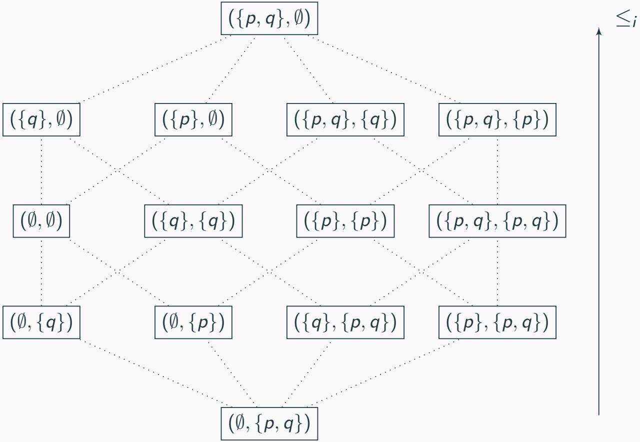
- Pairs of sets of atoms (x, y).
  - x contains all atoms that are definitely true.
  - y contains all atoms that are possibly true.
- How to compare such pairs of sets?
  - $(x_1, y_1) \leq_t (x_2, y_2)$ if $x_1 \subseteq x_2$ and $y_1 \subseteq y_2$.
  - $(x_1, y_1) \leq_i (x_2, y_2)$ if $x_1 \subseteq x_2$ and $y_2 \subseteq y_1$.

Example (Given $\mathcal{A} = \{p, q, r\}$)
$(\{p\}, \{p, q\})$: *p is true and q can be true.*

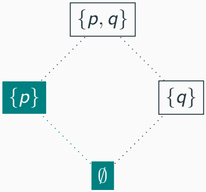
$(\{p\}, \{p\}) \leq_t (\{p\}, \{p, q\})$
$(\{p\}, \{p, q\}) \leq_i (\{p\}, \{p\})$

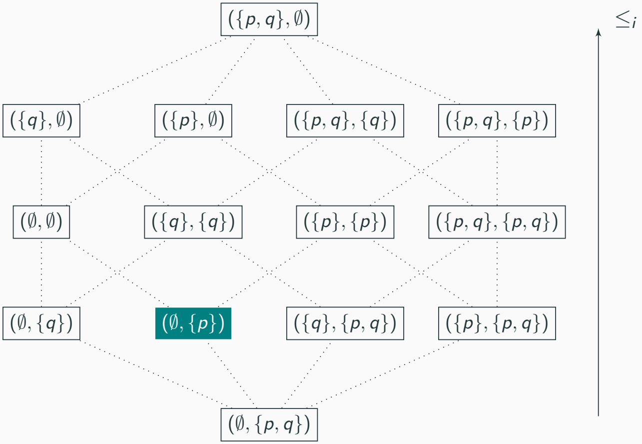
Graphical Depiction



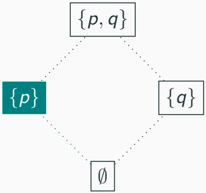


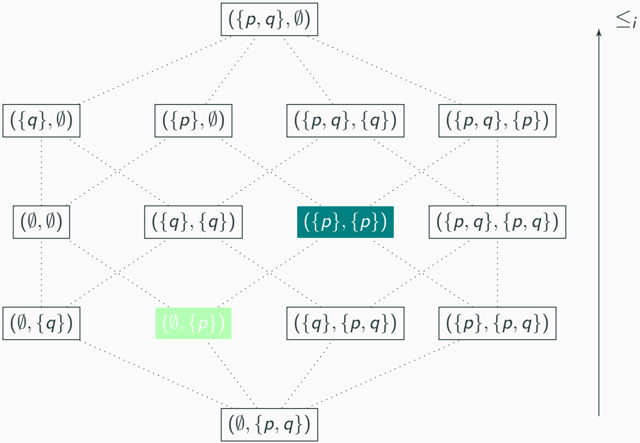
Graphical Depiction



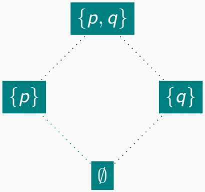


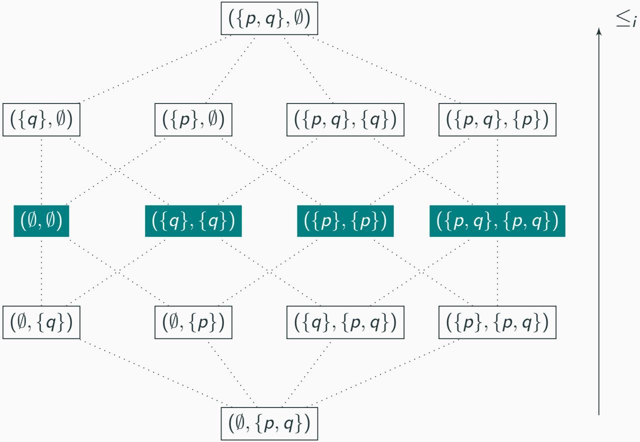
Graphical Depiction



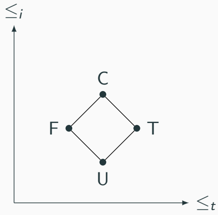


Graphical Depiction





Approximations as Four-Valued Interpretations



- $-\mathsf{F} = \mathsf{T}$, $-\mathsf{T} = \mathsf{F}$, $-\mathsf{U} = \mathsf{U}$ and $-\mathsf{C} = \mathsf{C}$
- $(x, y)(p) = \begin{cases} \mathsf{T} & \text{if } p \in x \text{ and } p \in y, \\ \mathsf{U} & \text{if } p \notin x \text{ and } p \in y, \\ \mathsf{F} & \text{if } p \notin x \text{ and } p \notin y, \\ \mathsf{C} & \text{if } p \in x \text{ and } p \notin y. \end{cases}$
- $(x, y)(\neg\phi) = -(x, y)(\phi)$,
- $(x, y)(\psi \wedge \phi) = lub_{\leq_t}\{(x, y)(\phi), (x, y)(\psi)\}$,
- $(x, y)(\psi \vee \phi) = glb_{\leq_t}\{(x, y)(\phi), (x, y)(\psi)\}$.

Example

$(\{p\}, \{p, q\})(p) = \mathsf{T} \quad (\{p\}, \{p, q\})(q) = \mathsf{U} \quad (\{p\}, \{p, q\})(r) = \mathsf{F}$.

$(\{p\}, \{p, q\})(\neg p) = \mathsf{F} \quad (\{p\}, \{p, q\})(\neg q) = \mathsf{U}$

$(\{p\}, \{p, q\})(p \wedge q) = \mathsf{U} \quad (\{p\}, \{p, q\})(q \vee r) = \mathsf{U}$

Approximating $T_{\mathcal{P}}$ (from below)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

Approximating $T_{\mathcal{P}}$ (from below)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

$$\mathcal{IC}^{l}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ b_1, \ldots, b_n \in x \text{ and } c_1, \ldots, c_m \notin y\}$$

Approximating $T_{\mathcal{P}}$ (from below)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

$$\mathcal{IC}^{l}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ b_1, \ldots, b_n \in x \text{ and } c_1, \ldots, c_m \notin y\}$$

or, equivalently

$$\mathcal{IC}^{l}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ (x, y)(b_1 \wedge \ldots \ldots b_n \wedge \neg c_1 \wedge \ldots \wedge \neg c_m) \in \{\mathsf{T}, \mathsf{C}\}\}$$

Approximating $T_{\mathcal{P}}$ (from below)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

$$\mathcal{IC}^{l}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ b_1, \ldots, b_n \in x \text{ and } c_1, \ldots, c_m \notin y\}$$

or, equivalently

$$\mathcal{IC}^{l}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ (x, y)(b_1 \wedge \ldots \ldots b_n \wedge \neg c_1 \wedge \ldots \wedge \neg c_m) \in \{\mathsf{T}, \mathsf{C}\}\}$$

Example ($\{p \leftarrow p, \neg q\}$)
$\mathcal{IC}^{l}_{\mathcal{P}}(\{p\}, \{p, q\}) = \emptyset$
$\mathcal{IC}^{l}_{\mathcal{P}}(\{p\}, \{p\}) = \{p\}$

Approximating $T_{\mathcal{P}}$ (from above)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

Approximating $T_{\mathcal{P}}$ (from above)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

$$\mathcal{IC}^{u}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P},$$
$$b_1, \ldots, b_n \in y \text{ and } c_1, \ldots, c_m \notin x\}$$

Approximating $T_{\mathcal{P}}$ (from above)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

$$\mathcal{IC}^{u}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ b_1, \ldots, b_n \in y \text{ and } c_1, \ldots, c_m \notin x\}$$

or, equivalently

$$\mathcal{IC}^{u}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ (x, y)(b_1 \wedge \ldots \ldots b_n \wedge \neg c_1 \wedge \ldots \wedge \neg c_m) \in \{\mathsf{U}, \mathsf{T}\}\}$$

Approximating $T_{\mathcal{P}}$ (from above)

$$\mathcal{IC}_{\mathcal{P}} : \mathcal{A} \times \mathcal{A} \mapsto \mathcal{A} \times \mathcal{A}$$

We input an approximation and output an approximation.

$$\mathcal{IC}^{u}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ b_1, \ldots, b_n \in y \text{ and } c_1, \ldots, c_m \notin x\}$$

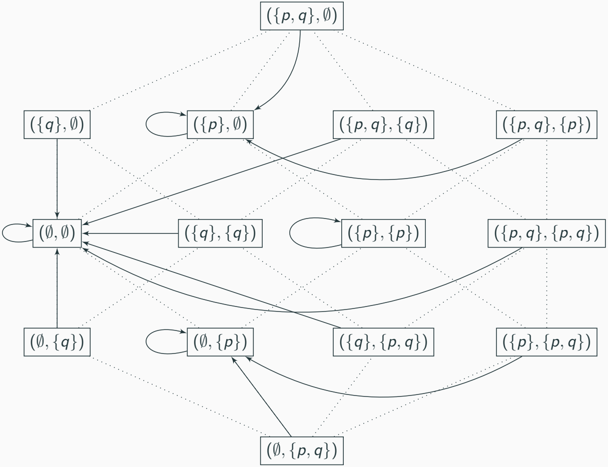
or, equivalently

$$\mathcal{IC}^{u}_{\mathcal{P}}(x, y) = \{a \in \mathcal{A} \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}, \\ (x, y)(b_1 \wedge \ldots \ldots b_n \wedge \neg c_1 \wedge \ldots \wedge \neg c_m) \in \{\mathsf{U}, \mathsf{T}\}\}$$

Example ($\{p \leftarrow p, \neg q\}$)
$\mathcal{IC}^{u}_{\mathcal{P}}(\{p\}, \{p, q\}) = \{p\}$
$\mathcal{IC}^{u}_{\mathcal{P}}(\{p\}, \{p\}) = \{p\}$

The Approximation Operator $\mathcal{IC}_\mathcal{P}$ (for $\mathcal{P} = \{p \leftarrow p, \neg q\}$)

$$\mathcal{IC}_\mathcal{P}(x, y) = (\mathcal{IC}^l_\mathcal{P}(x, y), \mathcal{IC}^u_\mathcal{P}(x, y))$$



Properties of $\mathcal{IC}_\mathcal{P}$

- $\mathcal{IC}_\mathcal{P}$ approximates $T_\mathcal{P}$:

$\mathcal{IC}_\mathcal{P}(x, x) = (T_\mathcal{P}(x), T_\mathcal{P}(x))$ for any $x \subseteq \mathcal{A}$.

Properties of $\mathcal{IC}_\mathcal{P}$

- $\mathcal{IC}_\mathcal{P}$ approximates $T_\mathcal{P}$:

$\mathcal{IC}_\mathcal{P}(x, x) = (T_\mathcal{P}(x), T_\mathcal{P}(x))$ for any $x \subseteq \mathcal{A}$.

- $\mathcal{IC}_\mathcal{P}$ is $\leq_i$-monotonic:

if $(x_1, y_1) \leq_i (x_2, y_2)$ then $\mathcal{IC}_\mathcal{P}(x_1, y_1) \leq_i \mathcal{IC}_\mathcal{P}(x_2, y_2)$.

Properties of $\mathcal{IC}_\mathcal{P}$

- $\mathcal{IC}_\mathcal{P}$ approximates $T_\mathcal{P}$:

$\mathcal{IC}_\mathcal{P}(x, x) = (T_\mathcal{P}(x), T_\mathcal{P}(x))$ for any $x \subseteq \mathcal{A}$.

- $\mathcal{IC}_\mathcal{P}$ is $\leq_i$-monotonic:

if $(x_1, y_1) \leq_i (x_2, y_2)$ then $\mathcal{IC}_\mathcal{P}(x_1, y_1) \leq_i \mathcal{IC}_\mathcal{P}(x_2, y_2)$.

We say $\mathcal{IC}_\mathcal{P}$ is an approximation operator. It is also symmetric, in the sense that $\mathcal{IC}_\mathcal{P}(x, y) = (\mathcal{IC}^l_\mathcal{P}(x, y), \mathcal{IC}^l_\mathcal{P}(y, x))$.

Kripke-Kleene Fixpoint

The $\leq_i$-monotonicity is our *indulgentia* back into Tarski's heaven:

Proposition
$\mathcal{IC}_\mathcal{P}$ *has a least fixpoint, obtainable as* $\bigsqcup_{i \geq 0} \mathcal{IC}^i_\mathcal{P}(\emptyset, \mathcal{A})$[1].

[1] $(x_1, y_1) \sqcup (x_2, y_2) = (x_1 \cup x_2, y_1 \cap y_2)$.

Kripke-Kleene Fixpoint

The $\leq_i$-monotonicity is our *indulgentia* back into Tarski's heaven:

Proposition
$\mathcal{IC}_{\mathcal{P}}$ has a least fixpoint, obtainable as $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$[1].

$\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$ is called the Kripke-Kleene Fixpoint

[1] $(x_1, y_1) \sqcup (x_2, y_2) = (x_1 \cup x_2, y_1 \cap y_2)$.

Kripke-Kleene Fixpoint

The $\leq_i$-monotonicity is our *indulgentia* back into Tarski's heaven:

Proposition
$\mathcal{IC}_{\mathcal{P}}$ has a least fixpoint, obtainable as $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$[1].

$\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$ is called the Kripke-Kleene Fixpoint

Proposition
For any fixpoint of $x = T_{\mathcal{P}}(x)$, $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A}) \leq_i (x, x)$.

[1] $(x_1, y_1) \sqcup (x_2, y_2) = (x_1 \cup x_2, y_1 \cap y_2)$.

Kripke-Kleene Fixpoint

The $\leq_i$-monotonicity is our *indulgentia* back into Tarski's heaven:

Proposition
$\mathcal{IC}_{\mathcal{P}}$ has a least fixpoint, obtainable as $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$[1].

$\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$ is called the Kripke-Kleene Fixpoint

Proposition
For any fixpoint of $x = T_{\mathcal{P}}(x)$, $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A}) \leq_i (x, x)$.

Proposition
$\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$ is consistent (i.e. where $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A}) = (x, y)$, $x \subseteq y$).

---

[1] $(x_1, y_1) \sqcup (x_2, y_2) = (x_1 \cup x_2, y_1 \cap y_2)$.

Kripke-Kleene Fixpoint

The $\leq_i$-monotonicity is our *indulgentia* back into Tarski's heaven:

Proposition
$\mathcal{IC}_{\mathcal{P}}$ has a least fixpoint, obtainable as $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$[1].

$\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$ is called the Kripke-Kleene Fixpoint

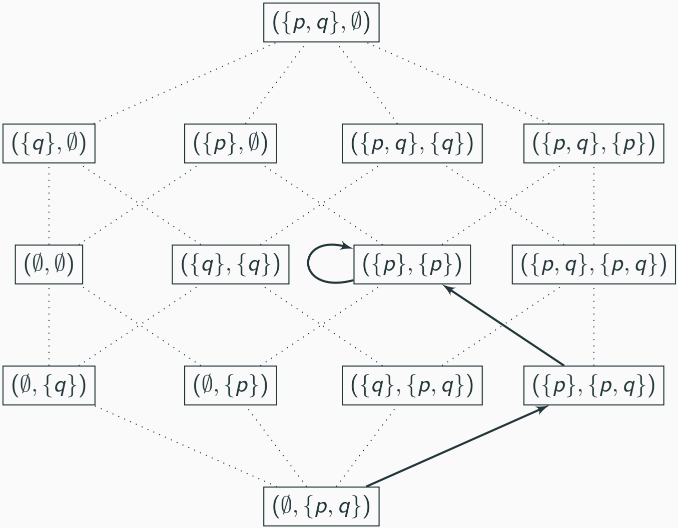
Proposition
For any fixpoint of $x = T_{\mathcal{P}}(x)$, $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A}) \leq_i (x, x)$.

Proposition
$\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A})$ is consistent (i.e. where $\bigsqcup_{i \geq 0} \mathcal{IC}^i_{\mathcal{P}}(\emptyset, \mathcal{A}) = (x, y)$, $x \subseteq y$).

If $(x, y) = \mathcal{IC}_{\mathcal{P}}(x, y)$ then we call it a partial supported model.

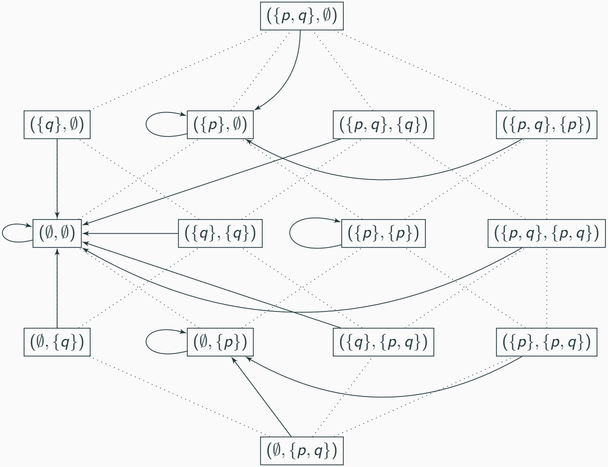
[1] $(x_1, y_1) \sqcup (x_2, y_2) = (x_1 \cup x_2, y_1 \cap y_2)$.

Example: $\mathcal{P} = \{p \leftarrow; q \leftarrow \neg p\}$



The Approximation Operator $\mathcal{IC}_\mathcal{P}$ (for $\mathcal{P} = \{p \leftarrow p, \neg q\}$)

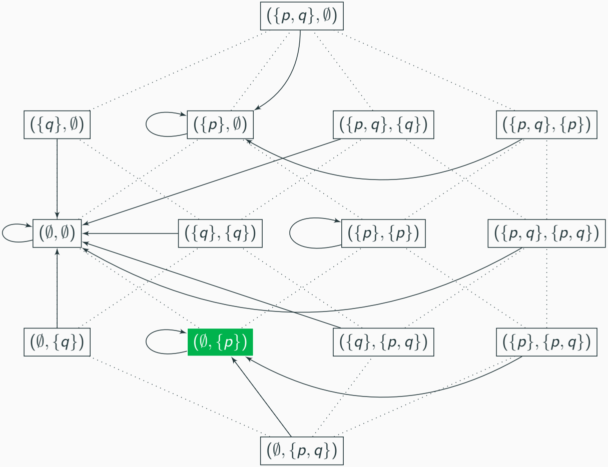
$$\mathcal{IC}_\mathcal{P}(x, y) = (\mathcal{IC}^l_\mathcal{P}(x, y), \mathcal{IC}^u_\mathcal{P}(x, y))$$



The Approximation Operator $\mathcal{IC}_\mathcal{P}$ (for $\mathcal{P} = \{p \leftarrow p, \neg q\}$)

$$\mathcal{IC}_\mathcal{P}(x, y) = (\mathcal{IC}^l_\mathcal{P}(x, y), \mathcal{IC}^u_\mathcal{P}(x, y))$$

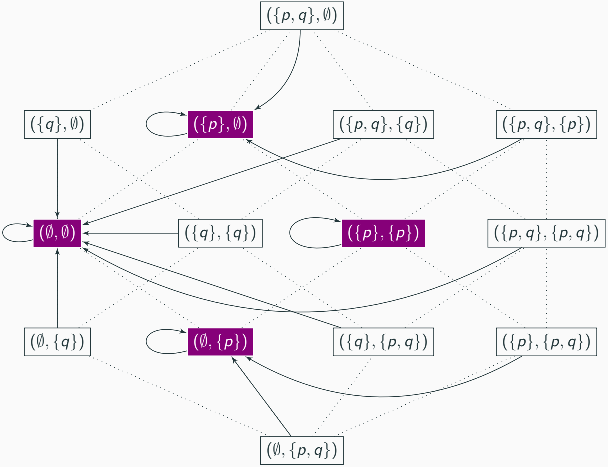
Kripke-Kleene Fixpoint



The Approximation Operator $\mathcal{IC}_{\mathcal{P}}$ (for $\mathcal{P} = \{p \leftarrow p, \neg q\}$)

$$\mathcal{IC}_{\mathcal{P}}(x, y) = (\mathcal{IC}^{l}_{\mathcal{P}}(x, y), \mathcal{IC}^{u}_{\mathcal{P}}(x, y))$$

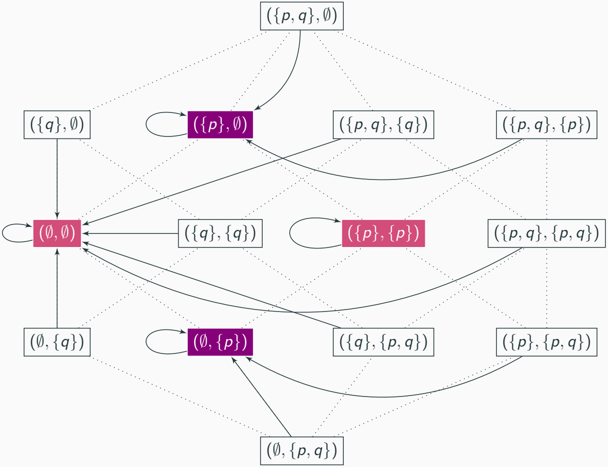
Partial Supported models



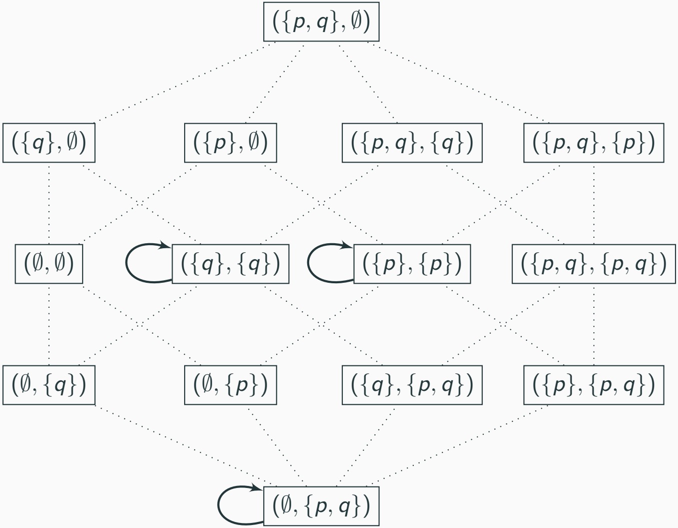
The Approximation Operator $\mathcal{IC}_{\mathcal{P}}$ (for $\mathcal{P} = \{p \leftarrow p, \neg q\}$)

$$\mathcal{IC}_{\mathcal{P}}(x, y) = (\mathcal{IC}^{l}_{\mathcal{P}}(x, y), \mathcal{IC}^{u}_{\mathcal{P}}(x, y))$$

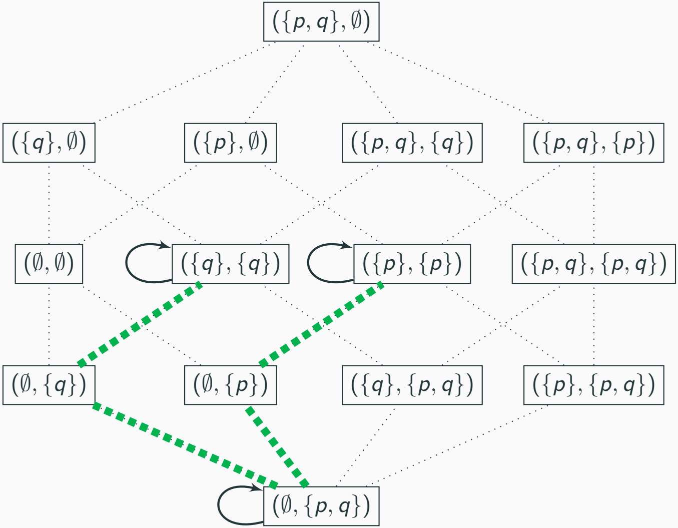
Supported models



Example: $\mathcal{P} = \{p \leftarrow \neg q; q \leftarrow \neg p\}$



Example: $\mathcal{P} = \{p \leftarrow \neg q; q \leftarrow \neg p\}$



Non-Deterministic Operators

Motivation

```
{processed(M, T, 1..A): availability(M,T,A)} C :-
 capacity(T,C).
processed(M, T, N-1) :- processed(M, T, N), N>1.
result(P,0,I) :- inventory(P,0,I), I>=0.
result(P,0,-I) :- backlog(P,0,I), I>0.
result(P,T,R1) :- demand(P,T,D), result(P,T-1,R), T>0,
S=#sum{Y,M,X : processed(M,T,X), yield(M,P,Y)}, R1=S-D+R.
```

Choice Atoms

A *choice atom* is an expression $C = (\mathsf{dom}, \mathsf{sat})$ where $\mathsf{dom} \subseteq \mathcal{A}$ and $\mathsf{sat} \subseteq \wp(\mathsf{dom})$.

Example

$1\{p, q, r\}2$ *corresponds to the choice atom* $C_1 = (\{p, q, r\}, \{\{p\}, \{q\}, \{r\}, \{p, q\}, \{p, r\}, \{q, r\}\})$.

Choice Atoms

A *choice atom* is an expression $C = (\mathsf{dom}, \mathsf{sat})$ where $\mathsf{dom} \subseteq \mathcal{A}$ and $\mathsf{sat} \subseteq \wp(\mathsf{dom})$.

A set of atoms x satisfies $(\mathsf{dom}, \mathsf{sat})$ if $x \cap \mathsf{dom} \in \mathsf{sat}$.

Example

$1\{p, q, r\}2$ *corresponds to the choice atom*
$C_1 = (\{p, q, r\}, \{\{p\}, \{q\}, \{r\}, \{p, q\}, \{p, r\}, \{q, r\}\})$.

Choice Atoms

A *choice atom* is an expression $C = (\mathsf{dom}, \mathsf{sat})$ where $\mathsf{dom} \subseteq \mathcal{A}$ and $\mathsf{sat} \subseteq \wp(\mathsf{dom})$.

A set of atoms x satisfies $(\mathsf{dom}, \mathsf{sat})$ if $x \cap \mathsf{dom} \in \mathsf{sat}$.

Example

$1\{p, q, r\}2$ *corresponds to the choice atom*
$C_1 = (\{p, q, r\}, \{\{p\}, \{q\}, \{r\}, \{p, q\}, \{p, r\}, \{q, r\}\})$.

- $\{p, q, s\}$ *satisfies* C_1 *as* $\{p, q, s\} \cap \{p, q, r\} = \{p, q\} \in \{\{p\}, \{q\}, \{r\}, \{p, q\}, \{p, r\}, \{q, r\}\}$.
- $\{p, q, r\}$ *does not satisfy* C_1 *as* $\{p, q, r\} \cap \{p, q, r\} = \{p, q, r\} \notin \{\{p\}, \{q\}, \{r\}, \{p, q\}, \{p, r\}, \{q, r\}\}$.

Choice Rules

Where C, $C_1, \ldots, C_n$ are choice atoms, a choice rule is of the form:

$$C \leftarrow C_1, \ldots, C_n.$$

Choice Rules

Where C, $C_1, \ldots, C_n$ are choice atoms, a choice rule is of the form:

$$C \leftarrow C_1, \ldots, C_n.$$

A program with choice rules is a set of choice rules.

Example
$\{1\{p, q\}2 \leftarrow \{p, q\} \neq 1\}$

Choice Rules

Where C, $C_1, \ldots, C_n$ are choice atoms, a choice rule is of the form:

$$C \leftarrow C_1, \ldots, C_n.$$

A program with choice rules is a set of choice rules.

Example
$\{1\{p, q\}2 \leftarrow \{p, q\} \neq 1\}$

- A choice rule is normal if $\mathrm{sat}(C_i) = \{\{a\}\}$ (for some $a \in \mathcal{A}$) or $\mathrm{sat}(C_i) = \{\emptyset\}$ for every $i = 1 \ldots n$,
  - $(\{a\}, \{\{a\}\})$ is denoted by a, and
  - $(\{a\}, \{\emptyset\})$ is denoted by $\neg a$.

Non-Deterministic Immediate Consequence Operator

Given a choice program $\mathcal{P}$ and a set of atoms x:

- A rule $r \in \mathcal{P}$ is x-applicable (in symbols, $r \in \mathcal{P}(x)$) if x satisfies the body of r.
- $IC_{\mathcal{P}}(x) = \{z \subseteq \bigcup_{r \in \mathcal{P}(x)} \mathrm{dom}(\mathrm{hd}(r)) \mid \forall r \in \mathcal{P}(x) : z \models \mathrm{hd}(r)\}$

Non-Deterministic Immediate Consequence Operator

Given a choice program $\mathcal{P}$ and a set of atoms x:

- A rule $r \in \mathcal{P}$ is x-applicable (in symbols, $r \in \mathcal{P}(x)$) if x satisfies the body of r.
- $IC_{\mathcal{P}}(x) = \{z \subseteq \bigcup_{r \in \mathcal{P}(x)} \mathrm{dom}(\mathrm{hd}(r)) \mid \forall r \in \mathcal{P}(x) : z \models \mathrm{hd}(r)\}$
  All ways to make the heads of x-applicable rules true.

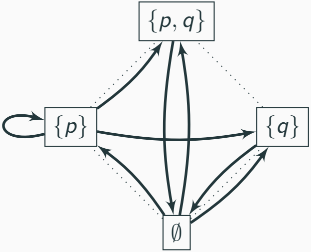
Non-Deterministic Immediate Consequence Operator

Given a choice program $\mathcal{P}$ and a set of atoms x:

- A rule $r \in \mathcal{P}$ is x-applicable (in symbols, $r \in \mathcal{P}(x)$) if x satisfies the body of r.
- $IC_{\mathcal{P}}(x) = \{z \subseteq \bigcup_{r \in \mathcal{P}(x)} \mathrm{dom}(\mathrm{hd}(r)) \mid \forall r \in \mathcal{P}(x) : z \models \mathrm{hd}(r)\}$
  All ways to make the heads of x-applicable rules true.

Example

$\mathcal{P} = \{1\{p, q\}2 \leftarrow \neg q\}$



Non-Deterministic Approximators for Normal Choice Programs

$$\mathcal{HD}^l_{\mathcal{P}}(x, y) = \{C \mid C \leftarrow a_1, \ldots, a_n, \neg b_1, \ldots, \neg b_m \in \mathcal{P},$$
$$\{a_1, \ldots, a_n\} \subseteq x, \{b_1, \ldots, b_m\} \cap y = \emptyset\}\}$$
$$\mathcal{IC}^l_{\mathcal{P}}(x, y) = \{z \subseteq \bigcup_{C \in \mathcal{HD}^l_{\mathcal{P}}(x,y)} \mathrm{dom}(C) \mid \forall C \in \mathcal{HD}^l_{\mathcal{P}}(x, y), z \cap \mathrm{dom}(C) \in \mathrm{sat}(C)\}$$
$$\mathcal{IC}^u_{\mathcal{P}}(x, y) = \mathcal{IC}^l_{\mathcal{P}}(y, x)$$
$$\mathcal{IC}_{\mathcal{P}}(x, y) = (\mathcal{IC}^l_{\mathcal{P}}(x, y), \mathcal{IC}^u_{\mathcal{P}}(x, y))$$

Non-Deterministic Approximators for Normal Choice Programs

$$\mathcal{HD}^l_{\mathcal{P}}(x, y) = \{C \mid C \leftarrow a_1, \ldots, a_n, \neg b_1, \ldots, \neg b_m \in \mathcal{P},$$
$$\{a_1, \ldots, a_n\} \subseteq x, \{b_1, \ldots, b_m\} \cap y = \emptyset\}\}$$

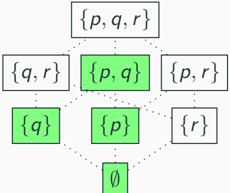
$$\mathcal{IC}^l_{\mathcal{P}}(x, y) = \{z \subseteq \bigcup_{C \in \mathcal{HD}^l_{\mathcal{P}}(x,y)} \mathrm{dom}(C) \mid \forall C \in \mathcal{HD}^l_{\mathcal{P}}(x, y), z \cap \mathrm{dom}(C) \in \mathrm{sat}(C)\}$$

$$\mathcal{IC}^u_{\mathcal{P}}(x, y) = \mathcal{IC}^l_{\mathcal{P}}(y, x)$$

$$\mathcal{IC}_{\mathcal{P}}(x, y) = (\mathcal{IC}^l_{\mathcal{P}}(x, y), \mathcal{IC}^u_{\mathcal{P}}(x, y))$$

Example: $\mathcal{P} = \{1\{p, q\}2 \leftarrow \neg r.\}$

- $\mathcal{HD}^{c,l}_{\mathcal{P}}(\emptyset, \{r\}) = \emptyset$,
- $HDc^{c,u}_{\mathcal{P}}(\emptyset, \{r\}) = \{1\{p, q\}2\}$,
- $\mathcal{IC}^c_{\mathcal{P}}(\emptyset, \{r\}) = (\{\emptyset\}, \{\{p\}, \{q\}, \{p, q\}\})$.



Approximators for Choice Programs (aka the real fun)

Given a choice program $\mathcal{P}$ and pair of sets of atoms (x, y) let:

$$\mathcal{HD}^{\mathsf{GZ},l}_{\mathcal{P}}(x,y) = \{C \mid \exists C \leftarrow C_1, \ldots, C_i \in \mathcal{P}, \forall i = 1 \ldots n : \\ x \cap \mathsf{dom}(C_i) = y \cap \mathsf{dom}(C_i) \in \mathsf{sat}(C_i)\},$$

$$\mathcal{HD}^{\mathsf{LPST},l}_{\mathcal{P}}(x,y) = \{C \mid \exists C \leftarrow C_1, \ldots, C_n \in \mathcal{P}, \forall i = 1 \ldots n : \\ \forall z \in [x,y] : z(C_i) = \mathsf{T}\},$$

$$\mathcal{HD}^{\mathsf{MR},l}_{\mathcal{P}}(x,y) = \{C \mid \exists C \leftarrow C_1, \ldots, C_n \in \mathcal{P}, \exists z \subseteq x : \\ \forall i = 1 \ldots n : y(C_i) = \mathsf{T} \text{ and } z(C_i) = \mathsf{T}\},$$

For $\mathsf{x} \in \{\mathsf{LPST}, \mathsf{MR}, \mathsf{GZ}\})$ let:

$$\mathcal{IC}^{\mathsf{x},l}(x,y) = \{z \subseteq \bigcup_{C \in \mathcal{HD}^{\mathsf{x},l}_{\mathcal{P}}(x,y)} \mathsf{dom}(C) \mid \forall C \in \mathcal{HD}^{\mathsf{x},l}_{\mathcal{P}}(x,y) : z \cap \mathsf{dom}(C) \in \mathsf{sat}(C)\}$$

For $\dagger \in \{\mathsf{MR}, \mathsf{LPST}, \mathcal{U}\}$ let:

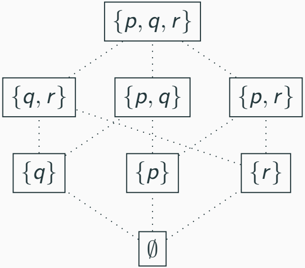
$$\mathcal{IC}^{\mathcal{U},l}_{\mathcal{P}}(x,y) = \mathcal{IC}^{\dagger,u}_{\mathcal{P}}(x,y) \qquad = \bigcup_{x \subseteq z \subseteq y} IC_{\mathcal{P}}(z)$$

whereas $\mathcal{IC}^{\mathsf{GZ},u}_{\mathcal{P}}(x,y) = \mathcal{IC}^{\mathsf{GZ},l}_{\mathcal{P}}(x,y)$. $\mathcal{IC}^{\mathsf{x}}(x,y) = (\mathcal{IC}^{\mathsf{x},l}(x,y), \mathcal{IC}^{\mathsf{x},u}(x,y))$ (for $\mathsf{x} \in \{\mathsf{LPST}, \mathsf{MR}, \mathsf{GZ}, \mathcal{U}\}$).

Intermezzo on Orders

Let $L = \langle \mathcal{L}, \leq \rangle$ be a lattice and $X, Y \in 2^{\mathcal{L}}$.

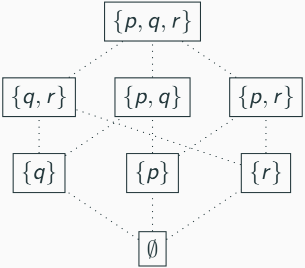
- $X_1 \preceq_L^S X_2$ if for every $x_2 \in X_2$ there is an $x_1 \in X$ s.t. $x_1 \leq x_2$.
- $Y_1 \preceq_L^H Y_2$ if for every $y_1 \in Y_1$ there is a $y_2 \in Y_2$ s.t. $y_1 \leq y_2$.
- $(X_1,\ Y_1) \preceq_i^A (X_2,\ Y_2)$ iff $X_1 \preceq_L^S X_2$ and $Y_2 \preceq_L^H Y_1$.



Intermezzo on Orders

Let $L = \langle \mathcal{L}, \leq \rangle$ be a lattice and $X, Y \in 2^{\mathcal{L}}$.

- $X_1 \preceq_L^S X_2$ if for every $x_2 \in X_2$ there is an $x_1 \in X$ s.t. $x_1 \leq x_2$.
- $Y_1 \preceq_L^H Y_2$ if for every $y_1 \in Y_1$ there is a $y_2 \in Y_2$ s.t. $y_1 \leq y_2$.
- $(X_1,\ Y_1) \preceq_i^A (X_2,\ Y_2)$ iff $X_1 \preceq_L^S X_2$ and $Y_2 \preceq_L^H Y_1$.

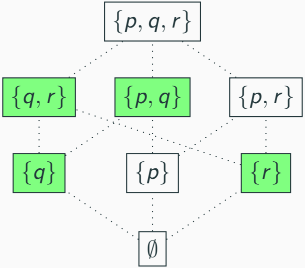


Pairs of sets (X, Y) as convex sets.

Intermezzo on Orders

Let $L = \langle \mathcal{L}, \leq \rangle$ be a lattice and $X, Y \in 2^{\mathcal{L}}$.

- $X_1 \preceq_L^S X_2$ if for every $x_2 \in X_2$ there is an $x_1 \in X$ s.t. $x_1 \leq x_2$.
- $Y_1 \preceq_L^H Y_2$ if for every $y_1 \in Y_1$ there is a $y_2 \in Y_2$ s.t. $y_1 \leq y_2$.
- $(X_1,\ Y_1) \preceq_i^A (X_2,\ Y_2)$ iff $X_1 \preceq_L^S X_2$ and $Y_2 \preceq_L^H Y_1$.

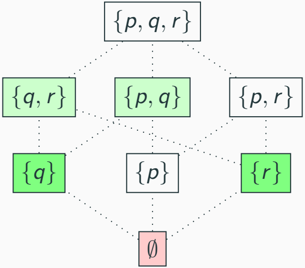


Pairs of sets (X, Y) as convex sets.

Intermezzo on Orders

Let $L = \langle \mathcal{L}, \leq \rangle$ be a lattice and $X, Y \in 2^{\mathcal{L}}$.

- $X_1 \preceq_L^S X_2$ if for every $x_2 \in X_2$ there is an $x_1 \in X$ s.t. $x_1 \leq x_2$.
- $Y_1 \preceq_L^H Y_2$ if for every $y_1 \in Y_1$ there is a $y_2 \in Y_2$ s.t. $y_1 \leq y_2$.
- $(X_1,\ Y_1) \preceq_i^A (X_2,\ Y_2)$ iff $X_1 \preceq_L^S X_2$ and $Y_2 \preceq_L^H Y_1$.

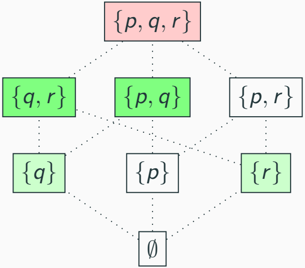


$$\{\emptyset\} \preceq_L^S \{\{q\}, \{r\}\}$$

Intermezzo on Orders

Let $L = \langle \mathcal{L}, \leq \rangle$ be a lattice and $X, Y \in 2^{\mathcal{L}}$.

- $X_1 \preceq_L^S X_2$ if for every $x_2 \in X_2$ there is an $x_1 \in X$ s.t. $x_1 \leq x_2$.
- $Y_1 \preceq_L^H Y_2$ if for every $y_1 \in Y_1$ there is a $y_2 \in Y_2$ s.t. $y_1 \leq y_2$.
- $(X_1,\ Y_1) \preceq_i^A (X_2,\ Y_2)$ iff $X_1 \preceq_L^S X_2$ and $Y_2 \preceq_L^H Y_1$.

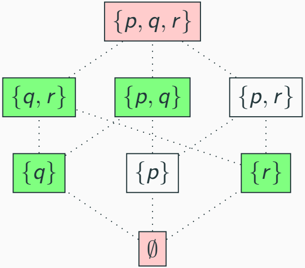


$$\{\{q, r\}, \{p, q\}\} \preceq_L^H \{\{p, q, r\}\}$$

Intermezzo on Orders

Let $L = \langle \mathcal{L}, \leq \rangle$ be a lattice and $X, Y \in 2^{\mathcal{L}}$.

- $X_1 \preceq_L^S X_2$ if for every $x_2 \in X_2$ there is an $x_1 \in X$ s.t. $x_1 \leq x_2$.
- $Y_1 \preceq_L^H Y_2$ if for every $y_1 \in Y_1$ there is a $y_2 \in Y_2$ s.t. $y_1 \leq y_2$.
- $(X_1,\ Y_1) \preceq_i^A (X_2,\ Y_2)$ iff $X_1 \preceq_L^S X_2$ and $Y_2 \preceq_L^H Y_1$.



$$(\{\emptyset\},\ \{\{p, q, r\}\}) \preceq_i^A (\{\{q\}, \{r\}\},\ \{\{q, r\}, \{p, q\}\})$$

Non-Deterministic Approximation Operators

O is approximated using a non-deterministic approximation operator $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$:

- that is $\preceq_i^A$-monotonic,
- $\mathcal{O}(x, x) = (O(x),\ O(x))$ for every $x \in \mathcal{L}$.

Non-Deterministic Approximation Operators

O is approximated using a non-deterministic approximation operator $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$:

- that is $\preceq_i^A$-monotonic,
- $\mathcal{O}(x, x) = (O(x),\ O(x))$ for every $x \in \mathcal{L}$.

Output of $\mathcal{O}(x, y) = (\{x_1, x_2, \ldots\},\ \{y_1, y_2, \ldots\})$ is a set of lower bounds respectively upper bounds on choices $O(z) = \{z_1, z_2, \ldots\}$ (with $x \leq z \leq y$).

Non-Deterministic Approximation Operators

O is approximated using a non-deterministic approximation operator $\mathcal{O} : \mathcal{L}^2 \rightarrow 2^{\mathcal{L}} \times 2^{\mathcal{L}}$:

- that is $\preceq_i^A$-monotonic,
- $\mathcal{O}(x, x) = (O(x),\ O(x))$ for every $x \in \mathcal{L}$.

Output of $\mathcal{O}(x, y) = (\{x_1, x_2, \ldots\},\ \{y_1, y_2, \ldots\})$ is a set of lower bounds respectively upper bounds on choices $O(z) = \{z_1, z_2, \ldots\}$ (with $x \leq z \leq y$).

Non-Deteterministic AFT: fixpoints

$\mathcal{IC}_{\mathcal{P}}$ allows to generalize supported models to the three-valued case:

Example $\mathcal{P} = \{\{p, q\} = 1 \leftarrow \neg p. \quad p \leftarrow q.\}$

- No (two-valued) supported models

Non-Deteterministic AFT: fixpoints

$\mathcal{IC}_{\mathcal{P}}$ allows to generalize supported models to the three-valued case:

Example $\mathcal{P} = \{\{p, q\} = 1 \leftarrow \neg p. \quad p \leftarrow q.\}$

- No (two-valued) supported models
- $\mathcal{IC}^c_{\mathcal{P}}(\emptyset, \{p\}) = (\{\emptyset\}, \{\{p\}, \{q\}, \{p, q\}\})$.
- (among others)

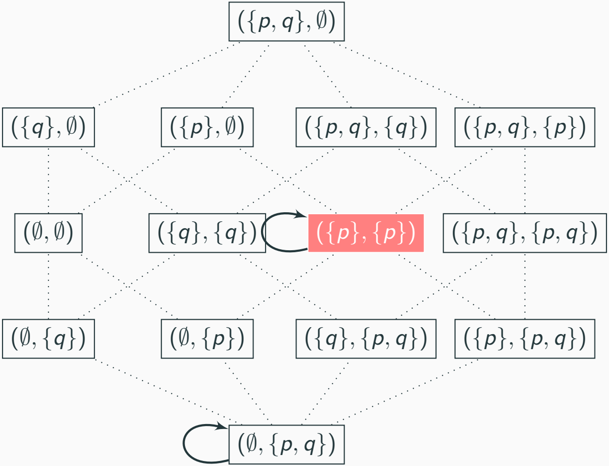
Proposition
Let a normal logic program $\mathcal{P}$ be given. Then $(x, y) \in \mathcal{IC}_{\mathcal{P}}(x, y)$ iff (x, y) is a partial supported model of $\mathcal{P}$.

Proposition
Let a choice program $\mathcal{P}$ and $\mathrm{x} \in \{\mathrm{LPST}, \mathrm{MR}, \mathrm{GZ}, \mathcal{U}\}$ be given. If $(x, y) \in \mathcal{IC}^{\mathrm{x}}_{\mathcal{P}}(x, y)$ then for every $a \in y$, there is a $C \leftarrow C_1, \ldots, C_n$ with $a \in \mathrm{dom}(C)$ s.t.
$\mathcal{IC}^{\mathrm{x}}_{\{p \leftarrow C_1, \ldots, C_n\}}(x, y) = (\{\{p\} \cap x\}, \{\{p\} \cap y\})$.

Stable Semantics

Example: Kripke-Kleene is rather weak

$$p \leftarrow \neg q;\ q \leftarrow q$$



Example: Kripke-Kleene is rather weak

$$\mathcal{P} = \{p \leftarrow \neg q;\ q \leftarrow q\}$$

Construction of the Kripke-Kleene fixpoint:

- $\mathcal{IC}_{\mathcal{P}}(\emptyset, \{p, q\}) = (\emptyset, \{p, q\})$.
- Fixpoint reached.

Can't get rid of the self-supporting atom q in the upper bound.

Example: Kripke-Kleene is rather weak

$$\mathcal{P} = \{p \leftarrow \neg q;\ q \leftarrow q\}$$

Construction of the Kripke-Kleene fixpoint:

- $\mathcal{IC}_{\mathcal{P}}(\emptyset, \{p, q\}) = (\emptyset, \{p, q\})$.
- Fixpoint reached.

Can't get rid of the self-supporting atom q in the upper bound.

Assuming that no atom is certainly true, construct the smallest upper bound possible:

$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \emptyset) = \{p\}$

$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \{p\}) = \{p\}$

Example: Kripke-Kleene is rather weak

$$\mathcal{P} = \{p \leftarrow \neg q;\ q \leftarrow q\}$$

Construction of the Kripke-Kleene fixpoint:

- $\mathcal{IC}_{\mathcal{P}}(\emptyset, \{p, q\}) = (\emptyset, \{p, q\})$.
- Fixpoint reached.

Can't get rid of the self-supporting atom q in the upper bound.

Assuming that no atom is certainly true, construct the smallest upper bound possible:

$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \emptyset) = \{p\}$

$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \{p\}) = \{p\}$

As $\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \cdot)$ is a $\subseteq$-monotonic operator, it admits a least fixed point.

Stable Operator

$$S(\mathcal{IC}^{l}_{\mathcal{P}})(y) = lfp(\mathcal{IC}^{l}_{\mathcal{P}}(\cdot, y))$$

Stable Operator

$$S(\mathcal{IC}^{l}_{\mathcal{P}})(y) = lfp(\mathcal{IC}^{l}_{\mathcal{P}}(\cdot, y))$$

Example ($\{p \leftarrow \neg q;\ q \leftarrow q\}$)

$S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p, q\}) = \emptyset$ *since:*

$\mathcal{IC}^{l}_{\mathcal{P}}(\emptyset, \{p, q\}) = \emptyset$*: fixpoint reached.*

Stable Operator

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y))$$

$$S(\mathcal{IC}^u_{\mathcal{P}})(x) = lfp(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot)) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, x))$$

Example ($\{p \leftarrow \neg q;\ q \leftarrow q\}$)

$S(\mathcal{IC}^l_{\mathcal{P}})(\{p, q\}) = \emptyset$ *since:*

$\mathcal{IC}^l_{\mathcal{P}}(\emptyset, \{p, q\}) = \emptyset$: *fixpoint reached.*

Stable Operator

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y))$$

$$S(\mathcal{IC}^u_{\mathcal{P}})(x) = lfp(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot)) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, x))$$

Example ($\{p \leftarrow \neg q;\ q \leftarrow q\}$)

$S(\mathcal{IC}^l_{\mathcal{P}})(\{p, q\}) = \emptyset$ *since:*
$\mathcal{IC}^l_{\mathcal{P}}(\emptyset, \{p, q\}) = \emptyset$: *fixpoint reached.*

$S(\mathcal{IC}^u_{\mathcal{P}})(\emptyset) = \{p\}$ *since:*
$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \emptyset) = \{p\}$
$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \{p\}) = \{p\}$: *fixpoint reached.*

Stable Operator

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y))$$

$$S(\mathcal{IC}^u_{\mathcal{P}})(x) = lfp(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot)) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, x))$$

$$S(\mathcal{IC}_{\mathcal{P}})(x, y) = (S(\mathcal{IC}^l_{\mathcal{P}})(y), \; S(\mathcal{IC}^u_{\mathcal{P}})(x))$$

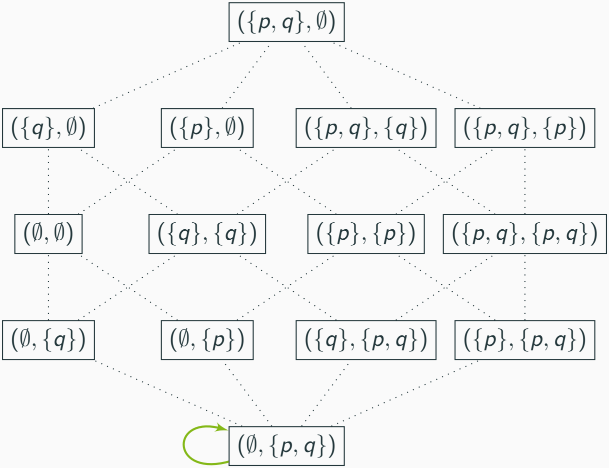
Example ($\{p \leftarrow \neg q; \; q \leftarrow q\}$)

$S(\mathcal{IC}^l_{\mathcal{P}})(\{p, q\}) = \emptyset$ *since:*
$\mathcal{IC}^l_{\mathcal{P}}(\emptyset, \{p, q\}) = \emptyset$: *fixpoint reached.*

$S(\mathcal{IC}^u_{\mathcal{P}})(\emptyset) = \{p\}$ *since:*
$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \emptyset) = \{p\}$
$\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \{p\}) = \{p\}$: *fixpoint reached.*

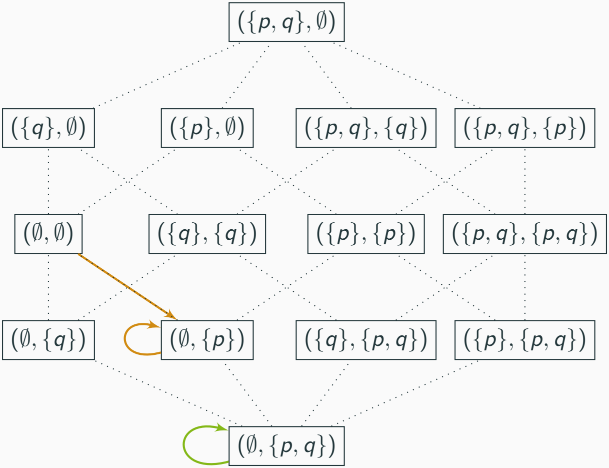
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$S(\mathcal{IC}^l_{\mathcal{P}})(\{p, q\}) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, \{p, q\}))$



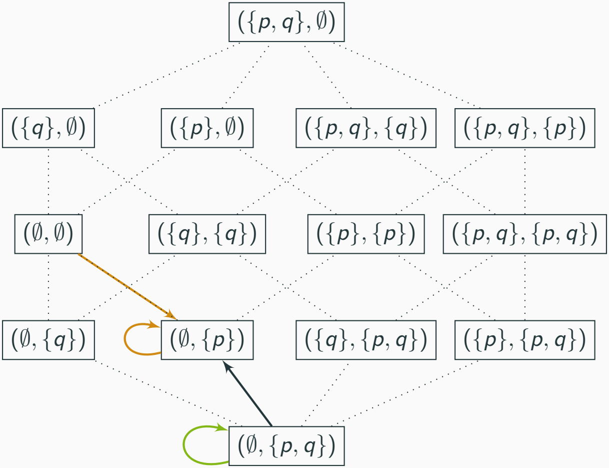
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$S(\mathcal{IC}^u_{\mathcal{P}})(\emptyset) = lfp(\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \cdot))$



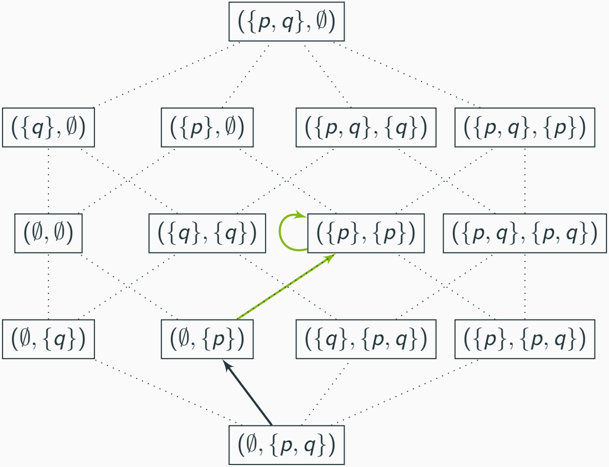
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$$S(\mathcal{IC}_{\mathcal{P}})(\emptyset, \{p, q\}) = (S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p, q\}),\ S(\mathcal{IC}^{u}_{\mathcal{P}})((\emptyset))$$



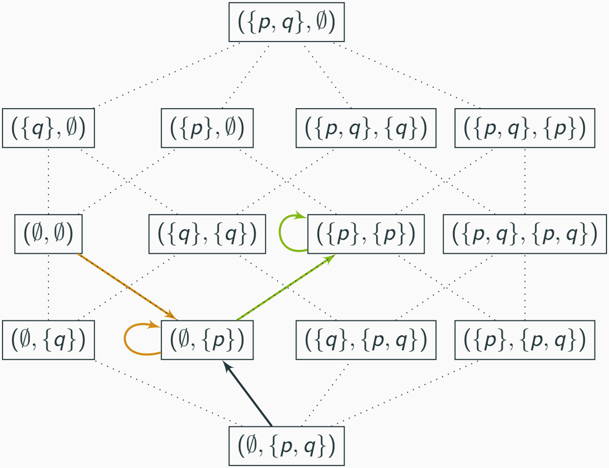
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$S(\mathcal{IC}^l_{\mathcal{P}})(\{p\}) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, \{p\}))$



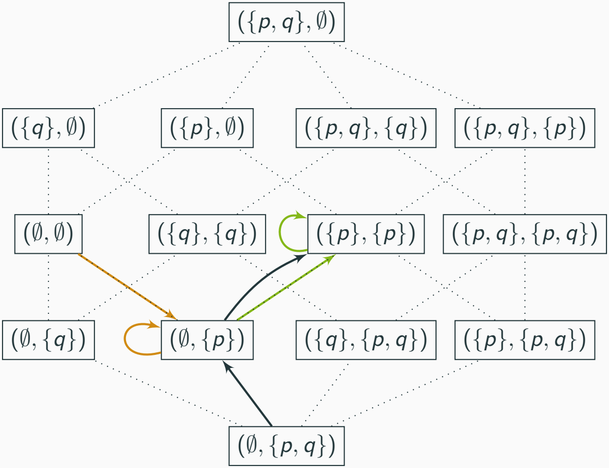
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$S(\mathcal{IC}^u_{\mathcal{P}})(\emptyset) = lfp(\mathcal{IC}^u_{\mathcal{P}}(\emptyset, \cdot))$



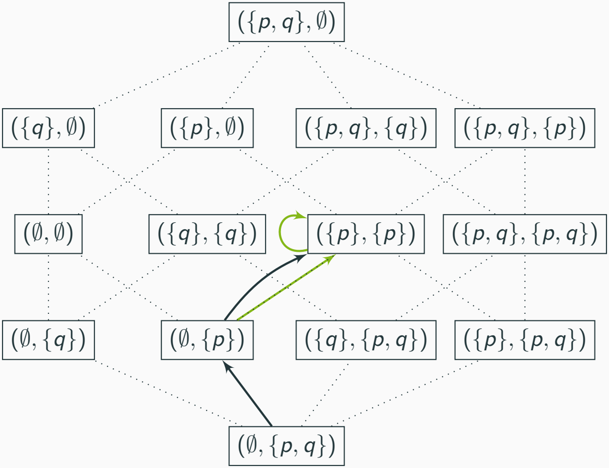
Stable Operator: Example $\{p \leftarrow \neg q;\, q \leftarrow q\}$

$$S(\mathcal{IC}_{\mathcal{P}})(\emptyset, \{p\}) = (S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p\}),\ S(\mathcal{IC}^{u}_{\mathcal{P}})(\emptyset))$$



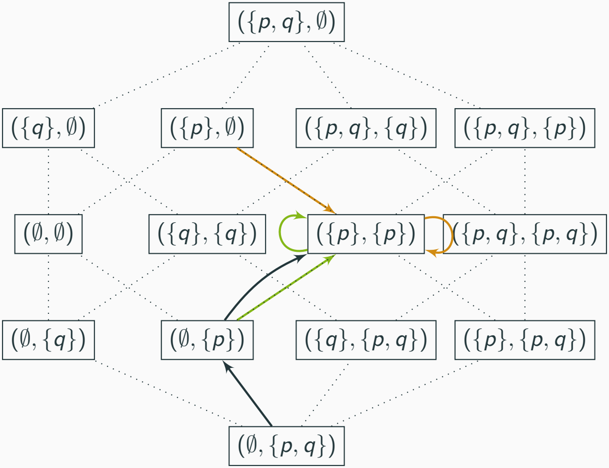
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$$S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p\}) = lfp(\mathcal{IC}^{l}_{\mathcal{P}}(\cdot, \{p\}))$$



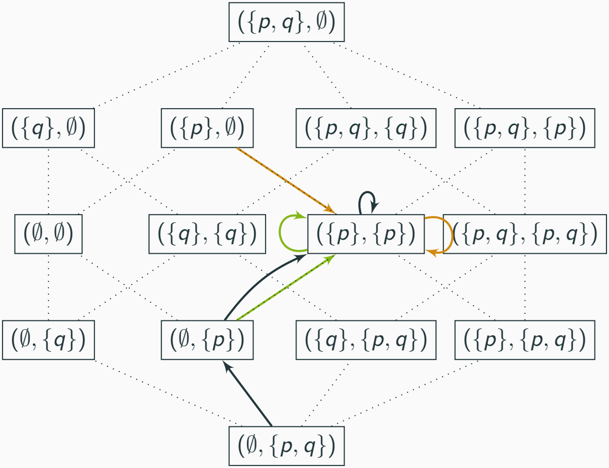
Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$S(\mathcal{IC}^u_{\mathcal{P}})(x) = lfp(\mathcal{IC}^u_{\mathcal{P}}(\{p\}, \cdot))$



Stable Operator: Example $\{p \leftarrow \neg q; q \leftarrow q\}$

$$S(\mathcal{IC_P})(\{p\}, \{p\}) = (S(\mathcal{IC}^l_\mathcal{P})(\{p\}),\ S(\mathcal{IC}^u_\mathcal{P})(\{p\}))$$



Stable Operator and Well-Founded Model

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = \mathit{lfp}(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y)) \qquad S(\mathcal{IC}^u_{\mathcal{P}})(x) = \mathit{lfp}(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot))$$

$$S(\mathcal{IC}_{\mathcal{P}})(x, y) = (S(\mathcal{IC}^l_{\mathcal{P}})(y),\ S(\mathcal{IC}^u_{\mathcal{P}})(x))$$

- $S(\mathcal{IC}_{\mathcal{P}})$ is a $\leq_i$-monotonic operator, so it admits a least fixpoint.

We call this the well-founded model, denoted $\mathrm{WF}(\mathcal{P})$.

Stable Operator and Well-Founded Model

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y)) \qquad S(\mathcal{IC}^u_{\mathcal{P}})(x) = lfp(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot))$$

$$S(\mathcal{IC}_{\mathcal{P}})(x, y) = (S(\mathcal{IC}^l_{\mathcal{P}})(y),\ S(\mathcal{IC}^u_{\mathcal{P}})(x))$$

- $S(\mathcal{IC}_{\mathcal{P}})$ is a $\leq_i$-monotonic operator, so it admits a least fixpoint. We call this the well-founded model, denoted $\mathrm{WF}(\mathcal{P})$.
- The well-founded model is more precise than the Kripke-Kleene fixpoint: $\mathrm{KK}(\mathcal{P}) \leq_i \mathrm{WF}(\mathcal{P})$.

Stable Operator and Well-Founded Model

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = lfp(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y)) \qquad S(\mathcal{IC}^u_{\mathcal{P}})(x) = lfp(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot))$$

$$S(\mathcal{IC}_{\mathcal{P}})(x, y) = (S(\mathcal{IC}^l_{\mathcal{P}})(y),\ S(\mathcal{IC}^u_{\mathcal{P}})(x))$$

• $S(\mathcal{IC}_{\mathcal{P}})$ is a $\leq_i$-monotonic operator, so it admits a least fixpoint. We call this the well-founded model, denoted $\mathrm{WF}(\mathcal{P})$.

• The well-founded model is more precise than the Kripke-Kleene fixpoint: $\mathrm{KK}(\mathcal{P}) \leq_i \mathrm{WF}(\mathcal{P})$.

• Any fixpoint of $S(\mathcal{IC}_{\mathcal{P}})$ is a minimal model of $\mathcal{P}$.
If $(x, y) = S(\mathcal{IC}_{\mathcal{P}})(x, y)$, we call it a (partial) stable model.
If $x = S(\mathcal{IC}_{\mathcal{P}})(x)$, we call it a stable model.

Stable Operator and Well-Founded Model

$$S(\mathcal{IC}^l_{\mathcal{P}})(y) = \mathit{lfp}(\mathcal{IC}^l_{\mathcal{P}}(\cdot, y)) \qquad S(\mathcal{IC}^u_{\mathcal{P}})(x) = \mathit{lfp}(\mathcal{IC}^u_{\mathcal{P}}(x, \cdot))$$

$$S(\mathcal{IC}_{\mathcal{P}})(x, y) = (S(\mathcal{IC}^l_{\mathcal{P}})(y),\ S(\mathcal{IC}^u_{\mathcal{P}})(x))$$

• $S(\mathcal{IC}_{\mathcal{P}})$ is a $\leq_i$-monotonic operator, so it admits a least fixpoint.
We call this the well-founded model, denoted $\mathrm{WF}(\mathcal{P})$.

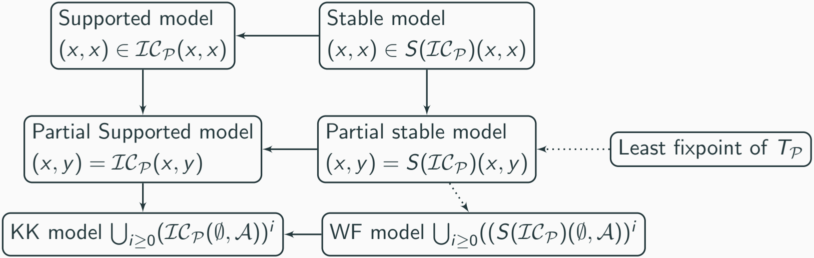
• The well-founded model is more precise than the Kripke-Kleene fixpoint: $\mathrm{KK}(\mathcal{P}) \leq_i \mathrm{WF}(\mathcal{P})$.

• Any fixpoint of $S(\mathcal{IC}_{\mathcal{P}})$ is a minimal model of $\mathcal{P}$.
If $(x, y) = S(\mathcal{IC}_{\mathcal{P}})(x, y)$, we call it a (partial) stable model.
If $x = S(\mathcal{IC}_{\mathcal{P}})(x)$, we call it a stable model.

• If $T_{\mathcal{P}}$ has a least fixpoint, it coincides with the well-founded model.

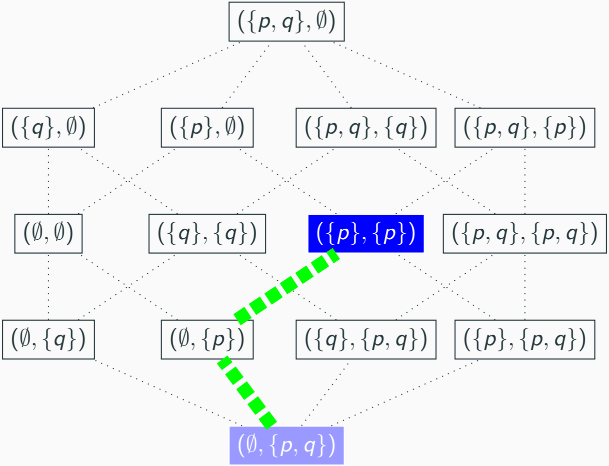
Taking Stock

Semantics =	Stable? +	Operator +	Fixpoint
	S	$(\mathcal{O})(x,y)$	$=(x,y)$
Kripke-Kleene	No	$\mathcal{IC}_{\mathcal{P}}$	least fp
Partial Supported	No	$\mathcal{IC}_{\mathcal{P}}$	any fp
Supported	No	$\mathcal{IC}_{\mathcal{P}}$	total fp $(x=y)$
Well-founded	yes	$\mathcal{IC}_{\mathcal{P}}$	least fp
Partial Stable	yes	$\mathcal{IC}_{\mathcal{P}}$	any fp
Stable	yes	$\mathcal{IC}_{\mathcal{P}}$	total fp $(x=y)$



Stable Operator: Example

$p \leftarrow \neg q; q \leftarrow q$



Stable Operator: Example 2

$$\mathcal{P} = \{p \leftarrow \neg q; \quad q \leftarrow \neg p; \quad r \leftarrow r; \quad s \leftarrow \neg r\}$$

- Kripke-Kleene fixpoint: $(\emptyset, \{p, q, r, s\})$.
- Well-founded model: $(\{s\}, \{p, q, s\})$.
- Stable models: $(\{p, s\}, \{p, s\})$, $(\{q, s\}, \{q, s\})$.

Stable Semantics and Reducts

$$\frac{\mathcal{P}}{x} = \{a \leftarrow b_1, \ldots, b_n \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}$$

$$c_1, \ldots, c_n \notin x\}$$

Definition

x is a stable model of $\mathcal{P}$ if it is a minimal model of $\frac{\mathcal{P}}{x}$.

Stable Semantics and Reducts

$$\frac{\mathcal{P}}{x} = \{a \leftarrow b_1, \ldots, b_n \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}$$

$$c_1, \ldots, c_n \notin x\}$$

Definition
x is a stable model of $\mathcal{P}$ if it is a minimal model of $\frac{\mathcal{P}}{x}$.

Example ($\mathcal{P} = \{p \leftarrow \neg p; q \leftarrow \neg p; p \leftarrow \neg q\}$)
$\frac{\mathcal{P}}{\{q\}} = \{p \leftarrow; q \leftarrow\}$. *$\{q\}$ is not a minimal model of $\mathcal{P}$, thus $\{q\}$ is not a stable model.*

$\frac{\mathcal{P}}{\{p\}} = \{p \leftarrow\}$. *$\{p\}$ is a minimal model of $\mathcal{P}$.$\{q\}$ is not a minimal model of $\mathcal{P}$, thus $\{p\}$ is a stable model.*

Stable Semantics and Reducts

$$\frac{\mathcal{P}}{x} = \{a \leftarrow b_1, \ldots, b_n \mid a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \in \mathcal{P}$$

$$c_1, \ldots, c_n \notin x\}$$

Definition
x is a stable model of $\mathcal{P}$ if it is a minimal model of $\frac{\mathcal{P}}{x}$.

Proposition
$S(\mathcal{IC}^l_{\mathcal{P}})(y)$ is the set of minimal models of $\frac{\mathcal{P}}{y}$.

Proposition
$(x, x) = S(\mathcal{IC}_{\mathcal{P}})(x, x)$ if and only if x is a stable model of $\mathcal{P}$ (iff $x = S(\mathcal{IC}^l_{\mathcal{P}})(x)$).

Stable Operator for Deterministic Approximation Operators

Basic idea: look for smallest fixpoint that the upper bound allows us to construct:

$$S(\mathcal{O}_l)(y) = glb\{x \in \mathcal{L} \mid x = \mathcal{O}_l(x, y)\}$$

Stable Operator for Deterministic Approximation Operators

Basic idea: look for smallest fixpoint that the upper bound allows us to construct:

$$S(\mathcal{O}_I)(y) = glb\{x \in \mathcal{L} \mid x = \mathcal{O}_I(x, y)\} = \bigcup_{i=1}^{\infty} \mathcal{O}_I^i(\emptyset, y).$$

Stable Operator for Deterministic Approximation Operators

Basic idea: look for smallest fixpoint that the upper bound allows us to construct:

$$S(\mathcal{O}_I)(y) = glb\{x \in \mathcal{L} \mid x = \mathcal{O}_I(x, y)\} = \bigcup_{i=1}^{\infty} \mathcal{O}_I^i(\emptyset, y).$$

Fix what is false and accept only what is absolutely necessary in view of that.

Stable Operator for Deterministic Approximation Operators

Basic idea: look for smallest fixpoint that the upper bound allows us to construct:

$$S(\mathcal{O}_I)(y) = glb\{x \in \mathcal{L} \mid x = \mathcal{O}_I(x,y)\} = \bigcup_{i=1}^{\infty} \mathcal{O}_I^i(\emptyset, y).$$

Fix what is false and accept only what is absolutely necessary in view of that.

(for finite lattices this is just the $\leq$-minimal fixpoint of $\mathcal{O}_I(.,y)$)

Stable Operator for Deterministic Approximation Operators

Basic idea: look for smallest fixpoint that the upper bound allows us to construct:

$$S(\mathcal{O}_I)(y) = glb\{x \in \mathcal{L} \mid x = \mathcal{O}_I(x, y)\} = \bigcup_{i=1}^{\infty} \mathcal{O}_I^i(\emptyset, y).$$

Fix what is false and accept only what is absolutely necessary in view of that.
(for finite lattices this is just the $\leq$-minimal fixpoint of $\mathcal{O}_I(., y)$)

Stable Operator: Example

$$\mathcal{P} = \{1\{p, q\}2 \leftarrow .\}$$

x	$\emptyset$	$\{p\}$	$\{q\}$	$\{p, q\}$
$\mathcal{IC}^l_{\mathcal{P}}(x, y)$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{p\}, \{p, q\}$	$\{p\}, \{q\}, \{p, q\}$

1. glb of fixpoints of $\mathcal{IC}^{c,l}_{\mathcal{P}}(., y)$: $\{p\} \cap \{q\} \cap \{p, q\} = \emptyset$.
2. Minimal fixpoints: $\{p\}$ and $\{q\}$.
3. $\bigcup_{i=1}^{\infty} (\mathcal{IC}^l_{\mathcal{P}})^i(., y) = ?$

Stable Operator: Example

$$\mathcal{P} = \{1\{p, q\}2 \leftarrow .\}$$

x	$\emptyset$	$\{p\}$	$\{q\}$	$\{p, q\}$
$\mathcal{IC}^{l}_{\mathcal{P}}(x, y)$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{p\}, \{p, q\}$	$\{p\}, \{q\}, \{p, q\}$

1. glb of fixpoints of $\mathcal{IC}^{c,l}_{\mathcal{P}}(., y)$: $\{p\} \cap \{q\} \cap \{p, q\} = \emptyset$.
2. Minimal fixpoints: $\{p\}$ and $\{q\}$.
3. $\bigcup_{i=1}^{\infty}(\mathcal{IC}^{l}_{\mathcal{P}})^{i}(., y) = ?$

We have two choices now: minimality based or iterative.

Minimality-Based Stable Operator

Let an ndao $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ and some $x, y \in \mathcal{L}$ be given. Then:

- the m-complete lower stable operator as:

$C^m(\mathcal{O}_l)(y) = \{x \in \mathcal{L} \mid x \in \mathcal{O}_l(x, y) \text{ and } \neg\exists x' < y : x' \in \mathcal{O}_l(x', y)\}$

- the *m-complete upper stable operator* as:

$C^m(\mathcal{O}_u)(x) = \{y \in \mathcal{L} \mid y \in \mathcal{O}_u(x, y) \text{ and } \neg\exists y' < y : z \in \mathcal{O}_u(x, y')\}$

- the m-stable operator as $S^m(\mathcal{O})(x, y) = (C(\mathcal{O}_l)(y), C(\mathcal{O}_u)(x))$.
- a m-stable fixpoint of $\mathcal{O}$ as any $(x, y) \in \mathcal{L}^2$ s.t.

$(x, y) \in S^m(\mathcal{O})(x, y)$.

Minimality-Based Stable Operator

Let an ndao $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ and some $x, y \in \mathcal{L}$ be given. Then:

- the m-complete lower stable operator as:

$C^m(\mathcal{O}_l)(y) = \{x \in \mathcal{L} \mid x \in \mathcal{O}_l(x, y) \text{ and } \neg\exists x' < y : x' \in \mathcal{O}_l(x', y)\}$

- the *m-complete upper stable operator* as:

$C^m(\mathcal{O}_u)(x) = \{y \in \mathcal{L} \mid y \in \mathcal{O}_u(x, y) \text{ and } \neg\exists y' < y : z \in \mathcal{O}_u(x, y')\}$

- the m-stable operator as $S^m(\mathcal{O})(x, y) = (C(\mathcal{O}_l)(y), C(\mathcal{O}_u)(x))$.
- a m-stable fixpoint of $\mathcal{O}$ as any $(x, y) \in \mathcal{L}^2$ s.t.

$(x, y) \in S^m(\mathcal{O})(x, y)$.

Example ($\mathcal{P} = \{1\{p, q\} \leftarrow \neg q\}$)
$C^m(\mathcal{IC}^l_{\mathcal{P}})(\{p\}) \min_{\subseteq}\{x \subseteq \mathcal{A}_{\mathcal{P}} \mid x \in \mathcal{IC}_{\mathcal{P}}(x, \{p\})\} = \{\{p\}, \{q\}\}$.
$(\{p\}, \{p\})$ *is a m-stable fixpoint.*

Minimality-Based Stable Operator

Let an ndao $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ and some $x, y \in \mathcal{L}$ be given. Then:

- the m-complete lower stable operator as:

$C^m(\mathcal{O}_l)(y) = \{x \in \mathcal{L} \mid x \in \mathcal{O}_l(x, y) \text{ and } \neg\exists x' < y : x' \in \mathcal{O}_l(x', y)\}$

- the *m-complete upper stable operator* as:

$C^m(\mathcal{O}_u)(x) = \{y \in \mathcal{L} \mid y \in \mathcal{O}_u(x, y) \text{ and } \neg\exists y' < y : z \in \mathcal{O}_u(x, y')\}$

- the m-stable operator as $S^m(\mathcal{O})(x, y) = (C(\mathcal{O}_l)(y), C(\mathcal{O}_u)(x))$.
- a m-stable fixpoint of $\mathcal{O}$ as any $(x, y) \in \mathcal{L}^2$ s.t.

$(x, y) \in S^m(\mathcal{O})(x, y)$.

Example ($\mathcal{P} = \{1\{p, q\} \leftarrow \neg q\}$)
$C^m(\mathcal{IC}^l_{\mathcal{P}})(\{p\}) \min_{\subseteq}\{x \subseteq \mathcal{A}_{\mathcal{P}} \mid x \in \mathcal{IC}_{\mathcal{P}}(x, \{p\})\} = \{\{p\}, \{q\}\}$.
$(\{p\}, \{p\})$ *is a m-stable fixpoint.*

Theorem Let a dlp $\mathcal{P}$ and a consistent $x \subseteq y \subseteq \mathcal{A}^2_{\mathcal{P}}$ be given. Then (x, y) is a stable model of $\mathcal{P}$ according to [Prz91] (i.e. the minimal models of the reduct) iff $(x, y) \in S(\mathcal{IC}_{\mathcal{P}})(x, y)$.

Constructive Stable Operator

Given a non-deterministic operator $O : \mathcal{L} \to \wp(\mathcal{L})$, a sequence $x_0, \ldots, x_n \subseteq \mathcal{L}$ is well-founded relative to O if:

- $x_0 = \perp$,
- $x_i \leq x_{i+1}$ and $x_{i+1} \in O(x_i)$ for every successor ordinal $i \geq 0$.
- $x_\lambda = (lub\{x_i\}_{i<\lambda})$ for a limit ordinal λ.

Constructive Stable Operator

Given a non-deterministic operator $O : \mathcal{L} \to \wp(\mathcal{L})$, a sequence $x_0, \ldots, x_n \subseteq \mathcal{L}$ is well-founded relative to O if:

- $x_0 = \bot$,
- $x_i \leq x_{i+1}$ and $x_{i+1} \in O(x_i)$ for every successor ordinal $i \geq 0$.
- $x_\lambda = (lub\{x_i\}_{i<\lambda})$ for a limit ordinal λ.

The complete constructive lower bound operator is defined as:

$$S^c(\mathcal{IC}^l_{\mathcal{P}})(y) = \{x \in \mathcal{O}_l(x, y) \mid \exists x_0, .., x \in \text{wfs}(\mathcal{IC}^l_{\mathcal{P}}(., y))\}$$

The complete constructive upper bound operator is defined analogously, and the constructive stable operator is defined as $S^c(\mathcal{IC}_{\mathcal{P}})(x, y) = (S^c(\mathcal{IC}^l_{\mathcal{P}})(y), S^c(\mathcal{IC}^u_{\mathcal{P}})(x))$.

Constructive Stable Operator

Given a non-deterministic operator $O : \mathcal{L} \to \wp(\mathcal{L})$, a sequence $x_0, \ldots, x_n \subseteq \mathcal{L}$ is well-founded relative to O if:

- $x_0 = \bot$,
- $x_i \leq x_{i+1}$ and $x_{i+1} \in O(x_i)$ for every successor ordinal $i \geq 0$.
- $x_\lambda = (lub\{x_i\}_{i<\lambda})$ for a limit ordinal λ.

The complete constructive lower bound operator is defined as:

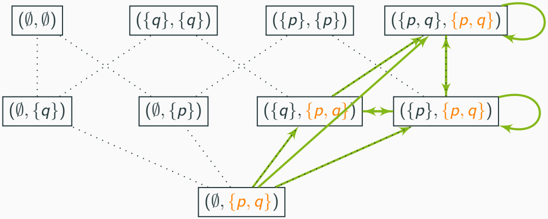
$$S^c(\mathcal{IC}^l_{\mathcal{P}})(y) = \{x \in \mathcal{O}_l(x, y) \mid \exists x_0, .., x \in \text{wfs}(\mathcal{IC}^l_{\mathcal{P}}(., y))\}$$

The complete constructive upper bound operator is defined analogously, and the constructive stable operator is defined as $S^c(\mathcal{IC}_{\mathcal{P}})(x, y) = (S^c(\mathcal{IC}^l_{\mathcal{P}})(y), S^c(\mathcal{IC}^u_{\mathcal{P}})(x))$.
A pair (x, y) is a constructive stable fixpoint iff $(x, y) \in S^c(\mathcal{IC}_{\mathcal{P}})(x, y)$.

Stable Operator: Example 1

$$\mathcal{P} = \{1\{p, q\}2 \leftarrow .,\quad p \leftarrow q\}$$

y	$\emptyset$	$\{p\}$	$\{q\}$	$\{p, q\}$
$\mathcal{IC}^{c,l}_{\mathcal{P}}(., y)$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{p, q\}$	$\{p\}, \{p, q\}$



$S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p, q\}) =$
$\{\{p\}, \{p, q\}\}$

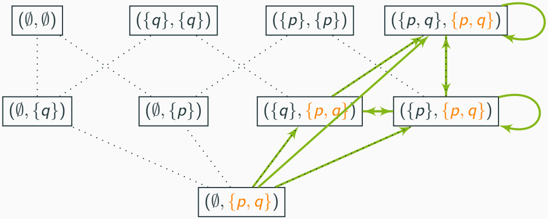
$(\{p\}, \{p\})$ is a stable fp.

$(\{p, q\}, \{p, q\})$ is a stable fp.

Stable Operator: Example 1

$$\mathcal{P} = \{1\{p, q\}2 \leftarrow ., \quad p \leftarrow q\}$$

y	$\emptyset$	$\{p\}$	$\{q\}$	$\{p, q\}$
$\mathcal{IC}^{c,l}_{\mathcal{P}}(., y)$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{q\}, \{p, q\}$	$\{p\}, \{p, q\}$	$\{p\}, \{p, q\}$



$S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p, q\}) =$
$\{\{p\}, \{p, q\}\}$

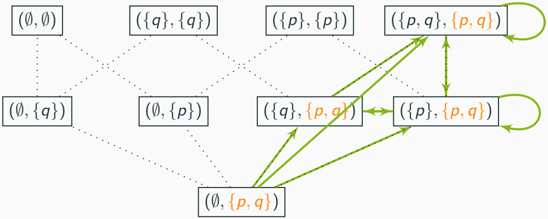
$(\{p\}, \{p\})$ is a stable fp.

$(\{p, q\}, \{p, q\})$ is a stable fp.

Stable Operator: Example 1

$$\mathcal{P} = \{1\{p,q\}2 \leftarrow ., \quad p \leftarrow q\}$$

y	$\emptyset$	$\{p\}$	$\{q\}$	$\{p,q\}$
$\mathcal{IC}^{c,l}_{\mathcal{P}}(.,y)$	$\{p\},\{q\},\{p,q\}$	$\{p\},\{q\},\{p,q\}$	$\{p\},\{p,q\}$	$\{p\},\{p,q\}$



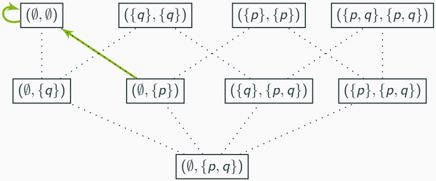
$S(\mathcal{IC}^{l}_{\mathcal{P}})(\{p,q\}) = \{\{p\},\{p,q\}\}$

$(\{p\},\{p\})$ is a stable fp.

$(\{p,q\},\{p,q\})$ is a stable fp.

Stable Operator: Example 2

$$\mathcal{P} = \{1\{p,q\} \leftarrow p.; \quad 1\{p,q\} \leftarrow q\}$$



Well-founded sequence of $IC^{l}_{\mathcal{P}}(., y)$ for any $y \subseteq \{p,q\}$: $\emptyset$

$(\emptyset, \emptyset)$ is the unique fixpoint of $S(\mathcal{IC}_{\mathcal{P}})$.

General results and characterisation theorem

Theorem
Let a choice program $\mathcal{P}$ s.t. for every $C_1 \leftarrow C_2, \ldots, C_n \in \mathcal{P}$, $\mathrm{dom}(C_i)$ is finite for $i = 1 \ldots n$, and $\mathrm{x} \in \{\mathrm{MR}, \mathrm{LPST}, \mathcal{U}\}$ and $x \subseteq y \subseteq \mathcal{A}$ be given. Then $S^c(\mathcal{IC}_{\mathcal{P}}^{\mathrm{x}})(x, y) \neq \emptyset$. Furthermore, $C^c(\mathcal{IC}_{\mathcal{P}}^{\mathrm{GZ},l})(y) \neq \emptyset$.

Theorem
Let a choice program $\mathcal{P}$ be given.

1. *x is a stable model according to [LPST10] iff (x, x) is a stable fixpoint of $\mathcal{IC}_{\mathcal{P}}^{\mathrm{LPST}}$.*
2. *x is a stable model according to [MR04] iff (x, x) is a stable fixpoint of $\mathcal{IC}_{\mathcal{P}}^{\mathrm{MR}}$.*
3. *If $\mathcal{P}$ is a aggregate program then x is a stable model according to [GZ14] iff $x \in C^c(\mathcal{IC}_{\mathcal{P}}^{\mathrm{GZ},l})(x)$.*

Disjunctions are Choice Constructs

$\mathtt{D2C}(\mathcal{P}) = \{1\Delta \leftarrow \phi \mid \bigvee \Delta \leftarrow \phi \in \mathcal{P}\}$.

Example $\mathtt{D2C}(\{p \vee q \leftarrow .\}) = \{1\{p \vee q\} \leftarrow .\}$.

Disjunctions are Choice Constructs

$\mathtt{D2C}(\mathcal{P}) = \{1\Delta \leftarrow \phi \mid \bigvee \Delta \leftarrow \phi \in \mathcal{P}\}$.

Example $\mathtt{D2C}(\{p \vee q \leftarrow .\}) = \{1\{p \vee q\} \leftarrow .\}$.

Proposition

For any disjunctive logic program $\mathcal{P}$, $\mathcal{IC}_{\mathcal{P}} = \mathcal{IC}^{c}_{\mathtt{D2C}(\mathcal{P})}$.

Disjunctions are Choice Constructs

$\mathtt{D2C}(\mathcal{P}) = \{1\Delta \leftarrow \phi \mid \bigvee \Delta \leftarrow \phi \in \mathcal{P}\}$.

Example $\mathtt{D2C}(\{p \vee q \leftarrow .\}) = \{1\{p \vee q\} \leftarrow .\}$.

Proposition
For any disjunctive logic program $\mathcal{P}$, $\mathcal{IC}_{\mathcal{P}} = \mathcal{IC}^{c}_{\mathtt{D2C}(\mathcal{P})}$.

Difference between DLP and choice programs: the difference is *not* in the treatment of disjunction and choice atoms (i.e. when they should be made true or false), but rather in how the stable semantics is constructed:

- Disjunctions: minimality-based stable semantics [?]
  $S(\mathcal{O}_u)(x) \;=\; \{y \in \mathcal{L} \mid y \in \mathcal{O}_u(x, y) \text{ and } \neg\exists y' < y : y' \in \mathcal{O}_u(x, y')\}$
  $\{p\}$ and $\{q\}$ are stable
- Choice constructs: constructive stable semantics
  $\{p\}$, $\{p, q\}$ and $\{q\}$ are stable

Taking Stock: Non-Deterministic Operators

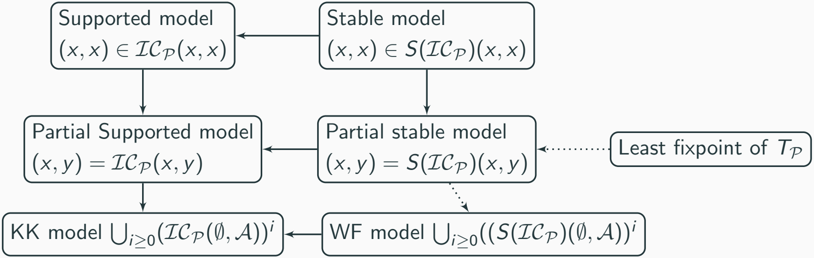
Semantics	=	Stable? S	+	Operator $(\mathcal{O})(x,y)$	+	Fixpoint $\ni (x,y)$
Partial Supported		No		$\mathcal{IC}_{\mathcal{P}}$		any fp
Supported		No		$\mathcal{IC}_{\mathcal{P}}$		total fp $(x = y)$
Partial Stable for DLPs		M-stable		$\mathcal{IC}_{\mathcal{P}}$		any fp
Stable for DLPs		M-stable		$\mathcal{IC}_{\mathcal{P}}$		total fp $(x = y)$
Partial Stable for CPs		C-stable		$\mathcal{IC}_{\mathcal{P}}$		any fp
Stable for CPs		C-stable		$\mathcal{IC}_{\mathcal{P}}$		total fp $(x = y)$

Taking Stock: Non-Deterministic Operators

Semantics	=	Stable? S	+	Operator $(\mathcal{O})(x, y)$	+	Fixpoint $\ni (x, y)$
Partial Supported		No		$\mathcal{IC}_{\mathcal{P}}$		any fp
Supported		No		$\mathcal{IC}_{\mathcal{P}}$		total fp $(x = y)$
Partial Stable for DLPs		M-stable		$\mathcal{IC}_{\mathcal{P}}$		any fp
Stable for DLPs		M-stable		$\mathcal{IC}_{\mathcal{P}}$		total fp $(x = y)$
Partial Stable for CPs		C-stable		$\mathcal{IC}_{\mathcal{P}}$		any fp
Stable for CPs		C-stable		$\mathcal{IC}_{\mathcal{P}}$		total fp $(x = y)$
		C-stable		$\mathcal{IC}_{\mathcal{P}}^{\mathsf{LPST}}$		any fp
[LPST10]		C-stable		$\mathcal{IC}_{\mathcal{P}}^{\mathsf{LPST}}$		total fp $(x = y)$
		C-stable		$\mathcal{IC}_{\mathcal{P}}^{\mathsf{MR}}$		any fp
[MR04]		C-stable		$\mathcal{IC}_{\mathcal{P}}^{\mathsf{MR}}$		total fp $(x = y)$
		C-stable		$\mathcal{IC}_{\mathcal{P}}^{\mathsf{GZ}}$		any fp
[GZ14]		C-stable		$\mathcal{IC}_{\mathcal{P}}^{\mathsf{GZ}}$		total fp $(x = y)$

Approximation Fixpoint Theory

Recap



- Operator-based framework
  - Non-monotonic operator $T_{\mathcal{P}}$,
  - a $\leq_i$-monotonic approximation operator $\mathcal{IC}_{\mathcal{P}}$,
  - and its stable variant $S(\mathcal{IC}_{\mathcal{P}})$.
- Allows us to define semantics as fixpoints of these operators, with attractive properties.
- Generalized to the non-deterministic setting, and with other operators.
- This story can be told for a great number of formalisms

Other Semantics

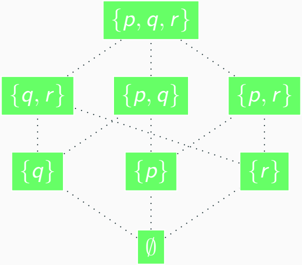
- State semantics for non-deterministic operators:
  - Kripke-Kleene state,
  - Well-founded state (relative to a stable construction),
  - Well-founded state with CWA (relative to a stable construction),
- Semi-equilibrium semantics,
- "Strongly" supported models,
- Regular models,
- M-Stable models,
- L-stable models.

Well-founded State

Basic Idea: generalize operator $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ to an operator $\mathcal{O}' : 2^{\mathcal{L}} \times 2^{\mathcal{L}} \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ as follows:

$$\mathcal{O}'((X,\ Y)) = \left(\bigcup_{x \in X, y \in Y} \mathcal{O}_l(x,y)\!\uparrow,\ \bigcup_{x \in X, y \in Y} \mathcal{O}_u(x,y)\!\downarrow \right)$$

Example $\mathcal{P} = \{1\{p,q\} \leftarrow .;\ q \leftarrow \neg r.\}$.

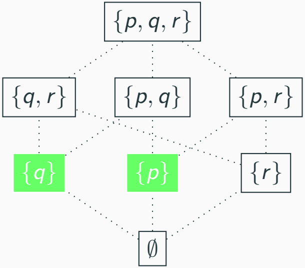


Well-founded State

Basic Idea: generalize operator $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ to an operator $\mathcal{O}' : 2^{\mathcal{L}} \times 2^{\mathcal{L}} \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ as follows:

$$\mathcal{O}'((X,\ Y)) = \left(\bigcup_{x \in X, y \in Y} \mathcal{O}_l(x, y)\!\uparrow,\ \bigcup_{x \in X, y \in Y} \mathcal{O}_u(x, y)\!\downarrow \right)$$

Example $\mathcal{P} = \{1\{p, q\} \leftarrow .; q \leftarrow \neg r.\}$.

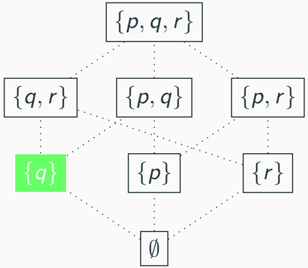


Well-founded State

Basic Idea: generalize operator $\mathcal{O} : \mathcal{L}^2 \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ to an operator $\mathcal{O}' : 2^{\mathcal{L}} \times 2^{\mathcal{L}} \to 2^{\mathcal{L}} \times 2^{\mathcal{L}}$ as follows:

$$\mathcal{O}'((X,\ Y)) = \left(\bigcup_{x \in X, y \in Y} \mathcal{O}_l(x, y)\uparrow,\ \bigcup_{x \in X, y \in Y} \mathcal{O}_u(x, y)\downarrow \right)$$

Example $\mathcal{P} = \{1\{p, q\} \leftarrow .; q \leftarrow \neg r.\}$.



HT-Models, algebraically [Tru06, HB23]

Given an ndao $\mathcal{O}$ approximating a non-determinstic operator O, a pair (x, y) is a HT-*pair* (denoted $(x, y) \in \mathrm{HT}(\mathcal{O})$) if the following three conditions are satisfied:

- $x \leq y$,
- $O(y) \preceq_L^S y$, and
- $\mathcal{O}_l(x, y) \preceq_L^S x$.

PropositionLet some normal disjunctive logic program $\mathcal{P}$ be given. Then: $\mathrm{HT}(\mathcal{P}) = \mathrm{HT}(\mathcal{IC}_{\mathcal{P}})$.

Proposition Given an upwards coherent ndao $\mathcal{O}$, (1) if $(x, x) \in \mathcal{O}(x, x)$ then $(x, x) \in \mathrm{HT}(\mathcal{O})$; and (2) $(x, x) \in \min_{\leq_t}(\mathrm{HT}(\mathcal{O}))$ iff $(x, x) \in S(\mathcal{O})(x, x)$.

Operator-Based Semantics for Dialects of Logic Programming

- $\vee$ Aggregates in the body: $p \leftarrow \#\mathtt{sum}\{2 : p; q : 1; r : 1\} \geq 2$.
- $\vee$ Propositional formulas in the body: $p \leftarrow q \wedge (r \vee (s \wedge \neg t))$.
- $\vee$ Disjunctions in the head: $p \vee q \leftarrow q \wedge (r \vee (s \wedge \neg t))$.
- $\vee$ Choice constructs in the head: $\#\mathtt{count}\{p; q; r\} = 2 \leftarrow \neg r$.
- $\vee$ DL-based logic programs: $\mathbf{K} C(x) \leftarrow \neg p(X); C \sqsubseteq D$.
- $\vee$ Higher-order logic programs: $S(P, Q) \leftarrow; P(X) \leftarrow \neg Q(X)$.
- ... Fuzzy logic programs: $p(X) \leftarrow 0.5 \cdot (q(x) + r(X))$.
- ... Semi-ring constraint programs: $p(X) \leftarrow 0.5 \otimes (q(x) \oplus r(X))$.
- ? Probabilistic logic programs: $0.3 :: p(X)$.
- ? Hex-programs: $tr(S, P, O) \leftarrow \& RDF[uri](S, P, O)$.

Operator-Based Semantics for other KR-formalisms

- autoepistemic logic [DMT03],
- default logic [DMT03],
- abstract argumentation [SW15],
- abstract dialectical frameworks [SW15],
- weighted abstract dialectical frameworks [Bog19],
- SCHACL [BJ21].

Operator-Based Studies

Top-Down approach:

- Instead of studying a concept for a specific framework, define and study it for operators over a lattice (and their approximations).
- We can then apply this concept to all formalisms that are or can be captured in AFT.

Examples:

V Stratification [VGD06]

V Conditional Independence [Hey23]

V Knowledge Compilation [BVdB15]

V Groundedness [BVdB15]

V Strong equivalence [Tru06]

V Argumentative dialogues [HA20]

? Belief dynamics

? Modular equivalence

? Neuro-symbolism

Round up

Summary

- Operators as the core for understanding answer set semantics.
- Paved the road towards approximation fixpoint theory.
- Algebraic theory that allows language independent work on KR.
- Requires some buy-in, but in my view a great bargain.
- Interested in cooperating? Questions on AFT? Come talk to me.

Bart Bogaerts and Maxime Jakubowski.
Fixpoint semantics for recursive shacl.
In *37th International Conference on Logic Programming*, pages 41–47. Open Publishing Association, 2021.

Bart Bogaerts.
Weighted abstract dialectical frameworks through the lens of approximation fixpoint theory.
In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2686–2693, 2019.

Bibliography ii

Bart Bogaerts and Guy Van den Broeck.
Knowledge compilation of logic programs using approximation fixpoint theory.
Theory and Practice of Logic Programming, 15(4-5):464–480, 2015.

Marc Denecker, Victor Marek, and Mirosław Truszczyński.
Approximations, stable operators, well-founded fixpoints and applications in nonmonotonic reasoning.
In *Logic-based Artificial Intelligence*, volume 597 of *The Springer International Series in Engineering and Computer Science*, pages 127–144. Springer, 2000.

Marc Denecker, Victor Marek, and Mirosław Truszczyński.
Uniform semantic treatment of default and autoepistemic logics.
Artificial Intelligence, 143(1):79–122, 2003.

Melvin Fitting.
Bilattices are nice things.
In *Self Reference*, volume 178 of *CSLI Lecture Notes*, pages 53–77. CLSI Publications, 2006.

Michael Gelfond and Yuanlin Zhang.
Vicious circle principle and logic programs with aggregates.
Theory and Practice of Logic Programming, 14(4-5):587–601, 2014.

Jesse Heyninck and Ofer Arieli.
Argumentative reflections of approximation fixpoint theory.
In *Computational Models of Argument*, pages 215–226. IOS Press, 2020.

Bibliography v

Jesse Heyninck and Bart Bogaerts.
Non-deterministic approximation operators: Ultimate operators, semi-equilibrium semantics, and aggregates.
Theory Pract. Log. Program., 23(4):632–647, 2023.

Jesse Heyninck.
An algebraic notion of conditional independence, and its application to knowledge representation (preliminary report).
2023.

Lengning Liu, Enrico Pontelli, Tran Cao Son, and Miroslaw Truszczyński.
Logic programs with abstract constraint atoms: The role of computations.
Artificial Intelligence, 174(3-4):295–315, 2010.

Victor W Marek and Jeffrey B Remmel.
Set constraints in logic programming.
In *Logic Programming and Nonmonotonic Reasoning: 7th International Conference, LPNMR 2004 Fort Lauderdale, FL, USA, January 6-8, 2004 Proceedings 7*, pages 167–179. Springer, 2004.

Teodor C Przymusinski.
Stable semantics for disjunctive programs.
New generation computing, 9(3):401–424, 1991.

Hannes Strass and Johannes Peter Wallner.
Analyzing the computational complexity of abstract dialectical frameworks via approximation fixpoint theory.
Artificial Intelligence, 226:34–74, 2015.

Mirosław Truszczyński.
Strong and uniform equivalence of nonmonotonic theories–an algebraic approach.
Annals of Mathematics and Artificial Intelligence, 48(3-4):245–265, 2006.

Joost Vennekens, David Gilis, and Marc Denecker.
Splitting an operator: Algebraic modularity results for logics with fixpoint semantics.
ACM Transactions on computational logic (TOCL), 7(4):765–797, 2006.

Allen Van Gelder, Kenneth A Ross, and John S Schlipf.
The well-founded semantics for general logic programs.
Journal of the ACM, 38(3):619–649, 1991.