# ON LOGIC PROGRAM UPDATES

INVITED TALK AT NMR'12

João Leite
CENTRIA, New University of Lisbon

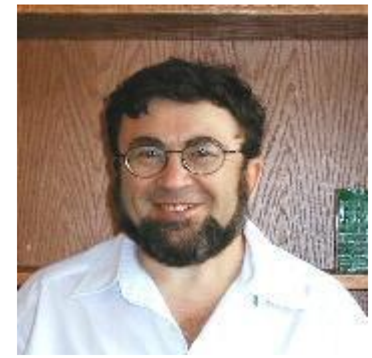
# Non-Monotonic Logic Programming



- Stable Models Semantics: A semantics for logic programs with negation developed by M. Gelfond and V. Lifschitz (1987-91), which lead to:

  Answer-Set Programming (ASP)

- ASP has good properties for Knowledge Representation and Problem Solving
  - expressive language;
  - 0, 1 or multiple answer sets (models);
  - two forms of negation to reason with a limited combination of the closed and open world assumptions;
    - we restrict to default negation.
  - fast answer-set solvers (DLV, CLASP, SMODELS, etc...);
  - theoretically well understood language;



# Logic Programs



- A (generalized) rule r is:

$$L_0 \leftarrow L_1,\ldots, L_n.$$

  - where each $L_i$ is a literal ie. an atom A or default literal ~A.
  - $H(r)= L_0$ is the head of rule r
  - $B(r)=\{L_1,\ldots, L_n\}$ is the body of rule r
- A (generalised) logic program is a set of rules
- Example:

| | |
|---|---|
| $a \leftarrow .$ | $\sim a \leftarrow b.$ |
| $b \leftarrow \sim c.$ | $d \leftarrow \sim e.$ |
| $c \leftarrow \sim b.$ | |

# Why default negation in heads?



- We need a way to update the truth value of an atom to "not being true".
  - In a dynamic setting, updating with a rule

$$A \leftarrow L_1,\ldots, L_n.$$

  means that if $L_1,\ldots, L_n$ is true, then A should now be true
  - while updating with a rule

$$\sim A \leftarrow L_1,\ldots, L_n.$$

  means that if $L_1,\ldots, L_n$ is true, then A should now not be true
- Why not use strong (classical) negation in the head instead?
  - LPs with two kinds of negation allow three different (consistent) states wrt. some atom A, namely $\{A\}$, $\{\neg A\}$ and $\{ \}$.
  - We need to be able to update from/to any of these states
    - Strong negation updates to $\{\neg A\}$
    - Default negation updates to $\{ \}$

# Logic Programs



- An interpretation I is a stable model of a program P if:

$$I' = least(P \cup Defaults)$$

  - $I' = I \cup \{\sim A \mid A \text{ is an atom and } A \notin I\}$
  - $Defaults = \{\sim A \mid A \text{ is an atom and } A \notin I\}$
  - least(.) denotes the least model of the (positive) program obtained by treating literals of the form ~A as new atoms.

- Example:

  $P = \{a. \quad b \leftarrow \sim c. \quad c \leftarrow \sim b. \quad \sim a \leftarrow b. \quad d \leftarrow \sim e.\}$

  $I = \{a, c, d\}$ $\qquad I' = \{a, \sim b, c, d, \sim e\}$ $\qquad Defaults = \{\sim b, \sim e\}$

$least(P \cup Defaults) =$

$= least(\{a.\ b \leftarrow \sim c.\ c \leftarrow \sim b.\ \sim a \leftarrow b.\ d \leftarrow \sim e.\} \cup \{\sim b.\ \sim e.\}) =$

$= \{a, \sim b, c, d, \sim e\} = I'$

$\Rightarrow \{a, c, d\}$ is a stable model.

# Belief Change



- Change operations on monotonic logics have been studied extensively in the area of belief change.
  - rationality postulates for operations play a central role
  - constructive operator definitions correspond to sets of postulates
- two different belief change operations have been distinguished [Katsuno and Mendelzon1991]:
  - Revision
    - recording newly acquired information about a static world
    - characterized by AGM postulates and their descendants
  - Update
    - recording changes in a dynamic world
    - characterized by KM postulates for update

# KM Postulates

[Katsuno and Mendelzon 91]



Postulates (KM 1) – (KM 8)

(KM 1) $\phi \diamond \psi \models \psi$.

(KM 2) If $\phi \models \psi$, then $\phi \diamond \psi \equiv \phi$.

(KM 3) If both $\phi$ and $\psi$ are satisfiable, then $\phi \diamond \psi$ is satisfiable.

(KM 4) If $\phi_1 \equiv \phi_2$ and $\psi_1 \equiv \psi_2$, then $\phi_1 \diamond \psi_1 \equiv \phi_2 \diamond \psi_2$.

(KM 5) $(\phi \diamond \psi) \wedge \chi \models \phi \diamond (\psi \wedge \chi)$.

(KM 6) If $\phi \diamond \psi_1 \models \psi_2$ and $\phi \diamond \psi_2 \models \psi_1$, then $\phi \diamond \psi_1 \equiv \phi \diamond \psi_2$.

(KM 7) $(\phi \diamond \psi_1) \wedge (\phi \diamond \psi_2) \models \phi \diamond (\psi_1 \vee \psi_2)$ if $\phi$ is complete.

(KM 8) $(\phi_1 \vee \phi_2) \diamond \psi \equiv (\phi_1 \diamond \psi) \vee (\phi_2 \diamond \psi)$.

# Logic Program Updates



- Problem

  Assing Semantics to a sequence of Logic Programs:

  $$(P_1, P_2, \ldots, P_n)$$

- …or a Dynamic Logic Program
- Several lines of research
  - Based on *Causal Rejection*
  - Based on *Abduction/Priorities/Preferences*
  - Based on *KM Postulates*
  - Based on *Structural Properties*
  - …

# Fact Updates

[Marek and Truszczynski 98]



- When the initial knowledge is just a set of facts ($I_i$)
  - an interpretation $I_u$ is a Justified Update of $I_i$ by a program Q if
    - $I_u$ is a model of Q
    - There is no other model $I_X$ of Q such that $\Delta(I_X,I_i) \subset \Delta(I_u,I_i)$
- Example:

  $I_i$ = {rain,clowdy}

  Q = ~rain ← play ← ~rain

  $I_u$ = {play,clowdy}
- If the initial program is just a set of facts, then the result of updating it should be like in Fact Updates.
- $\mathbf{P}_{\iota\upsilon}$: Generalisation of Fact Updates

  $$P_I=\{A \leftarrow \mid A \in I\} \Rightarrow SEM(P_I \oplus Q) = IU(I,Q)$$

# Program Updates

[L and Pereira 98]



- What if our initial KB is a Logic Program?
- Can we simply take each of its stable models and update it?
- Initial Program P:

  sleep ← ~tv_on.

  tv_on.

  watch_tv ← tv_on.
- Update Program U:

  power_failure.

  ~tv_on ← power_failure.
- Stable Model:

  {tv_on, watch_tv}
- Updated Model:

  {power_failure, watch_tv}
- Intended Model is {power_failure, sleep}!

# Support/Causal Rejection

[L and Pereira 98]



- Truth value of any element should be supported by some rule (either from the update program or from the initial program).
- **P**$_\sigma$: Support:

  if $a \in M$ then $\exists r \in P_i$ , $H(r)=a \wedge M \vDash B(r)$

- Inertia should be exerted on the program rules instead of model literals.
- Inertia in rules should only be blocked (or rules rejected) if there is a newer directly conflicting rule (or cause).
- **P**$_\gamma$: Causal Rejection:

  if $M \nvDash r \in P_j$ then $\exists r' \in P_k$ ,$j<k$, $H(r)=\sim H(r') \wedge M \vDash B(r')$

# Other Desirable Properties



**$P_{\nu}$** : Primacy of new information

$$M \in SEM(P \oplus Q) \Rightarrow M \vDash Q$$

**$P_{\emptyset}$** : Immunity to empty updates

$$SEM(P \oplus \emptyset) = SEM(\emptyset \oplus P) = SEM(P)$$

**$P_{\tau}$** : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where $\tau$ is any tautology i.e. any rule $\tau$ such that $H(\tau) \in B(\tau)$

**$P_{\rho\varepsilon}$** : Refined Extension Principle

Generalisation of **$P_{\tau}$** to certain circular updates.

# Logic Program Updates



- An interpretation I is a stable model of $(P_1,\ldots,P_n)$ if

$$I' = least( [\bigcup(P_i) - Reject(I)] \cup Defaults(I) )$$

# DLP – Justified Updates

[L and Pereira 98]



- Interpretation I is a Justified Update of $(P_1,\ldots,P_n)$ if

$$I' = least(\ [\bigcup(P_i) - Reject(I)] \cup Defaults(I)\ )$$

$$Reject(I) = \{r \in P_i \mid \exists r' \in P_j\ ,i<j,\ H(r)=\sim H(r') \wedge I \vDash B(r')\}$$

$$Defaults(I) = \{\sim A \mid A \text{ is an atom and } A \notin I\}$$

# DLP – Justified Updates

[L and Pereira 98]



- Interpretation I is a Justified Update of $(P_1,\ldots,P_n)$ if

$$I' = least(\ [\bigcup(P_i) - Reject(I)] \cup Defaults(I)\ )$$
$$Reject(I) = \{r \in P_i \mid \exists r' \in P_j\ ,i<j,\ H(r)={\sim}H(r') \wedge I \vDash B(r')\}$$
$$Defaults(I) = I^-$$

- Initial Program $P_1$:

  sleep ← ~tv_on.

  tv_on.

  watch_tv ← tv_on.

- Update Program $P_2$:

  power_failure.

  ~tv_on ← power_failure.

Intended Model I={sleep, power_failure}

I'={sleep, power_failure, ~tv_on, ~watch_tv}

Reject(I) = {tv_on.}

Defaults(I) = {~tv_on,~watch_tv}

least{sleep ←~tv_on. watch_tv←tv_on.

power_failure. ~tv_on ←power_failure. ~tv_on. ~watch_tv.} =

={sleep, power_failure, ~tv_on, ~watch_tv} = I'

# DLP – Justified Updates

[L and Pereira 98]



- Properties:

**P**$_\varnothing$ : Immunity to empty updates

$$SEM(P \oplus \varnothing) = SEM(\varnothing \oplus P) = SEM(P)$$

**P**$_\nu$ : Primacy of new information

$$M \in SEM(P \oplus Q) \Rightarrow M \vDash Q$$

**P**$_\sigma$ : Support

$$A \in M \in SEM(P \oplus Q) \Rightarrow \exists\, r \in (P \cup Q) : H(r)=A \wedge M \vDash B(r)$$

# DLP – Justified Updates

[L and Pereira 98]



- But, it doesn't obey:

$\mathbf{P}_\tau$ : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where $\tau$ is any tautology i.e. any rule $\tau$ such that $H(\tau) \in B(\tau)$

# DLP – Justified Updates

[L and Pereira 98]



- But, it doesn't obey:

**$P_\tau$** : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where

**Justified Updates and Immunity to Tautologies**

| | | | |
|---|---|---|---|
| $P_1$: a.<br>$P_2$: ~a ← ~a. | Intended<br>{a} | Obtained<br>{a} and { } | ✘ |
| $P_1$: a.<br>~a.<br>$P_2$: a ← a. | Intended<br>-no models- | Obtained<br>{a} | ✘ |

# DLP – Justified Updates

[L and Pereira 98]



- And, it also doesn't obey:

**$P_{\iota\upsilon}$** : Generalisation of Fact Updates

$$P_I = \{A \leftarrow \mid A \in I\} \Rightarrow SEM(P_I \oplus Q) = IU(I,Q)$$

# DLP – Justified Updates

[L and Pereira 98]



- And, it also doesn't obey:

**$P_{\iota\upsilon}$** : Generalisation of Fact Updates

$$P_I=\{A \leftarrow \mid A\in I\} \Rightarrow SEM(P_I \oplus Q) = IU(I,Q)$$

**Justified Updates and Generalisation of Fact Updates**

| | Intended | Obtained | |
|---|---|---|---|
| $P_1$: a.<br>$P_2$: ~a ← ~a. | {a} | {a} and { } | ✘ |

# DLP – Dynamic Stable Models

[Alferes, L, Pereira, Przymusinska and Przymusinski 98,00]



- Interpretation I is a Dynamic Stable Model of $(P_1, \ldots, P_n)$ if

$$I' = least( [\bigcup(P_i) - Reject(I)] \cup Defaults(I) )$$

$$Reject(I) = \{r \in P_i \mid \exists r' \in P_j, i<j, H(r) = \sim H(r') \wedge I \vDash B(r')\}$$

$$Defaults(I) = \{\sim A \mid \nexists r \in P_i, H(r) = A \wedge I \vDash B(r)\}$$

# DLP – Dynamic Stable Models

[Alferes, L, Pereira, Przymusinska and Przymusinski 98,00]



- Properties:

**P**$_{\varnothing}$ : Immunity to empty updates

$$SEM(P \oplus \varnothing) = SEM(\varnothing \oplus P) = SEM(P)$$

**P**$_{\nu}$ : Primacy of new information

$$M \in SEM(P \oplus Q) \Rightarrow M \vDash Q$$

**P**$_{\sigma}$ : Support

$$A \in M \in SEM(P \oplus Q) \Rightarrow \exists\, r \in (P \cup Q) : H(r)=A \wedge M \vDash B(r)$$

**P**$_{\iota\upsilon}$ : Generalisation of Fact Updates

$$P_I=\{A \leftarrow | A \in I\} \Rightarrow SEM(P_I \oplus Q) = IU(I,Q)$$

# DLP – Dynamic Stable Models

[Alferes, L, Pereira, Przymusinska and Przymusinski 98,00]



- But, it doesn't obey:

**$P_\tau$** : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where $\tau$ is any tautology i.e. any rule $\tau$ such that $H(\tau) \in B(\tau)$

# DLP – Dynamic Stable Models

[Alferes, L, Pereira, Przymusinska and Przymusinski 98,00]



- But, it doesn't obey:

**$P_\tau$** : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where τ

### Dynamic Stable Models and Immunity to Tautologies

| | Intended | Obtained | |
|---|---|---|---|
| $P_1$: a.<br>$P_2$: $\sim a \leftarrow \sim a$. | {a} | {a} | ✓ |
| $P_1$: a.<br>$\sim a$.<br>$P_2$: $a \leftarrow a$. | -no models- | {a} | ✘ |

# DLP – Dynamic Stable Models

[Alferes, L, Pereira, Przymusinska and Przymusinski 98,00]



- And it also doesn't obey:

$\mathbf{P}_{\rho\varepsilon}$ : Refined Extension Principle

Generalisation of $\mathbf{P}_{\tau}$ to certain circular updates.

# DLP – Dynamic Stable Models

[Alferes, L, Pereira, Przymusinska and Przymusinski 98,00]

- And it

**P**$_{\rho\varepsilon}$ : Refin

es.

Dynamic Stable Models and Refined Extension Principle

| Program | Intended | Obtained | |
|---|---|---|---|
| $P_1$: a ← ~b.<br>b ← ~a.<br>c ← b.<br>~c.<br>$P_2$: c ← c. | {a} | {a} and {b,c} | ✘ |
| $P_1$: a ← ~b.<br>b ← ~a.<br>c ← b.<br>~c.<br>$P_2$: c ← e.<br>e ← c. | {a} | {a} and {b,c,e} | ✘ |

# DLP – Refined Dynamic Stable Models

[Alferes, Banti, Brogi and L 05]



- Interpretation I is a Refined Dynamic Stable Model of $(P_1,\ldots,P_n)$ if

$$I' = \mathrm{least}\left( \left[\bigcup(P_i) - \mathrm{Reject}(I)\right] \cup \mathrm{Defaults}(I) \right)$$

$$\mathrm{Reject}(I) = \{r \in P_i \mid \exists r' \in P_j, i \leq j, H(r) = \sim H(r') \wedge I \vDash B(r')\}$$

$$\mathrm{Defaults}(I) = \{\sim A \mid \nexists r \in P_i, H(r) = A \wedge I \vDash B(r)\}$$

# DLP – Refined Dynamic Stable Models

[Alferes, Banti, Brogi and L 05]



- Interpretation I is a Refined Dynamic Stable Model of $(P_1,\ldots,P_n)$ if

$$I' = least(\ [\bigcup(P_i) - Reject(I)] \cup Defaults(I)\ )$$

$$Reject(I) = \{r \in P_i \mid \exists r' \in P_j\ , i \leq j,\ H(r) = \sim H(r') \wedge I \vDash B(r')\}$$

$$Defaults(I) = \{\sim A \mid \nexists r \in P_i\ ,\ H(r) = A \wedge I \vDash B(r))\}$$

- Initial Program $P_1$:

  a.

  ~a.

- Update Program $P_2$:

  $a \leftarrow a$

Unintended Model $I=\{a\}$

$Reject(I) = \{a.\ \sim a.\}$

$Defaults(I) = \{\ \}$

$least\{a \leftarrow a.\} = \{\ \} \neq I' = \{a\}$

# DLP – Refined Dynamic Stable Models

[Alferes, Banti, Brogi and L 05]



- Properties:

**$P_\emptyset$** : Immunity to empty updates

$$SEM(P \oplus \emptyset) = SEM(\emptyset \oplus P) = SEM(P)$$

**$P_\tau$** : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where $\tau$ is any tautology i.e. any rule $\tau$ such that $H(\tau) \in B(\tau)$

**$P_\nu$** : Primacy of new information

$$M \in SEM(P \oplus Q) \Rightarrow M \vDash Q$$

**$P_\sigma$** : Support

$$A \in M \in SEM(P \oplus Q) \Rightarrow \exists r \in (P \cup Q) : H(r)=A \wedge M \vDash B(r)$$

**$P_{\iota\upsilon}$** : Generalisation of Fact Updates

$$P_I=\{A \leftarrow \mid A \in I\} \Rightarrow SEM(P_I \oplus Q) = IU(I,Q)$$

**$P_{\rho\varepsilon}$** : Refined Extension Principle

Generalisation of **$P_\tau$** to certain circular updates.

# DLP – Dynamic Answer Sets

[Eiter, Fink, Sabbatini and Tompits 02]



- Interpretation I is a Dynamic Answer Set of $(P_1,\ldots,P_n)$ if

$$I' = least( [\bigcup(P_i) - Reject(I)] \cup Defaults(I) )$$

$$Reject(I)=\{r\in P_i \mid \exists r'\in P_j \setminus Reject(I), i<j, H(r)=\sim H(r') \wedge I \vDash B(r')\}$$

$$Defaults(I) = I^-$$

# DLP – Dynamic Answer Sets

[Eiter, Fink, Sabbatini and Tompits 02]



- It doesn't obey:

**P**$_\tau$ : Immunity to tautologies

$$SEM(P \oplus Q) = SEM(P \oplus (Q \cup \{\tau\})) = SEM((P \cup \{\tau\}) \oplus Q)$$

where $\tau$ is any tautology i.e. any rule $\tau$ such that $H(\tau) \in B(\tau)$

# DLP – Dynamic Answer Sets

[Eiter, Fink, Sabbatini and Tompits 02]



- It doesn't obey:

**$P_\tau$** : Immunity to tautologies

SE... ) ⊕ Q)

where

Dynamic Answer-Sets and Generalisation of Fact Updates and Immunity to Tautologies

| Program | Intended | Obtained | |
|---|---|---|---|
| $P_1$: a.<br>$P_2$: ~a ← ~a. | {a} | {a} and { } | ✘ |
| $P_1$: a.<br>$P_2$: ~a.<br>$P_3$: a ← a. | { } | { } and {a} | ✘ |

# Relationship between Semantics



**Dynamic Answer Sets**

$\supseteq$

**Justified Updates**

$\supseteq$

**Dynamic Stable Models**

$\supseteq$

**Refined Dynamic Stable Models**

They all coincide for acyclic LPs [Homola04,Banti et al. 05]

# Summary of Properties



| | $P_{\varnothing}$ Immunity to empty updates | $P_{\tau}$ Immunity to tautologies | $P_{\nu}$ Primacy of new information | $P_{\sigma}$ Support | $P_{\iota\upsilon}$ Generalisation of Fact Updates | $P_{\rho\varepsilon}$ Refined Extension Principle |
|---|---|---|---|---|---|---|
| Justified Updates | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ |
| Dynamic Stable Models | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ |
| Dynamic Answer Sets | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ |
| Refined Dynamic Stable Models | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

# Other Approaches



- Preference-based Semantics
  - Program Updates through Priorities
    - Zhang 06
  - Program Updates through Preferences
    - Delgrande et al 07.
  - Revision Semantics
    - Delgrande 10
- Abduction-based Semantics
  - Sakama and Inoue 03
- Using structural properties
  - Krumpelmann and Kern-Isberner 10
  - Sefranek 06

# Program Updates Through Priorities

[Zhang 06]



- Updates through a complex mixture of:
  - Fact Updates
  - Logic Programs with Priorities.
- To determine $P \oplus Q$:
  - For each Stable Model M of P, determine M'=IU(M,Q)
  - Determine a maximal subset of P, P', coherent with M'.
  - Define a prioritised Logic Program (P',Q) with Q>P'
  - Finally, determine the reducts of (P',Q) which are the result of updating P with Q.

# Program Updates Through Priorities

[Zhang 06]



- Updates through a complex mixture of:
  - Fact Updates
  - Logic Programs with Priorities.
- To determine $P \oplus Q$:
  - For each Stable Model M of P, determine M'=IU(M,Q)
  - Determine a maximal subset of P, P', coherent with M'.
  - Define a prioritised Logic Program (P',Q) with Q>P'
  - Finally, determine the reducts of (P',Q) which are the result of updating P with Q.

# Program Updates Through Priorities

[Zhang 06]



- Updates through a complex mixture of:
  - Fact Updates
  - Logi
- To de
  - For … =IU(M,Q)
  - Dete … with M'.
  - Defi… with Q>P'
  - Finally, determine the reducts of (P',Q) which are the result of updating P with Q.

Program Updates Through Priorities [Zhang06] and Immunity to Tautologies

| | Intended | Obtained | |
|---|---|---|---|
| $P_1$: $a \leftarrow \sim\neg a.$ <br> $\neg a \leftarrow \sim a.$ <br> $P_2$: $a \leftarrow a.$ | {a} and {¬a} | {a} | ✗ |

# Program Updates Through Priorities

[Zhang 06]



- Updates through a complex mixture of:
  - Fact Updates
  - Logi
- To de
  - For … =IU(M,Q)
  - Det … with M'.
  - Def … Q>P'
  - Finally, determine the reducts of (P',Q) which are the result of updating P with Q.

**Program Updates Through Priorities [Zhang06] and Undetected Conflicts**

| $P_1$: $a \leftarrow c.$ <br> $b \leftarrow c.$ <br> $P_2$: $c.$ <br> $\neg a \leftarrow b.$ | Intended <br> $\{\neg a, b, c\}$ | Obtained <br> -no models- | ✗ |
|---|---|---|---|

# Program Updates Through Preferences

[Delgrande, Schaub and Tompits 07]



- Updates through a mixture of:
  - Preferences
  - Defeasible Rules
- The update models of $P \oplus Q$ are the preferred models of a prioritised Logic Program $(\Pi, <)$ constructed in one of three possible ways (three different operators):
  - $(P^d \cup Q^d, P^d \times Q^d)$
  - $(P^d \cup Q^d, C(P^d, Q^d))$
  - $(c(P \cup Q)^d \cup ((P \cup Q) \setminus c(P \cup Q)), C(P^d, Q^d))$

  where
  - $P^d$ stands for the defeasible version of $P$ (i.e. obtained from adding $\sim\neg head(r)$ to the body of every rule $r$).
  - $C(P \times Q)$ stands for pairs of rules with conflicting heads
  - $c(P \cup Q)$ stands for the rules in $C(P \times Q)$.

# Program Updates Through Preferences

[Delgrande, Schaub and Tompits 07]



- Updates through a mixture of:
  - Preferences
  - Defeasible Rules
- The update models of P⊕Q are the preferred models of a prioritis … ree possible ways (tl …
  - (Pd∪ …
  - (Pd∪ …
  - (c(P∪ …

  where
  - Pd sta … adding ~¬head(r) to the body of every rule r).
  - C(P×Q) stands for pairs of rules with conflicting heads
  - c(P∪Q) stands for the rules in C(P×Q).

**Updates Through Preferences [Delgrande et al. 07] and Default Assumptions**

| | Intended | Obtained | |
|---|---|---|---|
| $P_1$: ¬a. <br> $P_2$: a ← ~¬a. | {¬a} | {a} | ✗ |

# Revision Semantics

[Delgrande 10]



- Updates are determined by:
  - Taking the most recent program and committ to a maximal set of default assumptions (default literals) needed to build one of its answer-sets.
  - Then, add a maximal coherent sub-set of rules of the predecessor program, and committ to more default asumptions
  - …

# *Revision* Semantics

[Delgrande 10]



- Updates are determined by:
  - Taking the most recent program and committ to a maximal set of default assumptions (default literals) nee
  - Ther … s of the pre … efault asur
  - …

**Revision Semantics [Delgrande10] and Default Assumptions**

| | Intended | Obtained | |
|---|---|---|---|
| $P_1$: a.<br>$P_2$: b ← ~a. | {a} | {b} | ✘ |

# Program Updates Through Abduction

[Sakama and Inoue 03]



- Updates through Abduction:
- $P' = (P \cup Q) \setminus R$ is the result of $P \oplus Q$ if:
  - $SEM(P') \neq \emptyset$
  - $R \subseteq P$
  - $\nexists R' \subset R \mid SEM((P \cup Q) \setminus R') \neq \emptyset$
- Main Problem – fails even the most basic property

  $\mathbf{P}_{\emptyset}$ : Immunity to empty updates

  $$SEM(P \oplus \emptyset) = SEM(\emptyset \oplus P) = SEM(P)$$
- Other issues:
  - Commits to rejected rules (R), which cannot be reused.
  - The result can be more than one program.
  - Higher Computational Complexity.

# Other Properties



$\mathbf{P}_{\mu\rho\rho}$ : Minimal Rule Rejection

$$SEM(P \cup Q) \neq \emptyset \Rightarrow SEM(P \oplus Q) = SEM(P \cup Q)$$

$\mathbf{P}_{\omega\mu\chi}$ : Weak Minimal Change

$$SEM(P \cup Q) \neq \emptyset \Rightarrow SEM(P \oplus Q) \subseteq SEM(P \cup Q)$$

$\mathbf{P}_{\upsilon\rho}$ : Universal Recoverability Principle

$$\forall P\ \exists Q : SEM(P \oplus Q) \neq \emptyset$$

# Summary of Properties



| | $P_{\varnothing}$ | $P_{\tau}$ | $P_{\nu}$ | $P_{\sigma}$ | $P_{\iota\upsilon}$ | $P_{\rho\varepsilon}$ | $P_{\mu\rho\rho}$ | $P_{\omega\mu\chi}$ | $P_{\upsilon\rho}$ |
|---|---|---|---|---|---|---|---|---|---|
| | Immunity to empty updates | Immunity to tautologies | Primacy of new information | Support | Generalisation of Fact Updates | Refined Extension Principle | Minimal Rule Rejection | Weak Minimal Change | Universal Recoverability Principle |
| Justified Updates | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ ✓ | ✓ | ✓ |
| Dynamic Stable Models | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ ✓ | ✓ | ✓ |
| Dynamic Answer Sets | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ ✓ | ✓ | ✓ |
| Refined Dynamic Stable Models | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ ✓ | ✓ | ✓ |
| LP Updates through Abduction | ✗ | ✗ | ✓ | ✓ | ✓ | - | ✓ | ✓ | ✓ |
| LP Updates through Priorities | ✓ | ✗ | ✓ | ✓ | ✓ | - | ✗ | ✗ | ✗ |
| LP Updates through Preferences[1,2] | ✗ | ✗ | ✓ | ✓ | ✗ | - | ✗ | ✗ | - |
| LP Updates through Preferences[3] | ✓ | ✗ | ✓ | ✓ | ✗ | - | ✗ | ✗ | - |
| Revision Semantics | ✗ | ✗ | ✓ | ✓ | ✗ | - | ✗ | ✗ | ✓ |

# What about Classical Belief Change?



- Directly applying the KM postulates and constructions from belief change to logic programs and answer-sets leads to a number of serious problems.
  - ambiguity of the postulates, often difficult to formulate for logic programs and answer-sets
  - leads to very counterintuitive results
  - at the heart of [Leite and Pereira 98] and thoroughly investigated in [Eiter, Fink, Sabbatini and Tompits 02]
- Reconciliation of belief change with rule evolution is still a very interesting open problem:
  - a more general understanding of knowledge evolution
  - a semantic approach to rule evolution, focusing only on the meaning of a logic program and not on its syntactic representation
- How to proceed?

# Belief Change and SE Models



- Belief Change on SE Models
  - AGM Revision on SE Models
    - [Delgrande, Schaub, Tompits and Woltran 08]
- SE Models [Turner 03]
  - semantic characterisation of logic programs, coinciding with the models in the Logic of Here and There for the fragment corresponding to logic programs.
  - richer structure – an SE interpretation X is a pair of ordinary interpretations $\langle I,J \rangle$ such that $I \subseteq J$
  - an interpretation $\langle I,J \rangle$ is an SE-model of a program P if J is a model of P and I is a model of $P^J$ (the GL reduct of P by I)
  - monotonic and more expressive than answer sets
  - characterise strong equivalence

# KM Updates and SE Models

[Slota and L 10]



Postulates (PU 1) – (PU 8)

(PU 1) $\mathcal{P} \oplus \mathcal{Q} \models_s \mathcal{Q}$.

(PU 2) If $\mathcal{P} \models_s \mathcal{Q}$, then $\mathcal{P} \oplus \mathcal{Q} \equiv_s \mathcal{P}$.

(PU 3) If both $\mathcal{P}$ and $\mathcal{Q}$ are satisfiable, then $\mathcal{P} \oplus \mathcal{Q}$ is satisfiable.

(PU 4) If $\mathcal{P}_1 \equiv_s \mathcal{P}_2$ and $\mathcal{Q}_1 \equiv_s \mathcal{Q}_2$, then $\mathcal{P}_1 \oplus \mathcal{Q}_1 \equiv_s \mathcal{P}_2 \oplus \mathcal{Q}_2$.

(PU 5) $(\mathcal{P} \oplus \mathcal{Q}) \dot{\wedge} \mathcal{R} \models_s \mathcal{P} \oplus (\mathcal{Q} \dot{\wedge} \mathcal{R})$.

(PU 6) If $\mathcal{P} \oplus \mathcal{Q}_1 \models_s \mathcal{Q}_2$ and $\mathcal{P} \oplus \mathcal{Q}_2 \models_s \mathcal{Q}_1$, then $\mathcal{P} \oplus \mathcal{Q}_1 \equiv_s \mathcal{P} \oplus \mathcal{Q}_2$.

(PU 7) $(\mathcal{P} \oplus \mathcal{Q}_1) \dot{\wedge} (\mathcal{P} \oplus \mathcal{Q}_2) \models_s \mathcal{P} \oplus (\mathcal{Q}_1 \dot{\vee} \mathcal{Q}_2)$ if $\mathcal{P}$ is basic.

(PU 8) $(\mathcal{P}_1 \dot{\vee} \mathcal{P}_2) \oplus \mathcal{Q} \equiv_s (\mathcal{P}_1 \oplus \mathcal{Q}) \dot{\vee} (\mathcal{P}_2 \oplus \mathcal{Q})$.

# KM Updates and SE Models

[Slota and L 10]



- Construction:

  $\omega$ - assigns a partial order $\leq^X_\omega$ to every SE interpretation X

  $$[\![P \oplus Q]\!]^{SE} = \bigcup_{X \in [\![P]\!]^{SE}} \min([\![Q]\!]^{SE}, \leq^X_\omega) \qquad (1)$$

- Representation Theorem: A program update operator $\oplus$ satisfies conditions (PU 1) – (PU 8) if and only if there exists a faithful and organised SE partial order assignment $\omega$ such that (1) is satisfied for all programs P;Q.
- We also defined a concrete operator.
- Great!
- But…

# Problem with SE Model Update

[Slota and L 10]



- Theorem A program update operator that satisfies (PU4) either does not respect support or it does not respect fact update.
- Proof
  - Let $\oplus$ be a program update operator that satisfies PU4 and let:

    P1: p. q.    P2: p⟵q. q.    Q: ~q.

  - Since $P_1 \equiv_S P_2$, by (PU4) we have that $P_1 \oplus Q \equiv_S P_2 \oplus Q$. Consequently, $P_1 \oplus Q$ has the same answer sets as $P_2 \oplus Q$.
  - Since $\oplus$ respects fact update, then $P_1 \oplus Q$ has the unique answer set {p}.
  - But then {p} is an answer set of $P_2 \oplus Q$ in which p is unsupported by $P_2 \cup Q$.
  - Hence $\oplus$ does not respect support.

# How to Proceed?

[Slota and L 11]



- Three ways to proceed:
  - abandon the classical postulates and constructions
  - use existing approaches (with a syntactic flavour)
    - Refined Dynamic Stable Models
  - find a more expressive characterisation of logic programs

# How to Proceed?

[Slota and L 11]



- Our idea:
  - View a Program as the Set of Sets of SE models of the rules it is composed of.

$P_1$: {r. s.} viewed as { {⟨r,r⟩, ⟨r,rs⟩ ⟨rs,rs⟩}, {⟨s,s⟩, ⟨s,rs⟩, ⟨rs,rs⟩} }

$P_2$: {r ← s. s.} viewed as { {⟨Ø,Ø⟩, ⟨Ø,r⟩ ⟨r,r⟩, ⟨Ø,rs⟩, ⟨r,rs⟩ ⟨rs,rs⟩}, {⟨s,s⟩, ⟨s,rs⟩, ⟨rs,rs⟩} }

- Closer to Base Change
- But…

$P_1$: ~a ← b. $P_2$: ~b ← a. $P_3$: ← a,b.

…are all SE-Equivalent because their rules are not distinguishable by the SE-models semantics!

…and we want to distinguish their effect when used to update the program {a. b.}

# How to proceed?



- Three ways to proceed:
  - abandon the classical postulates and constructions
  - use existing approaches (with a syntactic flavour)
    - Refined Dynamic Stable Models
  - find a more expressive characterisation of logic programs …
    - …not based on the Logic of Here and There (… and SE-models)!

# RE-Models

[Slota and L 12]



- An interpretation <I,J> is an RE-model of a program P if I is a model of $P^J$.
- Distinguishes

$P_1$: ~a ← b. $P_2$: ~b ← a. $P_3$: ← a,b.

- Viewing a program as the set of sets of RE-models of its rules
- … we defined an update operator that coincides with Justified Updates (apart from programs with local cycles).
- It can be seen as a semantic counterpart of Justified Updates.
- More about this at KR'12
  - Wednesday at 14:00 – Belief Revision II Session

# Conclusions/Open Problems



- Semantic counterpart of Refined Dynamic Stable Models.
- Other notions of equivalence, instead of the one based on RE-Models, that allow us to satisfy some additional KM postulates.
  - Difficulty resides in capturing non-tautological irrelevant updates [Alferes et al 05, Sefranek 06].
- Better understanding of differences between Revision and Update in Logic Programming.
- Postulates for Updates of LPs
  - Although we should proceed with caution…

# Conclusion…



The journey isn't over…

… but we are getting there.

# References




- J. J. Alferes, J. A. Leite , L. M. Pereira, H. Przymusinska and T. C. Przymusinski, Dynamic Logic Programming, In Procs of KR'98, Morgan Kaufmann Publishers, 1998.
- J. J. Alferes, J. A. Leite, L. M. Pereira, H. Przymusinska, and T. C. Przymusinski. Dynamic updates of non-monotonic knowledge bases. The Journal of Logic Programming, 45(1-3):43–70, September/October 2000.
- J. J. Alferes, F. Banti, A. Brogi, and J. A. Leite. The refined extension principle for semantics of dynamic logic programming. Studia Logica, 79(1): 7–32, 2005.
- F. Banti, J. J. Alferes, A.o Brogi, and P. Hitzler: The Well Supported Semantics for Multidimensional Dynamic Logic Programs, In Procs. of LPNMR'05, vol 3662 of LNAI, Springer, 2005.
- J. P. Delgrande. A Program-Level Approach to Revising Logic Programs under the Answer Set Semantics. Theory and Practice of Logic Programming, 10(4-6):565–580, July 2010.
- J. P. Delgrande, T. Schaub, and H. Tompits. A preference-based framework for updating logic programs. In Procs. of LPNMR'07, vol. 4483 of LNAI, Springer, 2007.
- J. P. Delgrande, T. Schaub, H. Tompits, and S. Woltran. Belief revision of logic programs under answer set semantics. In Procs. of KR'08, AAAI Press, 2008.

# References




- T. Eiter, M. Fink, G. Sabbatini, and H. Tompits. On properties of update sequences based on causal rejection. Theory and Practice of Logic Programming, 2(6):721–777, 2002.
- M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In Procs. of ICLP'88, MIT Press, 1988.
- M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. New Generation Computing, 9(3-4):365–385, 1991.
- M. Homola. Dynamic logic programming: Various semantics are equal on acyclic programs. In Procs. of CLIMA V, vol. 3487 of LNAI, Springer, 2004.
- H. Katsuno and A. O. Mendelzon. On the difference between updating a knowledge base and revising it. In Procs of KR'91, Morgan Kaufmann Publishers, 1991.
- P. Krümpelmann and G. Kern-Isberner. On belief dynamics of dependency relations for extended logic programs. In Procs. of NMR'10, 2010.
- J. A. Leite. Evolving Knowledge Bases, volume 81 of Frontiers of Artificial Intelligence and Applications, IOS Press, 2003.
- J. A. Leite and L. M. Pereira. Generalizing updates: From models to programs. In Procs. of LPKR'97, vol. 1471 of LNAI, Springer, 1998.

# References




- V. W. Marek and M. Truszczynski. Revision programming. Theoretical Computer Science, 190(2):241–277, 1998.
- C. Sakama and K. Inoue. An abductive framework for computing knowledge base updates. Theory and Practice of Logic Programming, 3(6):671–713, 2003.
- J. Šefránek. Irrelevant updates and nonmonotonic assumptions. In Procs. of JELIA'06, vol. 4160 of LNAI, Springer, 2006.
- M. Slota and J. Leite. On semantic update operators for answer-set programs. In Procs. of ECAI'10, volume 215 of Frontiers of Artificial Intelligence and Applications, IOS Press, 2010
- M. Slota and J.o Leite. Robust equivalence models for semantic updates of answer-set programs. In Procs. of KR'12, AAAI Press, 2012
- H. Turner. Strong equivalence made easy: nested expressions and weight constraints. Theory and Practice of Logic Programming, 3(4-5):609–622, 2003.
- Y. Zhang. Logic program-based updates. ACM Transactions on Computational Logic, 7(3):421–472, 2006.