An Abductive Paraconsistent Semantics – $MH_P$

Mário António Abrantes$^1$ and Luís Moniz Pereira$^2$

$^1$Instituto Politécnico de Bragança, $^2$CENTRIA, Portugal
INAP13' September, 2013
Introduction: $MH_p = MH + WFSX_p$

1. Introduction: $MH_p = MH + WFSX_p$
2. The $MH$ Abductive Spirit
3. $MH$ Models Computation
4. $WFSX_p$ Semantics
5. $MH_p$ Semantics
6. Conclusion
MHₚ is a semantics for extended normal logic programs whose models are **total** and **paraconsistent**.

By **total** and **partial** Models we mean (normal logic programs case):

\[ P \]
\[
\begin{align*}
  b & \leftarrow \text{not } d \\
  a & \leftarrow \text{not } a
\end{align*}
\]

\[ Q \]
\[
\begin{align*}
  b & \leftarrow \text{not } d \\
  c & \leftarrow
\end{align*}
\]

\[
\text{WFM}(P) = \langle \{b\}^+, \{a\}^u, \{d\}^- \rangle : \text{Partial Model}
\]
\[
\text{WFM}(Q) = \langle \{b, c\}^+, \{\}^u, \{d\}^- \rangle : \text{Total Model}
\]
The necessity of Explicit Negation

This is a classic example (due to John McCarthy). We do not want to cross the railway on basis of lack of a proof the train is coming.

\[
\text{cross} \leftarrow \text{not train}
\]

The adequate way, is to make the train is not coming: we need to be able to assert falsity!

\[
\text{cross} \leftarrow \neg \text{train}
\]
Sometimes **all the information** must be squeezed from a logic program.

For example, in an **emergency** situation,

\[
\begin{align*}
\text{danger} & \leftarrow \text{not run} \\
\text{run} & \leftarrow \text{not safe} \\
\text{safe} & \leftarrow \text{not danger}
\end{align*}
\]

**indecision** may not be acceptable. **Eliminate indecision** enforcing a 2-valued semantics.
Eliminate undecision via a 2-valued semantics.

- Add to $P$ a **minimal set of hypotheses**, $H$, such that $WFM^u(P \cup H) = \emptyset$.
- **Assumable set of hypotheses**: atoms that appear default negated: $\{run, danger, safe\}$.
- For example, $H = \{run\}$:

$$P \cup \{run\}$$

\begin{align*}
\text{danger} & \leftarrow \text{not run} \\
\text{safe} & \leftarrow \text{not danger} \\
\text{run} & \leftarrow \text{not safe} \\
\text{run} & \leftarrow \\
\end{align*}

$$WFM(P \cup \{run\}) = \{run, \text{not danger}, safe\}$$
Four friends are planning a holiday.

- **First friend** says "If we don’t go to Germany, then we must go to Sweden"
  \[ \text{sweden} \leftarrow \text{not germany}. \]
  etc. for the first 3 friends.

- **Fourth friend** says "We must go to Denmark"
  \[ \text{denmark} \leftarrow. \]
  \[ \text{sweden} \leftarrow \text{not germany} \]
  \[ \text{denmark} \leftarrow \text{not sweden} \]
  \[ \text{germany} \leftarrow \text{not denmark} \]
  \[ \text{denmark} \leftarrow \]
There is a single stable model solution.

\[ SM(P) = \{\text{denmark, not germany, sweden} \} \]

But on simple inspection, another solution is devised.

\[ SM(P) = \{\text{denmark, germany, not sweden} \} \]

Both solutions are obtained if we envisage the loop in the program as a choice device, by considering all the default negated atoms as assumable hypotheses.
The **WFM** of a logic program may be computed via the **remainder** of the program.

The **remainder** is computed by transforming the original program using 5 operations: **loop detection**, **failure**, **positive reduction**, **success**, **negative reduction**.

This reduction system is **terminating** and **confluent** for finite ground normal logic programs.
**Example**

- Rules and literals highlighted in program $Q$, below, are eliminated during remainder computation.

  \[
  \text{remainder}(Q) = \hat{Q}
  \]

  \[
  WFM(Q) = WFM(\hat{(Q)}) = \{d, s\} \cup \text{not } \{g, k, u, w\}.
  \]

- $u \leftarrow w$  \hspace{2cm} Loop Detection
- $w \leftarrow u$  \hspace{2cm} Loop Detection
- $k \leftarrow g$  \hspace{2cm} Failure
- $s \leftarrow \text{not } g, d$  \hspace{2cm} Positive Reduction + Success
- $g \leftarrow \text{not } d$  \hspace{2cm} Negative Reduction
- $d \leftarrow \text{not } s$  \hspace{2cm} Negative Reduction
Layered Remainder Computation (1/2)

The layered remainder uses the loops as choice devices. The key to preserve loops is to replace negative reduction by . . .

**layered negative reduction**: Use fact f to eliminate a rule h ← not f iff the rule is not in loop through not f.

The layered remainder is computed by transforming the original program using 5 operations: loop detection, failure, positive reduction, success, layered negative reduction.

The model obtained with the layered remainder is the **layered well-founded model**, LWFM.
Layered Remainder Computation (2/2): example

Example of **layered remainder** computation of program $Q$ below:

- The highlighted rules and literals are eliminated.
- Denote by $\hat{Q}$ the **layered remainder** of $Q$.
- $LWFM(Q) = \{d\} \cup \text{not} \{u, w\}$

\[
\begin{align*}
\text{Loop Detection} & \\
&
\begin{align*}
u & \leftarrow w \\
w & \leftarrow u
\end{align*}
\text{Loop Detection}
\begin{align*}
k & \leftarrow g \\
s & \leftarrow \text{not } g, \ d
\end{align*}
\text{Success}
\begin{align*}
g & \leftarrow \text{not } d \\
d & \leftarrow \text{not } s
\end{align*}
\begin{align*}
d & \leftarrow
\end{align*}
\end{align*}
\]
**MH Models Computation**

- Form the **assumable hypotheses set** of $Q$ (default negated atoms that are not facts in $\hat{Q}$): \{g, s\}.
- Compute all the 2-valued stable models of $Q \cup H$, for all nonempty minimal hypotheses sets $H \subseteq \{g, s\}$ and for $H = \emptyset$.
- $MH$ models of $Q$: \{d, not g, not k, s\} with hypotheses sets $H = \emptyset$ and $H = \{s\}$, and \{d, g, k, not s\} with hypotheses set $H = \{g\}$.

\[
\begin{align*}
\hat{Q} & : k \leftarrow g & s \leftarrow not g & d \leftarrow not s \\
g \leftarrow not d & \\
d \leftarrow &
\end{align*}
\]
Extended logic programs allow two types of negation: default negation \textit{not} $b$ and explicit negation $\neg b$.

\WFSXp: well-founded semantics for extended logic programs.

- Collapses into \WFS for normal logic programs.
- Relates default negation and explicit negation through the coherence principle: if $\neg l$ holds, then not $l$ also does (similarly, if $l$ then not $\neg l$).
- Detects dependencies on contradiction.
Example: WFSXp model

\[ P \]

\[ z \leftarrow \text{not } z \]

\[ \neg a \leftarrow \]

\[ u \leftarrow \text{not } a \]

\[ c \leftarrow \text{not } d \]

\[ \text{WFSXp}(P) = \langle \{ \neg a, c, u \}^+, \{ z, \neg z \}^u, \{ a, \neg c, \neg u \}^- \rangle. \]

\[ \neg a \] and \( u \leftarrow \text{not } a \) render \( u \) true via the coherence principle.
**WFSXp** Semantics (3/4)

**WFSXp** may be embedded into **WFS** by a simple transformation.

- Take an extended program $P$ and compute the $P^{t-o}$ transformed of $P$:

<table>
<thead>
<tr>
<th>$P$</th>
<th>$P^{t-o}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\neg a \leftarrow$</td>
<td>$\neg a \leftarrow$</td>
</tr>
<tr>
<td>$\neg a^o \leftarrow$</td>
<td>$\neg a^o \leftarrow \text{not } a$</td>
</tr>
<tr>
<td>$c \leftarrow \text{not } b$</td>
<td>$c \leftarrow \text{not } b^o$</td>
</tr>
<tr>
<td>$c^o \leftarrow \text{not } b, \text{not } \neg c$</td>
<td></td>
</tr>
<tr>
<td>$u \leftarrow \neg a$</td>
<td>$u \leftarrow \neg a$</td>
</tr>
<tr>
<td>$u^o \leftarrow \neg a^o, \text{not } \neg u$</td>
<td></td>
</tr>
</tbody>
</table>

$\neg a, \neg a^o, \neg c, \neg u$ in $P^{t-o}$ language are names of atoms, not explicit negations. **Bold** literals enforce the coherence **principle**.

- Compute the $\text{WFM}(P^{t-o})$: 
\[ WFM(P^{t-o}) = \langle \{ \neg a, \neg a^o, c, c^o, u, u^o \}^+, \{ \}^u \{ a, a^o, b, b^o, \neg b, \neg b^o, \neg c, \neg c^o, \neg u, \neg u^o \}^- \rangle. \]

- **Read** the \( WFSX_p(P) \) model from \( WFM(P^{t-o}) \)

\[
\begin{align*}
    a \in WFM_p(P) & \iff a \in WFM(P^{t-o}) \\
    \text{not } a \in WFM_p(P) & \iff \text{not } a^o \in WFM(P^{t-o}) \\
    \neg a \in WFM_p(P) & \iff \neg a \in WFM(P^{t-o}) \\
    \text{not } \neg a \in WFM_p(P) & \iff \text{not } \neg a^o \in WFM(P^{t-o})
\end{align*}
\]

\[ WFM_p(P) = \langle \{ \neg a, c, u \}^+, \{ \}^u \{ a, b, \neg b, \neg c, \neg u \}^- \rangle. \]
Computing $MH_P$ Models (1/2)

- Take an extended normal logic program $P$.
- Compute the transformed $P^{t-o}$.
- Compute the balanced layered remainder $bP^{t-o}$ (for preserving loops) by means of the balanced reduction system.

**balanced reduction system**, consists in 5 operations: loop detection, failure, positive reduction, success, **balanced layered negative reduction**.

**balanced layered negative reduction**: Use fact $f^o$ (resp. $f$) to eliminate rule $r = h \leftarrow \text{not } f^o$ (resp. $r^o = h^o \leftarrow \text{not } f$) iff $r, r^o$ are not in loop through $\text{not } f^o, \text{not } f$. 

Mário António Abrantes$^1$ and Luís Moniz Pereira$^2$
Compute the set of **assumable hypotheses** of $P$, $\text{Hyps}(P)$: all the literals $k$ such that $\neg k^o \in bP^{t-o}$ and $k$ is not a fact of $bP^{t-o}$.

MHp models: **total WFSXp models** of programs $P \cup H$, for all nonempty minimal hypotheses sets $H \subseteq \text{Hyps}(P)$ and for $H = \emptyset$. 
Computing $MH_P$ Models: an example (1/2)

- An extended program $P$ and its balanced layered remainder, $bP^{t-o}$

<table>
<thead>
<tr>
<th>$P$</th>
<th>$bP^{t-o}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b \leftarrow h$</td>
<td>$b \leftarrow h$</td>
</tr>
<tr>
<td>$h \leftarrow \neg p$</td>
<td>$h \leftarrow \neg p^o$</td>
</tr>
<tr>
<td>$p \leftarrow \neg b$</td>
<td>$p \leftarrow \neg b^o$</td>
</tr>
<tr>
<td>$b \leftarrow$</td>
<td>$b \leftarrow$</td>
</tr>
<tr>
<td>$\neg h \leftarrow$</td>
<td>$\neg h \leftarrow$</td>
</tr>
</tbody>
</table>

- $b^o \leftarrow h^o, \neg \rightarrow b$
- $h^o \leftarrow \neg p, \neg \rightarrow h$
- $p^o \leftarrow \neg b, \neg \rightarrow p$
- $b^o \leftarrow \neg \rightarrow b$
- $\neg h^o \leftarrow \neg \rightarrow h$

- **Assumable set of hypotheses**, $\text{Hyps}(P) = \{p\}$: $\neg p^o$ appears in $bP^{t-o}$ and $p$ is not a fact.
Computing $MH_P$ Models: an example (2/2)

- $MH_P$ models of $P$:

  $M_1 = \langle \{b, h, \neg h\}^+, \{} \rangle^u, \{\neg b, h, \neg h, p, \neg p\}^- \rangle$
  
  with hypotheses set $H = \emptyset$

  $M_2 = \langle \{b, p, \neg h\}^+, \{} \rangle^u, \{h, \neg b, \neg p\}^- \rangle$
  
  with hypotheses set $H = \{p\}$

- $M_1$ is default inconsistent (e.g. $h$ and not $h$ belong to $M_1$).
- $M_2$ is consistent: is a solution to this variant of the holiday problem.
Introduction: $MH_p = MH + WFSXp$

The $MH$ Abductive Spirit

$MH$ Models Computation

$WFSXp$ Semantics

$MH_p$ Semantics

Conclusion

$MH_p$ is a total models paraconsistent semantics that solves any extended normal logic program.

$MH_p$ models detect dependency on contradiction: objective literals $L$ that are dependent on contradiction exhibit default inconsistency, i.e. both $L$ and $notL$ are in the model.

Computing a $MH_p$ model is a $\Sigma^P_2$ task.

Belief revision, or contradiction removal is treated elsewhere, in MA’s forthcoming PhD thesis.
THANKS!