Implementation of the Top Down Approach for Modular Nonmonotonic Logic Programs

Tri Kurniawan Wijaya

KBS Group
Institute of Information Systems
Faculty of Informatics
Vienna University of Technology

Lisbon, 17-18 February 2011

Layout

Motivation

- Problem decomposition

Motivation

- Problem decomposition
- Code reusing

Motivation

- Problem decomposition
- Code reusing
- Exploit the modularity of logic programs

Basic idea

Imperative Programming

- Declaration:
  $int\ p(int\ x, int\ y)$
  $\{$
  $\ldots$
  $\}$

- Use:
  $x = p(a, b);$

Basic idea

Imperative Programming

- Declaration:
  $int\ p(int\ x,\ int\ y)$
  $\{$
  $\ldots$
  $\}$
- Use:
  $x = p(a, b);$

MLP

- Declaration:
  Module
  $m = (P[q_1, q_2], R)$
  - P is a module name
  - q_1, q_2 are predicate name (as inputs)
  - R is a set of rules
- Use:
  $p(X) \leftarrow P[q, r].out(X)$
  where *out* is the desired *output predicate*

Example: reachability

$m_1 = (P_1, R_1)$, $m_2 = (P_2[first, edge], R_2)$, where:

Example: reachability

$m_1 = (P_1, R_1),\ m_2 = (P_2[\mathit{first}, \mathit{edge}], R_2)$, where:

- $R_1 = \{v(a).$
  $e(a, b).\ e(b, c).\ e(c, d).$
  $rc(c) \leftarrow P_2[v, e].\mathit{reachable}(c).$
  $r(X) \leftarrow P_2[v, e].\mathit{reachable}(X).\}$

Example: reachability

$m_1 = (P_1, R_1)$, $m_2 = (P_2[first, edge], R_2)$, where:

- $R_1 = \{v(a).$
  $e(a, b).\ e(b, c).\ e(c, d).$
  $rc(c) \leftarrow P_2[v, e].reachable(c).$
  $r(X) \leftarrow P_2[v, e].reachable(X).\}$
- $R_2 = \{reachable(X) \leftarrow first(X).$
  $reachable(Y) \leftarrow reachable(X), edge(X, Y).\}$

Example: cardinality

$m_1 = (P_1, R_1)$, $m_2 = (P_2[q_1, q_2], R_2)$, where:

- $R_1 = \{q(a).\ q(b).$
  $r(a).\ r(b).$
  $ok \leftarrow P_2[q, r].equal.\}$

Example: cardinality

$m_1 = (P_1, R_1)$, $m_2 = (P_2[q_1, q_2], R_2)$, where:

- $R_1 = \{q(a).\ q(b).$
  $r(a).\ r(b).$
  $ok \leftarrow P_2[q, r].equal.\}$
- $R_2 = \{q_1'(X) \vee q_1'(Y) \leftarrow q_1(X), q_1(Y), X \neq Y.$
  $q_2'(X) \vee q_2'(Y) \leftarrow q_2(X), q_2(Y), X \neq Y.$
  $skip_1 \leftarrow q_1(X), not\ q_1'(X).$
  $skip_2 \leftarrow q_2(X), not\ q_2'(X).$

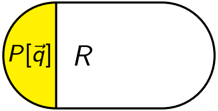
Example: cardinality

$m_1 = (P_1, R_1)$, $m_2 = (P_2[q_1, q_2], R_2)$, where:

- $R_1 = \{q(a).\ q(b).$
  $r(a).\ r(b).$
  $ok \leftarrow P_2[q, r].equal.\}$
- $R_2 = \{q_1'(X) \vee q_1'(Y) \leftarrow q_1(X), q_1(Y), X \neq Y.$
  $q_2'(X) \vee q_2'(Y) \leftarrow q_2(X), q_2(Y), X \neq Y.$
  $skip_1 \leftarrow q_1(X), not\ q_1'(X).$
  $skip_2 \leftarrow q_2(X), not\ q_2'(X).$
  $equal \leftarrow skip_1, skip_2, P_2[q_1', q_2'].equal.$
  $equal \leftarrow not\ skip_1, not\ skip_2.\}$

Syntax

- module
  - $m = (P[\mathbf{q}], R)$
  - P with associated input $\mathbf{q} = q_1, \ldots, q_k$ (list of predicate names)
  - R is a finite set of rules
  - *main modules* $|\mathbf{q}| = 0$, otherwise *library module*



- module atom
  - $P[p_1, \ldots, p_k].o(t_1, \ldots, t_l)$ where P is a module name, $p_1, \ldots, p_k$ are predicate names (module input list), $o(t_1, \ldots, t_l)$ is an ordinary atom (output predicate)

Syntax (2)

- modular logic program
  - $\mathbf{P} = (m_1, \ldots, m_n), n \geq 1$
  - All m_i are modules
  - At least contains one main module

Value call

- Notation: for $S \subseteq HB_{\mathbf{P}}$, $\mathbf{p} = p_1, \ldots, p_k$, and $\mathbf{q} = q_1, \ldots, q_k$
  - $S|_{\mathbf{p}} = \{p_i(\mathbf{c}) \in S | i \in \{1, \ldots, k\}\}$ (restriction)
  - $S|_{\mathbf{p}}^{\mathbf{q}} = \{q_i(\mathbf{c}) | p_i(\mathbf{c}) \in S, i \in \{1, \ldots, k\}\}$ (replacement)

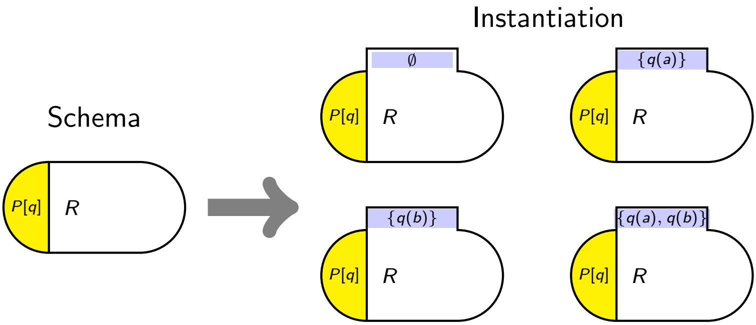
Value call

- Notation: for $S \subseteq HB_{\mathbf{P}}$, $\mathbf{p} = p_1, \ldots, p_k$, and $\mathbf{q} = q_1, \ldots, q_k$
  - $S|_{\mathbf{p}} = \{p_i(\mathbf{c}) \in S | i \in \{1, \ldots, k\}\}$ (restriction)
  - $S|_{\mathbf{p}}^{\mathbf{q}} = \{q_i(\mathbf{c}) | p_i(\mathbf{c}) \in S, i \in \{1, \ldots, k\}\}$ (replacement)
- Value Call: For $P \in \mathcal{M}$ with associated formal input $\mathbf{q}$
  - $P[S]$ is a *value call with input* S
  - $S \subseteq HB_{\mathbf{P}}|_{\mathbf{q}}$

Value call

- Notation: for $S \subseteq HB_{\mathbf{P}}$, $\mathbf{p} = p_1, \ldots, p_k$, and $\mathbf{q} = q_1, \ldots, q_k$
  - $S|_{\mathbf{p}} = \{p_i(\mathbf{c}) \in S | i \in \{1, \ldots, k\}\}$ (restriction)
  - $S|_{\mathbf{p}}^{\mathbf{q}} = \{q_i(\mathbf{c}) | p_i(\mathbf{c}) \in S, i \in \{1, \ldots, k\}\}$ (replacement)
- Value Call: For $P \in \mathcal{M}$ with associated formal input $\mathbf{q}$
  - $P[S]$ is a *value call with input* S
  - $S \subseteq HB_{\mathbf{P}}|_{\mathbf{q}}$
  - Example:
    - $HB_{\mathbf{P}}|_{\mathbf{q}} = \{q(a), q(b)\}$
    - $P[\emptyset]$
    - $P[\{q(a)\}]$
    - $P[\{q(b)\}]$
    - $P[\{q(a), q(b)\}]$
- $VC(\mathbf{P})$: set of all value calls

Call instantiation



Example

$m_1 = (P_1, R_1)$,
$m_2 = (P_2[r], R_2)$, and
$m_3 = (P_3[u], R_3)$, where:

- $R_1 =$
  $\{q(a).\ q(b).$
  $out_1 \leftarrow P_2[q].out_2.\}$
- $R_2 =$
  $\{s(c) \leftarrow r(c).$
  $out_2 \leftarrow not\ r(X).\}$
  $out_2 \leftarrow P_3[s].out_3.\}$
- $R_3 =$
  $\{out_3 \leftarrow P_2[u].out_2.\}$

Example

$m_1 = (P_1, R_1)$,
$m_2 = (P_2[r], R_2)$, and
$m_3 = (P_3[u], R_3)$, where:

- $R_1 = \{q(a).\ q(b).\ out_1 \leftarrow P_2[q].out_2.\}$
- $R_2 = \{s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\}\ out_2 \leftarrow P_3[s].out_3.\}$
- $R_3 = \{out_3 \leftarrow P_2[u].out_2.\}$

Module $m_2(P_2[r], R_2)$

- $R_2 = \{s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\}\ out_2 \leftarrow P_3[s].out_3.\}$
- with an input $\{r(a), r(b)\}$
- its instantiation $P_2[\{r(a), r(b)\}]$: $\{r(a).\ r(b).\ s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\}\ out_2 \leftarrow P_3[s].out_3.\}$

Interpretation

An interpretation **M** of an MLP **P**

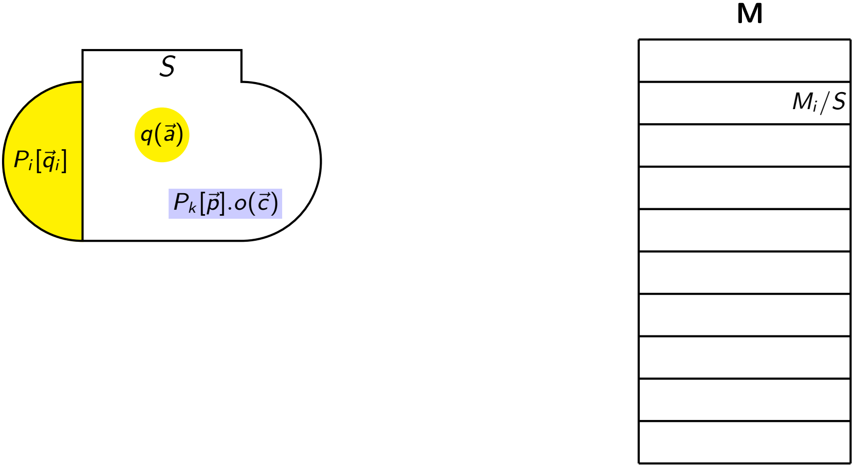
- A (indexed) tuple $(M_i/S|P_i[S] \in VC(\mathbf{P}))$
- All $M_i/S \subseteq HB_{\mathbf{P}}$ contain only ordinary atoms
- Example:
  - $M_1/\emptyset = \{q(a), q(b), out_1\}$
  - $M_2/\{r(a), r(b)\} = \{r(a), r(b), out_2\}$
  - $M_3/\emptyset = \{out_3\}$

Models

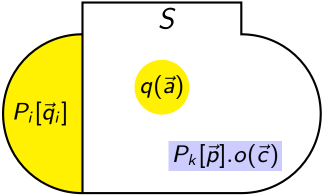
An interpretation **M** of an MLP **P** is a model of:

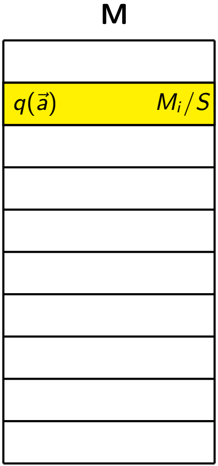
- a ground atom $\alpha \in HB_{\mathbf{P}}$ at $P_i[S]$ ($\mathbf{M}, P_i[S] \models \alpha$), iff:
  - α is ordinary: $\alpha \in M_i/S$
  - $\alpha = P_k[\mathbf{p}].o(\mathbf{c})$ is a module atom: $o(c) \in M_k/((M_i/S)|_{\mathbf{p}}^{\mathbf{q}_k})$

Models

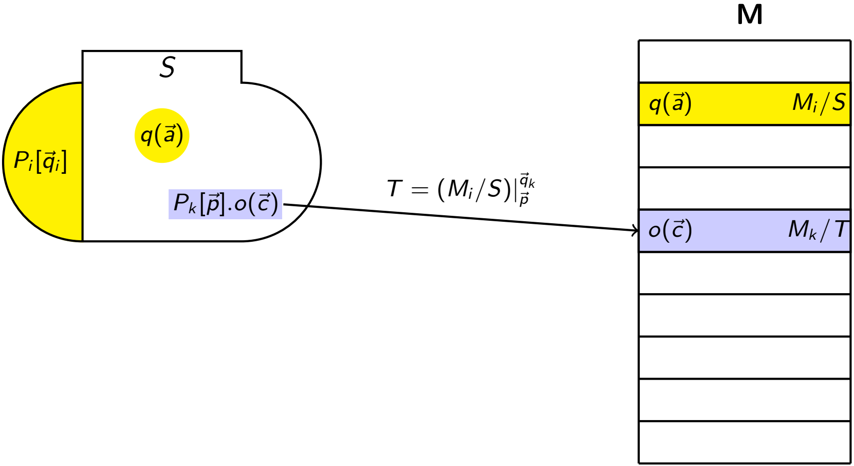


Models





Models


M
S
$P_i[\vec{q}_i]$
$q(\vec{a})$
$P_k[\vec{p}].o(\vec{c})$
$T = (M_i/S)|_{\vec{p}}^{\vec{q}_k}$
$q(\vec{a})$ M_i/S
$o(\vec{c})$ M_k/T

Call graph

Call graph of an MLP **P** is a labeled digraph $CG_{\mathbf{P}} = (V, E, l)$ with

- vertex set $V = VC(\mathbf{P})$
- edge $e \in E$ from $P_i[S]$ to $P_k[T]$ iff $P_k[\mathbf{p}].o(\mathbf{t})$ occurs in $R(m_i)$
- e is labeled with an input list **p**, denoted l(e)

Relevant call graph

- contains only relevant vertexes and edges w.r.t. **M**

Example

$m_1 = (P_1, R_1)$,
$m_2 = (P_2[r], R_2)$, and
$m_3 = (P_3[u], R_3)$, where:

- $R_1 = \{q(a).\ q(b).\ out_1 \leftarrow P_2[q].out_2.\}$
- $R_2 = \{s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\ out_2 \leftarrow P_3[s].out_3.\}$
- $R_3 = \{out_3 \leftarrow P_2[u].out_2.\}$

$$P_1[\emptyset] \rightarrow^q P_2[\{r(a), r(b)\}]$$

Example

$m_1 = (P_1, R_1)$,
$m_2 = (P_2[r], R_2)$, and
$m_3 = (P_3[u], R_3)$, where:

- $R_1 = \{q(a).\ q(b).\ out_1 \leftarrow P_2[q].out_2.\}$
- $R_2 = \{s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\ out_2 \leftarrow P_3[s].out_3.\}$
- $R_3 = \{out_3 \leftarrow P_2[u].out_2.\}$

$$P_1[\emptyset] \rightarrow^q P_2[\{r(a), r(b)\}] \rightarrow^q P_2[\emptyset]$$

Example

$m_1 = (P_1, R_1)$,
$m_2 = (P_2[r], R_2)$, and
$m_3 = (P_3[u], R_3)$, where:

- $R_1 = \{q(a).\ q(b).\ out_1 \leftarrow P_2[q].out_2.\}$
- $R_2 = \{s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\ out_2 \leftarrow P_3[s].out_3.\}$
- $R_3 = \{out_3 \leftarrow P_2[u].out_2.\}$

$$P_1[\emptyset] \rightarrow^q P_2[\{r(a), r(b)\}] \rightarrow^q P_2[\emptyset] \rightarrow^q P_2[\{r(a)\}]$$

Example

$m_1 = (P_1, R_1)$,
$m_2 = (P_2[r], R_2)$, and
$m_3 = (P_3[u], R_3)$, where:

- $R_1 = \{q(a).\ q(b).\ out_1 \leftarrow P_2[q].out_2.\}$
- $R_2 = \{s(c) \leftarrow r(c).\ out_2 \leftarrow not\ r(X).\ out_2 \leftarrow P_3[s].out_3.\}$
- $R_3 = \{out_3 \leftarrow P_2[u].out_2.\}$

$$
\begin{aligned}
P_1[\emptyset] &\rightarrow^q P_2[\{r(a), r(b)\}] \\
&\rightarrow^q P_2[\emptyset] \\
&\rightarrow^q P_2[\{r(a)\}] \\
&\rightarrow^q P_2[\{r(b)\}]
\end{aligned}
$$

Example: call graph vs relevant call graph

Call graph:

$$
\begin{aligned}
P_1[\emptyset] &\rightarrow^q P_2[\{r(a), r(b)\}] \\
&\rightarrow^q P_2[\emptyset] \\
&\rightarrow^q P_2[\{r(a)\}] \\
&\rightarrow^q P_2[\{r(b)\}]
\end{aligned}
$$

Example: call graph vs relevant call graph

Call graph:

$$P_1[\emptyset] \to^q P_2[\{r(a), r(b)\}]$$
$$\to^q P_2[\emptyset]$$
$$\to^q P_2[\{r(a)\}]$$
$$\to^q P_2[\{r(b)\}]$$

Relevant call graph:

$\mathbf{M} =$

- $M_1/\emptyset = \{q(a), q(b), out_1\}$
- $M_2/\{r(a), r(b)\} = \{r(a), r(b), out_2\}$
- $M_3/\emptyset = \{out_3\}$

Example: call graph vs relevant call graph

Call graph:

$$P_1[\emptyset] \rightarrow^q P_2[\{r(a), r(b)\}]$$
$$\rightarrow^q P_2[\emptyset]$$
$$\rightarrow^q P_2[\{r(a)\}]$$
$$\rightarrow^q P_2[\{r(b)\}]$$

Relevant call graph:

$\mathbf{M} =$

- $M_1/\emptyset = \{q(a), q(b), out_1\}$
- $M_2/\{r(a), r(b)\} = \{r(a), r(b), out_2\}$
- $M_3/\emptyset = \{out_3\}$

$CG_{\mathbf{P}}(\mathbf{M}) =$

$$P_1[\emptyset] \rightarrow^q P_2[\{r(a), r(b)\}]$$

Example: call graph vs relevant call graph

Call graph:

$$\begin{aligned} P_1[\emptyset] &\rightarrow^q P_2[\{r(a), r(b)\}] \\ &\rightarrow^q P_2[\emptyset] \\ &\rightarrow^q P_2[\{r(a)\}] \\ &\rightarrow^q P_2[\{r(b)\}] \end{aligned}$$

Relevant call graph:

$\mathbf{M} =$

- $M_1/\emptyset = \{q(a), q(b), out_1\}$
- $M_2/\{r(a), r(b)\} = \{r(a), r(b), out_2\}$
- $M_3/\emptyset = \{out_3\}$

$CG_{\mathbf{P}}(\mathbf{M}) =$

$$P_1[\emptyset] \rightarrow^q P_2[\{r(a), r(b)\}] \rightarrow^\emptyset P_3[\emptyset] \rightleftharpoons^\emptyset P_2[\emptyset]$$

Answer set

- A context for an interpretation **M** of an MLP **P** is C, where:
  - $V(CG_{\mathbf{P}}(\mathbf{M})) \subseteq C \subseteq VC(\mathbf{P})$
- Let **M** be an interpretation of a ground MLP **P**.
  Then **M** is an answer set of **P** w.r.t. a context C for **M**, iff **M** is a minimal model of $f\mathbf{P}^{\mathbf{M},C}$.
- The reduct used here is FLP-reduct [Faber et al., 2004]

c-stratified

P is c-stratified (call stratified) w.r.t. **M** (where **M** is an interpretation of **P**) iff

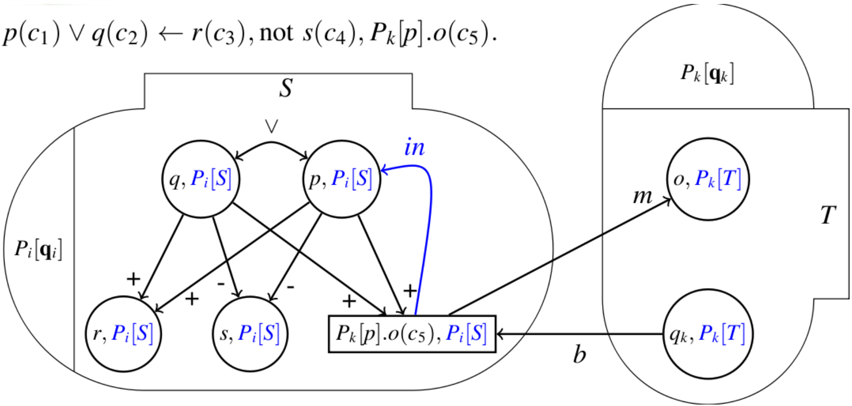
- cycles in $CG_{\mathbf{P}}(\mathbf{M})$ contain only nodes of the form $P_i[\emptyset]$

Example:

$P_1[\emptyset] \rightarrow^{q} P_2[\{r(a), r(b)\}] \rightarrow^{\emptyset} P_3[\emptyset] \rightleftharpoons^{\emptyset} P_2[\emptyset]$

Dependency graph

$$p(c_1) \vee q(c_2) \leftarrow r(c_3), \text{not } s(c_4), P_k[p].o(c_5).$$



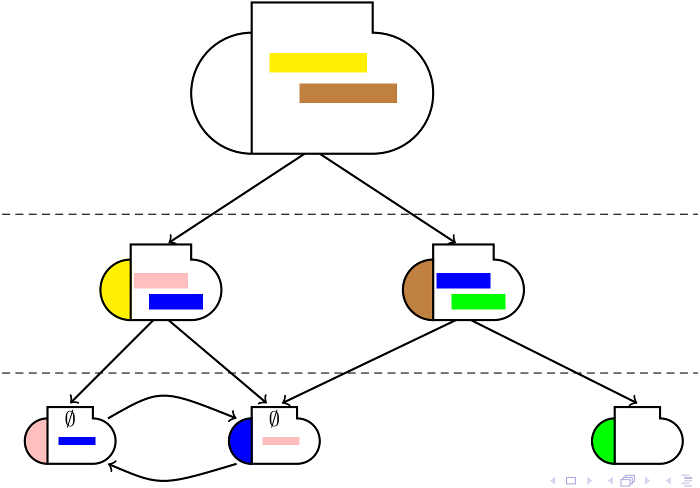
i-stratified and ic-stratified

P is i-stratified w.r.t. **M** (where **M** is the interpretation of **P**), iff

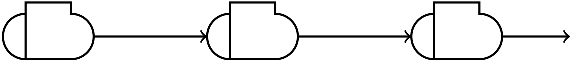
- cycles with in-edges in $G_{\mathbf{P}}^{\mathbf{M}}$ contain only nodes of the form $(X, P_i[\emptyset])$
- allowed us to "prepare" the input predicates completely.

P is ic-stratified iff it is both i-stratified and c-stratified.

Intuition



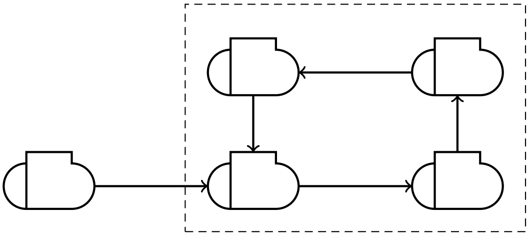
Cycle detection



Maintaining a "path" of sets of visited value calls.

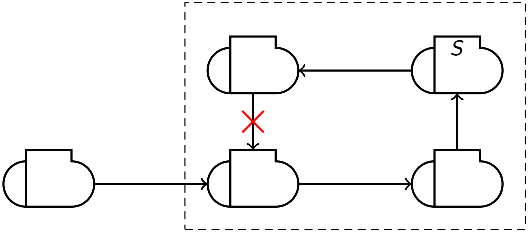
Each element in "path" is initialized by a value call.

Cycle detection



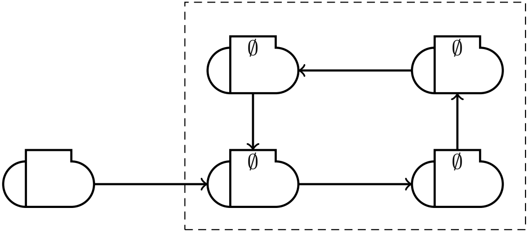
When a cycle is detected, there are two cases:

Cycle detection



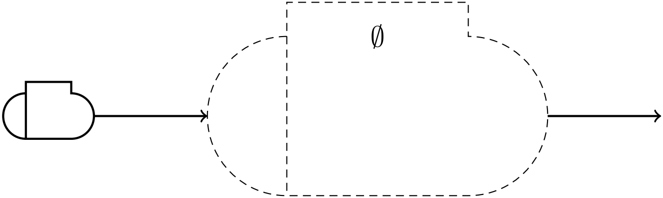
(1) A member of the cycle has non-empty input, not c-stratified.

Cycle detection



(2) All members of the cycle have empty input

Cycle detection



Combine all value calls of the cycle

Main ingredients

- Cycles detection
- *rewrite*(C, M, A)
  - C = set of value calls
- When at least a module atom α in R
  - α = smallest $ill(R)$
  - b_U: rule set, the bottom of R for α
- $ans(R)$: compute the answer set of an *ordinary* rule set
- $M = mlpize(N, C)$
  - Put the answer set N from the module calls in C into a partial interpretation M

Conclusions and future works

Conclusions

- Extending an ordinary logic program to the modular ones
- Defining stratifications that admit an "efficient" top down evaluation algorithm

Future works

- Optimize the current algorithm
- Finer stratifications
- Another approach to evaluate a bigger subclass of MLP

Further reading: [Dao-Tran et al., 2009]

Dao-Tran, M., Eiter, T., Fink, M., and Krennwallner, T. (2009).
Relevance-driven evaluation of modular nonmonotonic logic programs.
In *Proceedings of the 10th International Conference on Logic Programming and Nonmonotonic Reasoning*, LPNMR '09, pages 87–100, Berlin, Heidelberg. Springer-Verlag.

Faber, W., Leone, N., and Pfeifer, G. (2004).
Recursive aggregates in disjunctive logic programs: Semantics and complexity.
In *In Proceedings of European Conference on Logics in Artificial Intelligence (JELIA*, pages 200–212. Springer.

Appendix:

Definition 1 (model)

An interpretation $\mathbf{M}$ of an MLP $\mathbf{P}$ is a (indexed) tuple $(M_i/S | P_i[S] \in VC(\mathbf{P}))$, where all $M_i/S \subseteq HB_{\mathbf{P}}$ contain only ordinary atoms.
An interpretation $\mathbf{M}$ of an MLP $\mathbf{P}$ is a model of:

- a ground atom $\alpha \in HB_{\mathbf{P}}$ at $P_i[S]$, denoted $\mathbf{M}, P_i[S] \models \alpha$, iff:
  - $\alpha \in M_i/S$ when α is ordinary, and
  - $o(c) \in M_k/((M_i/S)|_{\mathbf{p}}^{\mathbf{q}_k})$, when $\alpha = P_k[\mathbf{p}].o(\mathbf{c})$ is a module atom
- a ground rule r at $P_i[S]$ ($\mathbf{M}, P_i[S] \models r$), iff
  - $\mathbf{M}, P_i[S] \models H(r)$, or
  - $\mathbf{M}, P_i[S] \not\models B(r)$
- a set of ground rules R at $P_i[S](\mathbf{M}, P_i[S] \models R)$ iff $\mathbf{M}, P_i[S] \models r$ for all $r \in R$
- a ground rule base $\mathbf{R}(\mathbf{M} \models \mathbf{R})$ iff $\mathbf{M}, P_i[S] \models \mathbf{R}_{P_i[S]}$ for all $P_i[S] \in VC(\mathbf{P})$

Definition 2 (call graph)

Call graph of an MLP **P** is a labeled digraph $CG_{\mathbf{P}} = (V, E, l)$ with

- vertex set $V = VC(\mathbf{P})$
- edge $e \in E$ from $P_i[S]$ to $P_k[T]$ iff $P_k[\mathbf{p}].o(\mathbf{t})$ occurs in $R(m_i)$
- e is labeled with an input list **p**, denoted l(e)

Relevant call graph $CG_{\mathbf{P}}(\mathbf{M}) = (V', E')$ of **P** w.r.t. **M**

- subgraph of $CG_{\mathbf{P}}$
- E' contains all edges from $P_i[S]$ to $P_k[T]$ of $CG_{\mathbf{P}}$ s.t. $(M_i/S)|^{\mathbf{q_k}}_{l(e)} = T$
- V' contains all $P_i[S]$ that are main module instantiations or induced by E'

Definition 3 (context-based reduct)

- A context for an interpretation **M** of an MLP **P** is any set $C \subseteq VC(\mathbf{P})$ such that $V(CG_{\mathbf{P}}(\mathbf{M})) \subseteq C$.
- The reduct of **P** at $P[S]$ w.r.t. **M** and C, denoted $f\mathbf{P}(P[S])^{\mathbf{M},C}$, is the rule set $I_{gr(\mathbf{P})}(P[S])$ from which, if $P[S] \in C$, all rules r such that $\mathbf{M}, P[S] \models B(r)$ are removed.
- The reduct of **P** w.r.t. **M** and C is $f\mathbf{P}^{\mathbf{M},C} = (f\mathbf{P}(P[S])^{\mathbf{M},C} | P[S] \in VC(P))$

Definition 4 (Answer Set)

Let $\mathbf{M}$ be an interpretation of a ground MLP $\mathbf{P}$. Then $\mathbf{M}$ is an answer set of $\mathbf{P}$ w.r.t. a context C for $\mathbf{M}$, iff $\mathbf{M}$ is a minimal model of $f\mathbf{P}^{\mathbf{M},C}$.

Definition 5 (c-stratified)

$\mathbf{P}$ is c-stratified (call stratified) w.r.t. $\mathbf{M}$ (where $\mathbf{M}$ is an interpretation of $\mathbf{P}$) iff cycles in $CG_{\mathbf{P}}(\mathbf{M})$ contain only nodes of the form $P_i[\emptyset]$

Instance stratification

Instance dependency graph = $G_{\mathbf{P}}^{\mathbf{M}} = (IV, IE)$ of $\mathcal{P}$ w.r.t. an interpretation $\mathcal{M}$

- $IV = (p, P_i[S])$ or $(\alpha, P_i[S])$, where p (resp. α) is a predicate name (resp. module atom)
- $IE =$
  - (i') - (iv') similar to (i') - (iv')
  - (v') $(\alpha, P_i[S]) \rightarrow^m (o, P_i[(M_i/S)|_{\mathbf{p}}^{\mathbf{q_j}}])$
  - (vi') $(q_l, P_j[(M_i/S)|_{\mathbf{p}}^{\mathbf{q_j}}]) \rightarrow^b (p_l, P_i[S])$, where $q_l \in \mathbf{q_j}$ of $P_j[\mathbf{q_j}]$ and $p_l \in \mathbf{p}$ of α

Definition 6 (i-stratified)

P is i-stratified w.r.t. **M** (where **M** is the interpretation of **P**), iff cycles with in-edges in $G_{\mathbf{P}}^{\mathbf{M}}$ contain only nodes of the form $(X, P_i[\emptyset])$.

Definition 7 (ic-stratified)

P is ic-stratified iff it is both i-stratified and c-stratified.

Splitting set

Definition 8 (input splitting set)

Given an MLP $\mathbf{P}$, a set of ground rules R, and α as a ground module atom of the form $P_k[\mathbf{p}].o(\mathbf{c})$, then U is an input splitting set of R for α, iff:

1. $U \subseteq HB_{\mathbf{P}}$
2. for any rule $r \in R$, if $H(r) \cap U \neq \emptyset$ then $at(r) \subseteq U$
3. $\alpha \notin U$
4. $def(\mathbf{p}, R) \subseteq U$, where $\mathbf{p} = \{p_1, \ldots, p_k\}$ is a list of predicate names, $def(\mathbf{p}, R) = \{p_l(\mathbf{d}) | \exists r \in R, p_l(\mathbf{d}) \in H(r), p_l \in \mathbf{p}\}$

The *bottom* of a set of ground rules R w.r.t. a set of atoms $U \subseteq HB_{\mathbf{P}}$ is $b_U(R) = \{r \in R | H(r) \cap U \neq \emptyset\}$

Helper subroutines

- *rewrite*$(C, \mathbf{M}, \mathbf{A})$:
  - For all $P_i[S] \in C$, put into a set R all rules in $I_{\mathbf{P}}(P_i[S])$, and M_i/S as facts if not nil,
  - prefixing every ordinary atom (appearing in a rule or fact) with $P_i[S]$.
  - replace each module atom $\alpha = P_j[p].o(t)$ in R, such that $\alpha \in A_i/S$, by o prefixed with $P_j[T]$,
  - where $T = (M_i/S)|_{\mathbf{p_i}}^{\mathbf{q_j}}$, and p_i is p without prefixes;
  - add any atoms from $(M_j/T)|_o$ prefixed by $P_j[T]$ to R.
- *mlpize*(N, C): Convert a set of ordinary atoms N to a partial interpretation $\mathbf{N}$ (having undefined components nil)
  - projecting atoms in N to module instances $P_i[S] \in C$,
  - removing module prefixes, and
  - putting the result at position N_i/S in $\mathbf{N}$
- *ans*(R): Find the answer sets of a set of ordinary rules R.

Top Down Evaluation Algorithm

Input: MLP **P**, set of value calls C, list of sets of value calls *path*, partial model **M**, indexed set of sets of module atoms A, set of answer sets $\mathcal{AS}$

Output: set of answer sets $\mathcal{AS}$

if $\exists P_i[S] \in C$ *s.t.* $P_i[S] \in Cprev$ *for some* $Cprev \in path$ **then**

  if $S = \emptyset$ *for some* $P_i[S] \in C$ **then** return;

  repeat

    $C' := tail(path)$ and remove the last element of *path*;

    if $\exists P_j[T] \in C'$ *s.t.* $T = \emptyset$ **then** return; **else** $C := C \cup C'$;

  until $C' = Cprev$;

end

R := rewrite(C, M, A);

...

Top Down Evaluation Algorithm (2)

if *R is ordinary* **then**
  if *path is empty* **then**
    forall $N \in ans(R)$ **do** $\mathcal{AS} := \mathcal{AS} \cup \{\mathbf{M} \uplus mlpize(N, C)\}$
  else
    C := tail(path) and remove the last element of path;
    forall $P_i[S] \in C$ **do** $A_i/S := fin$;
    forall $N \in ans(R)$ **do**
    $comp(P, C, path, \mathbf{M} \uplus mlpize(N, C), \mathbf{A}, \mathcal{AS})$;
  end
end
else
  ...
end

Top Down Evaluation Algorithm (3)

pick an $\alpha := P_j[p].o(c)$ in R with smallest $ill_R(\alpha)$ and find splitting set U of R for α ;

forall $P_i[S] \in C$ **do**

  if $A_i/S = nil$ **then** $A_i/S := \alpha$ **else** $A_i/S := A_i/S \cup \alpha$;

end

forall $N \in ans(b_U(R))$ **do**

  $T := N|_{\mathbf{p}}^{\mathbf{q_j}}$;

  if $(M_j/T = nil) \wedge (A_j/T = fin)$ **then**

    $C' := C$ and $path' := path$

  else $C := \{P_j[T]\}$ and path:=append(path,C)

  $comp(P, C, path, M \uplus mlpize(N, C), \mathbf{A}, \mathcal{AS})$;

end

Thank you!