Semantic Considerations on Rejection

Ján Šefránek
Institute of Informatics, Comenius University, Bratislava, Slovakia,
e-mail: sefranek@fmph.uniba.sk

Abstract

Rejection of some beliefs when a more faithful information is acquired belongs among fundamental features of human (nonmonotonic) reasoning. However, this feature has been noticed usually more as a side effect than as a subject of value for a theoretical investigation.

Recently rejections of rules attracted an attention of researchers in logic program updates. An important insight has been obtained thanks to dynamic logic programming paradigm: set of models is not a sufficient conceptual basis for a semantic characterization of logic program updates. The main idea can be characterized by the causal rejection principle: if there is a conflict between rules, then more preferred rules override those less preferred.

We argue that it is not sufficient to be focused only on the conflicts between rules. Meaning of a nonmonotonic knowledge base is determined both by rules and by nonmonotonic assumptions. Therefore, also conflicts between assumptions (beliefs) are important. We are turning back to a more semantic approach. However, it is needed to include also dependencies between interpretations into the semantics.

A Kripkean semantics of logic program updates is presented in this paper. The semantics enables to record the dependencies between interpretations. An update operation on Kripke structures is defined. The main idea of the update operation consists in rejecting some dependencies between interpretations. The approach is evaluated with respect to well known troubles with irrelevant updates in the approaches based on the causal rejection principle.

Introduction

Background Dynamic aspects of knowledge representations didn't attract an adequate attention of logic programming community for a long time. Recently the problem has been tackled by a variety of the approaches, see (Alferes and Pereira 1996; Alferes et al. 2000; Eiter et al. 2002; Leite and Pereira 1997; Leite and Pereira 1998; Leite, Alferes and Pereira 2001; Marek and Truszczynski 1998; Przymusinski and Turner 1997) and others.

An important turn in understanding logic program updates has been based on the observation that "not all the information borne by a logic program is contained within its set of models" (Leite and Pereira 1997). The research in the field shifted its focus from interpretation updates to program updates. The main attention is devoted to the conflicts between rules. They are resolved according to the causal rejection principle, used in most important approaches to logic program updates, see (Alferes et al. 2000; Eiter et al. 2002; Leite 2003) and others: if there is a conflict between rules, then more preferred rules override those less preferred.

Problem Causal rejection principle does not apply to all cases relevant for updates of logic program (and nonmonotonic knowledge bases). There are some conflicts between programs that are not identifiable on the level of conflicts between rules.

On the other hand, the causal rejection principle applies with respect to some irrelevant cases and produces undesirable consequences.

Proposed solution Meaning of a nonmonotonic knowledge base is determined both by rules and by nonmonotonic assumptions. Dependencies on assumptions are relevant from the semantical point of view and conflicts between those dependencies are relevant for updates. We need a semantics which is sensitive both to conflicts between rules and to conflicts between belief sets. Our main goal is a more general semantic characterization of rejections.

The semantics presented in this paper is focused on rejection of some dependencies between belief sets. A more rich semantic structure is needed, in order to reach the goal described above. Our semantic characterization of logic programs records dependencies between belief sets in terms of the accessibility relation in a sort of Kripke structure. We believe that it is an appropriate structure, which is able to identify more of the information borne by a logic program than models do.

Contributions and structure of the paper Main contributions of the approach are as follows:

- Our semantics is able to distinguish conflicts not identifiable on the level of conflicts between rules.
- The well known problems with tautological and cyclic (and more generally, irrelevant) updates are removed in our semantics.

The language of generalized extended programs (GELP) is used in this paper. A symmetric treatment of assumptions is enabled thanks to the decision to use GELP: if A is an atom it can be assumed both A and $\neg A$. Let $\mathcal{B}$ be a belief/assumption operator. We can denote assumptions (of literals) by $\mathcal{B}A$ (in terms of default negation as $not\ \neg A$), or $\mathcal{B}\neg A$ ($not\ A$). Moreover, this language is suitable for a comparison of various approaches to logic program updates, see (Leite 2003; Homola 2004).

A recap of basic notions connected to semantics based on rejection of rules is presented and some problems inherent in that approach are illustrated by examples.

Next, Kripke structures associated with a program are presented. Then updates on Kripke structures are motivated and defined. Finally, properties of the updates on Kripke structures are described and our approach is evaluated.

Generalized extended logic programs

Let a set of propositional symbols (atoms) $\mathcal{A}$ be given. An *objective literal* is an atom (a positive literal) or an atom preceded by the explicit negation $\neg A$ (a negative literal). The set of all objective literals is denoted by Obj. A *subjective* literal is an objective literal L preceded by the default negation: $not\ L$. The set $\{not\ L : L \in Obj\}$ will be denoted by $\mathcal{D}$ (defaults). The set of *literals* is defined by $Lit = Obj \cup \mathcal{D}$.

A convention as follows is used: if literal L is of the form $\neg A$ ($not\ L'$), where $A \in \mathcal{A}$ ($L' \in Obj$) then $\neg L = A$ ($not\ L = L'$). For each literal L, the pairs L and $\neg L$ (L and $not\ L$) are called *conflicting literals.* A set of literals is called *consistent*, if it does not contain a pair of conflicting literals.

A *rule* is a formula r of the form $L \leftarrow L_1, \ldots, L_k$, where $k \geq 0$, and L, L_i are literals (for each i). We will denote L also by $head(r)$ and the set of literals $\{L_1, \ldots, L_k\}$ by $body(r)$. $body(r)$ can be split into two parts, $body^+(r) \subseteq Obj$ and $body^-(r) \subseteq \mathcal{D}$, where $body(r) = body^+(r) \cup body^-(r)$. The set of all rules forms the language $\mathcal{L}$. A subset of $\mathcal{L}$ is called a *generalized extended logic program* (program hereafter).

A *partial interpretation* of the language $\mathcal{L}$ is a consistent set of literals. The set of all partial interpretations of the language $\mathcal{L}$ is denoted by $Int_{\mathcal{L}}$.

A *total interpretation* is a partial interpretation I such that for each $L \in Obj$ holds $L \in I$ or $not\ L \in I$. An important special case is as follows: both $not\ A \in I$ and $not\ \neg A \in I$, where $A \in \mathcal{A}$ and I is a total interpretation.

The satisfaction relation and the notion of model are defined as usual. Notice that (propositional generalized extended logic) programs may be treated as Horn theories: each literal of the form $\neg A$, where $A \in \mathcal{A}$, or of the form $not\ L$, where $L \in Obj$, may be considered as a new propositional symbol. The least model of a Horn theory H is denoted by $least(H)$.

The definition of answer sets follows the idea of (Alferes et al. 2000). It respects the original definitions of (Gelfond and Lifschitz 1988; Gelfond and Lifschitz 1990), see Theorem 3. Answer sets do not contain subjective literals usually. However, in this paper answer sets contain also subjective literals. Default negations in heads of rules make such representation more appropriate.

Definition 1 (Answer set) Let P be a program and S be a total interpretation. Let $S^- = S \cap \mathcal{D}$.

S is an *answer set* of P iff $S = least(P \cup S^-)$. □

This, rather non-standard definition of answer sets, corresponds in a clear sense to the classical definition:

Definition 2 Let P be a program, $r \in P$ be a rule and S be a total interpretation, r^+ be the rule of the form $head(r) \leftarrow body^+(r)$.

$$\begin{aligned} supptd(P, S) &= \{L \in S : (\exists r \in P)\ S \models body(r), \\ &\quad L = head(r)\} \\ GL(P, S) &= \{r^+ : r \in P, body^-(r) \subseteq S^-\}. \square \end{aligned}$$

Theorem 3 *Let P be a program, S an interpretation, $S \setminus supptd(P, S) \subseteq \mathcal{D}$.*

Then $least(GL(P, S)) = supptd(P, S)$ iff S is an answer set of P. □

Causal rejection principle

In order to do the paper self-contained we recap the basic notions connected to the causal rejection principle and to semantics based on rejection of rules (the last term is borrowed from (Homola 2004)). All concepts are defined for the case of multidimensional dynamic logic programs.

Definition 4 ((Leite, Alferes and Pereira 2001)) A *multidimensional dynamic logic program* (also *multiprogram* hereafter) is a pair $(\mathcal{P}, G)$, where $G = (V, E)$ is an acyclic digraph, $|V| \geq 2$, and $\mathcal{P} = \{P_i : i \in V\}$ is a set of (generalized extended logic) programs.

We denote by $i \prec j$ that there is a path from i to j and $i \preceq j$ means that $i \prec j$ or $i = j$. If $i \prec j$, we say that P_j is *more preferred* than P_i. □

The class of multiprograms is denoted hereafter by $MDyLoP$.

Definition 5 A *semantics* of multiprograms is a function $SEM : MDyLoP \longrightarrow 2^{Int_{\mathcal{L}}}$.

Let $\Pi \in \mathit{MDyLoP}$. It is said that *SEM* is *based on rejection of rules* iff every $I \in \mathit{SEM}(\Pi)$ satisfies a condition of the form

$$I = \mathit{least}((\tau(\Pi) \setminus \tau(\mathit{RejR}(\Pi, I))) \cup \mathit{Assumpt}(\Pi, I)),$$

where $\tau(\Pi)$ is the multiset of all rules from all programs from Π, $\tau(\mathit{RejR}(\Pi, I))$ is a multiset of all rejected rules from Π w.r.t. I, $\mathit{Assumpt}(\Pi, I)$ is the set of all assumptions accepted by the semantics (with respect to the program Π and interpretation I) and the difference of multisets $\tau(S_1) \setminus \tau(S_2)$ contains no occurrence of a member of $\tau(S_2)$. □

There are some strategies how to implement $\mathit{RejR}(\Pi, I))$ and $\mathit{Assumpt}(\Pi, I)$. We present the strategy of (Leite, Alferes and Pereira 2001), see the definitions of sets $\mathit{rejected}(\Pi, I)$ and $\mathit{defaults}(\Pi, I)$:

Definition 6 It is said that two rules, r and r', are *conflicting*, if their heads are conflicting literals (notation: $r \bowtie r'$).

$$\begin{aligned} \mathit{rejected}(\Pi, I) &= \tau(\{r \in P_i : (\exists r' \in P_j) \\ &\quad i \prec j \wedge r \bowtie r' \wedge M \models \mathit{body}(r')\}) \\ \mathit{defaults}(\Pi, I) &= \{\mathit{not}\ L : (\neg\exists r \in \Pi)\ L \in \mathit{Obj}\ \wedge \\ &\quad \mathit{head}(r) = L \wedge I \models \mathit{body}(r)\}. \square \end{aligned}$$

The semantics of multiprograms based on rejection of rules implemented using $\mathit{rejected}(\Pi, I)$ and $\mathit{defaults}(\Pi, I)$ is dubbed here *dynamic answer set semantics.*

Remark 7 Another strategy of accepting assumptions is that of answer set semantics: if I is an interpretation then assumptions of I are in the set $\mathcal{D} \cap I$. A semantics based on this strategy is called *justified update.*

A rule $r \in P_i$ is rejected by a rule $r' \in P_j$ only if r' itself is not rejected according to the strategy of rule rejection proposed by (Eiter et al. 2002). The semantics which respect this strategy are called *backward* dynamic answer set semantics and *backward* justified update in (Šefránek 2003) and also in (Homola 2004).

The semantics based on the refined extension principle (REP) (Alferes et al. 2004) modifies the Definition 6 of $\mathit{rejected}(\Pi, I)$ as follows: instead of $i \prec j$ it is required $i \preceq j$. It is possible to distinguish refined dynamic answer sets semantics and refined backward dynamic answer set semantics (Homola 2004).

For a comparison of various semantics see (Eiter et al. 2002; Leite 2003; Homola 2004).

Example 8 (Cyclic updates)

$$P = \{a \leftarrow \qquad U = \{a \leftarrow b$$
$$\mathit{not}\ a \leftarrow\} \qquad b \leftarrow a\}$$

Let multiprogram Π consists of two programs P, U. Then $\mathit{rejected}(\Pi, \{a, b\}) = \{\mathit{not}\ a \leftarrow\}$, $\mathit{defaults}(\Pi, \{a, b\}) = \emptyset$ and $\{a, b\}$ is the dynamic answer set of Π. It has to be mentioned that the problem illustrated by this example is removed by the semantics based on REP (Alferes et al. 2004). □

However, the problem of tautological and cyclic updates is not solved by REP for (general) multiprograms.

Example 9 ((Šiška 2004)) Let a multiprogram Π is given by a set of programs $\{P_i, P_j, P_k, P_l\}$, and by partial ordering $P_i \prec P_j \prec P_k$ and $P_l \prec P_k$, i.e. P_k is the most dominant module and there are some incomparable modules.

Let $P_k = \{b \leftarrow\}$, $P_l = \{\mathit{not}\ a \leftarrow\}$, $P_j = \{\mathit{not}\ a \leftarrow \mathit{not}\ a\}$, $P_i = \{a \leftarrow\}$.

Rule $a \leftarrow$ is rejected with respect to interpretation $S = \{b, \mathit{not}\ a\}$ because of the rule $\mathit{not}\ a \leftarrow \mathit{not}\ a$. S is the dynamic answer set. Unfortunately, it is not known how to extend the semantics based on REP to the case of general multiprograms and it seems that it is impossible (Šiška 2004). □

There are some other examples of problematic behaviour of semantics based on rejection of rules.

Example 10 ((Eiter et al. 2002)) Let Π consists of P, U, where $P \prec U$.

$$P = \{a \leftarrow b \qquad U = \{\mathit{not}\ b \leftarrow \mathit{not}\ a\}$$
$$b \leftarrow\}$$

If $S = \{\mathit{not}\ b, \mathit{not}\ a\}$ then $\mathit{rejected}(\Pi, S) = \{b \leftarrow\}$, $\mathit{defaults}(\Pi, S) = \{\mathit{not}\ a\}$ and S is a dynamic answer set of $P \cup U$.

If we apply (intuitively) the principle of inertia, we have no reason to reject the fact from P. Notice that rejected rules and defaults are based on a choice of an interpretation. It seems that some problems with semantics based on rejection of rules are caused by that (too free) choice. Anyway, a step from intuitions and examples to a more fundamental theory (theories) is needed. For an attempt to give such a theory see (Šefránek 2004). □

Notice that the proposal based on the REP (Alferes et al. 2004) does not solve the problem of irrelevant updates of the type presented in Example 10. The proposal is focused only on the inconsistent programs.

Later, in Example 21, we present a conflict between programs that is even not identifiable on the level of conflicts between rules.

In order to summarize this Section: We have seen that a semantics based on rejection of rules leads to some problems. We propose to turn to a more semantic approach. Our semantics is focused not only on the conflicts between rules but also on conflicts between assumptions. Hence, semantics should be sensitive both to rejection of rules and to rejection of assumptions. We believe that it is necessary to record dependencies between interpretations (an between literals) in the semantics in order to be able understand the topic of rejections. A sort of Kripkean semantics (aiming at a representation of dependencies between interpretations) is presented in this paper.

In general, rejections play a fundamental role in human (nonmonotonic) reasoning. Therefore, understanding rejections can play an important role in nonmonotonic reasoning theory.

Kripke structure associated with a program

In this Section a semantic treatment of dependencies between sets of literals is presented. The dependencies are encoded in (rather nonstandard) Kripke structures. First an example.

Example 11 Let P be

$$\begin{aligned} \{a &\leftarrow \textit{not } b \\ \textit{not } \neg b &\leftarrow \neg a\}. \end{aligned}$$

Dependencies between interpretations of P can be recorded in a graph. Nodes of the graph are interpretations. Some examples of edges: If we believe in $\{\textit{not } b\}$ and in P then we have to accept also $\{\textit{not } b, a\}$. Hence, the edge $(\{\textit{not } b\}, \{\textit{not } b, a\})$ should be included in the graph.

Observe that it is possible accept also $\{\textit{not } b, a, \textit{not } \neg a\}$. If we accept a then we should accept also $\textit{not } \neg a$ in order to be consistent.

On the other hand, $\{\textit{not } \neg b, \neg a\}$ is accessible from $\{\neg a\}$, but it is not reasonable to make also $\{\textit{not } \neg b, b, \neg a\}$ accessible from $\{\neg a\}$ – b is not forced by $\textit{not } \neg b$.

Finally, consider interpretation $\{\textit{not } b, \neg a\}$. Inconsistent set of literals $\{a, \textit{not } \neg a, \textit{not } b, \neg a\}$ should be accessible from it.

Each edge is generated - in a sense - by a rule. It is useful to label the edge by the head of the rule. Labels are important for the update operation. □

We will call the graphs illustrated by Example 11 Kripke structures. A more detailed motivation is postponed after the definition 12. Here only the basic intuition. If our knowledge of the world is represented by an interpretation w then (the meaning of) a program P may be viewed as a set of transitions to other worlds, dependent, in a sense, on w. If $\textit{body}(r)$ is satisfied in w for some $r \in P$ then the world $w \cup \{\textit{head}(r)\}$ is dependent on w. Example 11 motivates why and when it is reasonable to consider $w \cup \{\textit{head}(r), \textit{not } \neg\textit{head}(r)\}$ to be dependent on w. The relation of dependency should be transitive, asymmetric and irreflexive.

Definition 12 (Kripke structure) Let P be a program. A *Kripke structure* $\mathcal{K}^P$ *associated* with P is a triple $(W, \rho, \textit{label})$, where:

- $W = \textit{Int}_{\mathcal{L}} \cup \{w_\perp\}$, W is called the set of possible worlds, $w_\perp$ is the representative of the set of all inconsistent sets of literals,
- ρ is a binary relation on $W \times W$, it is called the accessibility relation and it contains the set of all pairs (w, w') such that $w \neq w'$, $\exists r \in P\ (w \models \textit{body}(r))$ and w' satisfies exactly one of the conditions:
  1. $w' = w \cup \{\textit{head}(r), \textit{not } \neg\textit{head}(r)\}$ iff $\textit{head}(r) \in \textit{Obj}$,
  2. $w' = w \cup \{\textit{head}(r)\}$ iff $\textit{head}(r) \in \mathcal{D}$,
  3. $w' = w_\perp$ iff $\textit{not } \textit{head}(r) \in w \vee \neg\textit{head}(r) \in w$,
- $\textit{label} \subseteq \rho \times \textit{Lit}$ is a relation such that $((w, w'), \textit{head}(r)) \in \textit{label}$, whenever $(w, w') \in \rho$ is generated by r; an edge $(w, w') \in \rho$ is *generated* by r if $w \models \textit{body}(r)$ and
  – $\textit{head}(r) \in w' \setminus w$,
  – or $w' = w_\perp$ and either $\textit{not } \textit{head}(r) \in w$ or $\neg\textit{head}(r) \in w$. □

Example 13 Let w be $\{a, b\}$ and P contains rules $\neg a \leftarrow b$ and $\neg b \leftarrow a$. Then $((w, w_\perp), \neg a) \in \textit{label}$, $((w, w_\perp), \neg b) \in \textit{label}$. □

$\mathcal{K}^P$ may be viewed as a derivation graph. But we emphasize that it is a semantic structure. First, $\mathcal{K}^P$ records many irrelevant "derivations". Assume that a program P contains a rule $b \leftarrow a$, and a is not derivable from P. Consider the world $w = \{a, c\}$. The meaning of P allows a transition from w to $w_1 = \{a, c, b, \textit{not } \neg b\}$. More importantly, the transition (the accessibility relation) carries (encodes) a semantic information about dependencies between possible worlds. A semantic concept *true because of* (introduced in (Šefránek 2004)) may be based on edges (in Kripke structures). The deep Kripke's insight into the dependencies between knowledge states provides a very flexible, multiform and useful tool for doing semantics.

Definition 14 Let $\mathcal{K}^P = (W, \rho, \textit{label})$ be a Kripke structure associated with a program P. If $e = (u, v) \in \rho$, it is said that e is a *ρ-edge* and u (v) is called the *source* (the *target*) of e.

Let $w_0 \in W$. It is said that $\langle w_0 \rangle$ is a *trivial* sequence.

A *ρ-path* is a trivial sequence or a sequence σ of ρ-edges $(w_0, w_1), (w_1, w_2), \ldots, (w_{n-1}, w_n)$, where $n \geq 1$. The path is denoted also by a shorthand of the form $\langle w_0, w_1, \ldots, w_n \rangle$.

Let $\sigma = \langle w_0, w_1, \ldots, w_n \rangle$, where $n \geq 0$ be a ρ-path. It is said that σ is *rooted* in w_0 (also w_0-rooted) and w_0 is the root of σ. If there is no ρ-edge (w_n, w) in $\mathcal{K}^P$, we say that σ is *terminated* in w_n, and w_n is called the *terminal* of σ. □

Dependencies between interpretations (and literals) are encoded by paths.

We are now ready to state (in terms of nodes and paths in $\mathcal{K}^P$) conditions of being an answer set of a program P.

Definition 15 (Good worlds) Let σ be a ρ-path $\langle w_0, \ldots, w_n \rangle$, $n \geq 0$. We say that σ is *correctly rooted*, if $w_0 \subseteq \mathcal{D}$.

A correctly rooted ρ-path σ terminated in a total interpretation w is called a *distinguished* path and w is called a *good world*. □

Remark 16 It is possible to change the definition of accessibility relation. Instead of

$$w' = w \cup \{head(r), not\ \neg head(r)\}$$

can be used $w' = w \cup \{head(r)\}$.

Let r, with $head(r) \in Obj$, generates an edge on a distinguished path from w_0 to w_n according to Definition 12. Obviously, there is a distinguished path from $w_0 \cup \{not\ \neg head(r)\}$ to w_n according to the modified definition.

However, we decided to use heuristically more useful definition. □

Theorem 17 *Let P be a program. Then w_n is a good world in $\mathcal{K}^P$ iff it is an answer set of P.* □

Remark 18 If $\mathcal{D}$ is a terminal in $\mathcal{K}^P$, it is the answer set of P. Even the trivial sequence $\langle \mathcal{D} \rangle$ is correctly rooted and terminated in $\mathcal{D}$. □

Updates on Kripke structures

We now define the elementary case of an update operation on two Kripke structures.

Suppose in this Section two programs, P and U, and the Kripke structures, $\mathcal{K}^P = (W, \rho^P, label^P)$ and $\mathcal{K}^U = (W, \rho^U, label^U)$, associated with P and U, respectively. U (the updating program) is more preferred than P (the original program). The multiprogram defined by $P \prec U$ is denoted here by Π.

We intend to define an operation $\oplus$ on Kripke structures. The resulting Kripke structure $\mathcal{K}^U \oplus \mathcal{K}^P = \mathcal{K}^{U \oplus P}$ should be based on $\mathcal{K}^U$ while a reasonable part of $\mathcal{K}^P$ is preserved. The set of nodes W remains unchanged, but some ρ^P-edges should be rejected.

Example 19 Let P be $\{a \leftarrow b\}$ and U be $\{\neg a \leftarrow b\}$. Then $\mathcal{K}^P$ contains an edge $e_1 = (\{b\}, \{b, a, not\ \neg a\})$ and $(e_1, a) \in label^P$. Similarly, $\mathcal{K}^U$ contains an edge $e_2 = (\{b\}, \{b, \neg a, not\ a\})$ and $(e_2, \neg a) \in label^U$.

e_2 attacks e_1, in a sense. □

Definition 20 (Attacked edges) Let $e_1 = (w, w_1) \in \rho^P$, $e_2 = (w, w_2) \in \rho^U$.

We say that e_1 is *attacked* by e_2 iff $((w, w_1), L_1) \in label^P$, $((w, w_2), L_2) \in label^U$ and L_1, L_2 are conflicting literals. □

Attacked edges correspond to conflicts between rules, see Proposition 41.

However, there are more complicated conflicts observable on pairs of Kripke structures. A series of examples below motivates the need for a finer analysis.

Example 21

$$P = \{a \leftarrow c\} \qquad U = \{b \leftarrow not\ a,\ c \leftarrow b\}$$

There is no conflict between rules of both programs. No rule can be rejected and the meaning of $P \cup U$ cannot be updated according to the causal rejection principle.

However, there is a sort of conflict between the programs: The literal a in the less preferred program P depends on the literal c. On the other hand, the literal c depends on the default assumption *not a* in the more preferred program U. This dependency of a on *not a* can be resolved by rejecting the less preferred dependency.

The intuition described above can be expressed (formalized) in terms of our Kripkean semantics. There is a ρ^U-path from $\{not\ a\}$ to $w = \{not\ a, b, not\ \neg b, c, not\ \neg c\}$ and there is a ρ^P-edge $e = (w, w_\bot)$ labeled by a. We propose to reject e from $\mathcal{K}^{U \oplus P}$. □

However, a carefree rejection of a ρ^P-edge $(w, w_\bot)$ because of a ρ^U-path $\langle w_0, \dots, w \rangle$ could cause some complications:

Example 22 Consider Example 10.

$$P = \{a \leftarrow b,\ b \leftarrow\} \qquad U = \{not\ b \leftarrow not\ a\}$$

$e_1 = (\{not\ a\}, \{not\ a, not\ b\}) \in \rho^U$ and $e_2 = (\{not\ a, not\ b\}, w_\bot) \in \rho^P$, $(e_2, a) \in label^P$.

Observe that there is a conflict between the nonmonotonic assumption *not a* in the source of the ρ^U-edge and the literal a dependent on the empty interpretation in P.

We prefer facts over assumptions. In other words: an interesting information can be expressed by the rule of U; however the information cannot be used, if our knowledge is represented by $P \cup U$.

We propose to insert the edge e_2 into $\mathcal{K}^{U \oplus P}$. It can be said that overriding of e_2 by e_1 is blocked by the path $\sigma = \langle \emptyset, \{b, not\ \neg b\}, \{b, not\ \neg b, a, not\ \neg a\} \rangle$.

The path σ falsifies e_1 in the sense as follows:

- it is rooted in $\emptyset$,
- no edge of σ is attacked by a ρ^U-edge,
- the terminal of σ contains a conflicting literal w.r.t. a literal from the source of e_1,
- finally, there is no ρ^U-edge $(\emptyset, \{not\ a\})$. □

Example 23 Let P be as in Example 10. U be $not\ b \leftarrow not\ c$. Similarly, we do not propose to reject the ρ^P-edge $(\{not\ b, not\ c\}, w_\bot)$. The target of the ρ^U-edge $(\{not\ c\}, \{not\ b, not\ c\})$ is incompatible with the world $\{b, not\ \neg b\}$, which is supported in $\mathcal{K}^P$ by the path $\sigma = \langle \emptyset, \{b, not\ \neg b\} \rangle$ rooted in the empty interpretation. While the dependency of *not b* on *not c* may be of interest, the information contained in $P \cup U$ does not allow to use this information. It can be said again that overriding is blocked. □

Some trivial paths are important from the updates point of view.

Example 24 Let P be $\{a \leftarrow not\ b\}$ and U be $\{b \leftarrow a\}$.

The edge $(\{not\ b, not\ a\}, w_\bot) \in \rho^P$ could be considered as overridden by the trivial ρ^U-path $\langle\{not\ b, not\ a\}\rangle$. In other, more intuitive, terms: the stable model $\{not\ b, not\ a\}$ of the more preferred program U should override an (unsupported) dependency of a on $not\ b$ in P.

Observe that b depends on $not\ b$ in $P \cup U$. This dependency should be overridden. □

The following definitions resulted from our intuitive analysis.

Definition 25 (Supported sets and literals) Let P be a program. A consistent *set* of literals w is *supported* in $\mathcal{K}^P$ iff there is a ρ^P-path σ from $\emptyset$ to w, $w \neq w_\bot$. σ is called a path *supporting* w.

A *literal* L is *supported* in $\mathcal{K}^P$ iff there is a set of literals w supported in $\mathcal{K}^P$ and $L \in w$. A path supporting w is called also the path supporting L. □

Definition 26 Let L_1, L_2 be conflicting literals.

It is said that L_1 is *falsified* by L_2 iff L_2 is supported and L_1 is not supported.

It is said that L_1 is *carefully falsified* by L_2 iff

- L_2 is supported in $\mathcal{K}^P$,
- there is no edge attacked by a ρ^U-edge in a ρ^P-path σ supporting L_2,
- and L_1 is not supported in $\mathcal{K}^U$. □

Definition 27 (Blocking) Let L_1, L_2 be conflicting literals, $L_1 \in w_1$. Let $\mathcal{D}$ be a terminal in $\mathcal{K}^U$.

A ρ^U-edge (w_0, w_1) is *blocked* iff L_1 is carefully falsified by L_2. A trivial path $\langle\mathcal{D}\rangle$ is *blocked* iff $L_1 \in \mathcal{D}$ and L_1 is carefully falsified by L_2. □

Remark 28 Notice that Definition 27 is sufficiently general: if a literal L_1 from w_0 is conflicting with a literal L_2 supported in $\mathcal{K}_P$ (as in Example 10) then also $L_1 \in w_1$. □

Definition 29 (Overriding) A ρ^U-edge $e = (w_0, w_1)$ *overrides* a ρ^P-edge $(w_1, w_\bot)$ if e is not blocked.

Let $\mathcal{D}$ be a terminal in $\mathcal{K}^U$. The trivial path $\langle\mathcal{D}\rangle$ overrides a ρ^P-edge $(\mathcal{D}, w_\bot)$ if it is not blocked. □

Definition 30 (Cancelled labels) Assume that $e_1 = (w_0, w_1) \in \rho^U$ overrides $e_2 = (w_1, w_\bot) \in \rho^P$. Let L_1, L_2 be conflicting literals, $(e_1, L_1) \in label^U$, $(e_2, L_2) \in label^P$. It is said that (e_2, L_2) is *canceled* by (e_1, L_1). The set of all canceled labels is denoted by *Can*.

Definition 31 (Rejected edges) Consider $\mathcal{K}^P = (W, \rho^P, label^P)$ and $\mathcal{K}^U = (W, \rho^U, label^U)$. We say that $e \in \rho^P$ is *rejected*, if $e \notin \rho^U$ and

- e is attacked by some $e' \in \rho^U$,
- or e is of the form $(w, w_\bot)$, it is overridden by a ρ^U-edge $e' = (w_0, w)$ and each $(e, L) \in label^P$ is canceled
- or e is of the form $(\mathcal{D}, w_\bot)$ and it is overridden by $\mathcal{D}$.

The set of all rejected edges from ρ^P is denoted by $Rejected_{\rho^U}(\rho^P)$. □

Definition 32 (Update on Kripke structures)

$$\begin{aligned} \mathcal{K}^U \oplus \mathcal{K}^P &= \mathcal{K}^{U\oplus P} = (W, \rho^{U\oplus P}, label^{U\oplus P}) \\ \rho^{U\oplus P} &= \rho^U \cup (\rho^P \setminus (Rejected_{\rho^U}(\rho^P)) \end{aligned}$$

and $label^{U\oplus P}$ is $label^U \cup (label^P \setminus Canc)$ restricted to $\rho^{U\oplus P}$. □

General case

Consider the set of all Kripke structures $\mathcal{K} = \{\mathcal{K}^i : i \in V\}$ associated to all programs from a multiprogram $(\mathcal{P}, G)$, $G = (V, E)$. We will use a notation as follows: $\mathcal{K}^i = (W, \rho^i, label^i)$. The preference relation on programs is extended also to Kripke structures, i.e. if $i \prec j$ then $\mathcal{K}^j$ is more preferred than $\mathcal{K}^i$.

Our goal is to define an updated Kripke structure $\oplus\mathcal{K}$.(We could, alternatively, define an updated Kripke structure with respect to a state $s \in V$. However, the essence of the definition remains unchanged.) The main idea of this update is the same as for the case of two Kripke structures. It is needed to handle only some technical details.

Let $i_1 \prec i_2 \prec i_3$, $e_1 \in \rho^{i_1}, e_2 \in \rho^{i_2}, e_1 \in \rho^{i_3}$. Suppose that e_2 attacks e_1 according to Definition 20. Of course, also e_1 attacks e_2 according to the same Definition. Therefore, we have to identify maximally preferred Kripke structures in which occurs an edge, in order to define attacked (and overridden) edges for the multidimensional case. Notice that there can be more maximal $i \in V$ such that $e_1 \in \rho^i$. Similarly for e_2.

Definition 33 (Attacked edges) Let L_1, L_2 be conflicting literals. Consider two edges e_1, e_2. It is said that e_1 is *attacked* by e_2 iff for each j such that $e_1 \in \rho^j$ and $(e_1, L_1) \in label^j$ there is k such that $j \prec k$, $e_2 \in \rho^k$ and $(e_2, L_2) \in label^k$.

The set of all attacked edges in $\bigcup_{i\in V} \rho^i$ is denoted by *Atck*. □

We have to be focused on paths built by edges from different Kripke structures in order to define blocking and overriding in an appropriate way. Therefore, we will speak about Δ-paths, where Δ is a set of vertices from V. A ρ^Δ-path is a sequence $\sigma = \langle x_0, \ldots, x_n\rangle$ such that for each $e = (x_i, x_{i+1})$ there is $k \in \Delta$ such that $e \in \rho^k$.

We now extend the definition of supported sets and literals to the case of ρ^Δ-paths.

Definition 34 (Supported sets and literals) Let $\Delta \subseteq V$. Let $\mathcal{K}^\Delta = \{\mathcal{K}^j : j \in \Delta\}$.

A consistent *set* of literals w is *supported* in $\mathcal{K}^\Delta$ iff there is a ρ^Δ-path σ from $\emptyset$ to w, $w \neq w_\perp$. σ is called a path *supporting* w.

A *literal* L is *supported* in $\mathcal{K}^\Delta$ iff there is a set of literals w supported in $\mathcal{K}^\Delta$ and $L \in w$. A path supporting w is called also the path supporting L. □

Observe that $\mathcal{K}^\Delta$ is a *set* of Kripke structures. We don't need to modify the definition of falsification.

A comment concerning a modified definition of *careful* falsification: A literal supported by a path from a less preferred Kripke structure can falsify an unsupported assumption from a more preferred Kripke structure. This idea is generalized to sets of Kripke structures:

Definition 35 Let $\Delta \subseteq V$, $\Gamma \subseteq V$, $(\forall i \in \Delta)$ $(\forall j \in \Gamma)$ $i \prec j$. It is said that L_1 is *carefully falsified* by L_2 iff

- L_2 is supported in $\mathcal{K}^\Delta$,
- there is a ρ^Δ-path σ supporting L_2 with no edge attacked by a ρ^Γ-edge,
- and L_1 is not supported in $\mathcal{K}^\Gamma$. □

Definition 36 (Blocking) Let L_1, L_2 be conflicting literals, $L_1 \in w_1$. Let $\mathcal{D}$ be a terminal in $\mathcal{K}^{P_j}, j \in V$.

An edge $(w_0, w_1) \in \rho^j \setminus Atck$ is *blocked* iff L_1 is carefully falsified by L_2. A trivial path $\langle \mathcal{D} \rangle$ is *blocked* iff $L_1 \in \mathcal{D}$ and L_1 is carefully falsified by L_2. □

Definition 37 (Overriding) Let $i \prec j$. A ρ^j-edge $e = (w_0, w_1)$ *overrides* a ρ^i-edge $(w_1, w_\perp)$ if e is not blocked.

Let $\mathcal{D}$ be a terminal in $\mathcal{K}^j$. If the trivial path $\langle \mathcal{D} \rangle$ is not blocked then it overrides the ρ^i-edge $(\mathcal{D}, w_\perp)$. □

Definition 38 (Cancelled labels) Assume that $e_1 = (w_0, w_1) \in \rho^j$ overrides $e_2 = (w_1, w_\perp) \in \rho^i$. Let L_1, L_2 be conflicting literals, $(e_1, L_1) \in label^j$, $(e_2, L_2) \in label^i$. It is said that (e_2, L_2) is *canceled* by (e_1, L_1). The set of all canceled labels is denoted by *Can*.

Definition 39 (Rejected edges) Consider $\mathcal{K}^i = (W, \rho^i, label^i)$ and $\mathcal{K}^j = (W, \rho^j, label^j)$. We say that $e \in \rho^i$ is *rejected*, if $e \notin \rho^\Gamma$, where $\Gamma) = \{k : i \prec k\}$ and

- e is attacked by an edge $e' \in \rho^\Gamma$,
- or e is of the form $(w, w_\perp)$, it is overridden by a ρ^Γ-edge $e' = (w_0, w)$ and each $(e, L) \in label^i$ is canceled,
- or e is of the form $(\mathcal{D}, w_\perp)$, it is overridden by the trivial path $\langle \mathcal{D} \rangle$ and $\langle \mathcal{D} \rangle$ is trivial in each $\mathcal{K}^k \in \mathcal{K}^\Gamma$.

We denote the set of rejected edges by *Rej*. □

Definition 40 $\oplus\mathcal{K}$ is called *updated* Kripke structure, if

$$\begin{aligned} \rho^\oplus &= (\bigcup_{i \in v} \rho_i) \setminus Rej \\ \oplus\mathcal{K} &= (W, \rho^\oplus, label^\oplus), \end{aligned}$$

where $label^\oplus$ is $(\bigcup_{i \in V} label^i) \setminus Can$ restricted to $\rho^\oplus$. □

Let $\oplus\mathcal{K}$ be an updated Kripke structure. Distinguished paths and good worlds in $\oplus\mathcal{K}$ are defined according to Definition 15 and they play an important role: they represent the meaning of the updated Kripke structure.

Notice that it is possible to construct a program P such that the set of all answer sets of P coincide with the set of all good worlds in $\oplus\mathcal{K}$: Let w be a good world in $\oplus\mathcal{K}$. Consider a distinguished path $\sigma = \langle w_0, w_1, \ldots, w_n \rangle$, where $w_n = w$. Let $((w_i, w_{i+1}), L) \in label^\oplus$. We add a rule $L \leftarrow w_i$ to P for each edge on one distinguished path for each good world.

Properties

This Section contains a brief summary of the results. First, the relation of rules rejection to edges rejection is discussed. Second, it is shown that the critical cases for updates based on causal rejection principle are solved by updates on Kripke structures. Finally, an evaluation of the updates on Kripke structures from the dependency theory of (Šefránek 2004) point of view is sketched.

The following proposition shows that the causal rule rejection principle is satisfied in updated Kripke structures in the following sense: if a rule $r \in P$ generates an attacked edge then the rule is rejected.

Proposition 41 *Let $e_1 = (w_0, w_1) \in Atck$, let $i \in V$ be maximal vertex such that $(e_1, L_1) \in label^i$.*

Then there is a rule r_1 of the form $L_1 \leftarrow Body$ in P_i such that $r_1 \in rejected(\Pi, w_0)$. □

Proof: There is $e_2 = (w_0, w_2) \in \rho^j$, $i \prec j$, $(e_2, L_2) \in label^j$, and L_1, L_2 are conflicting literals. Hence, $(\exists r_2 \in P_j)$ $w_0 \models body(r_2), head(r_2) = L_2$.

Remark 42 The converse of the Proposition 41 does not hold. If a rule is rejected, some edges generated by this rule may be not attacked. Let Π consists of P and U. P be $\{a \leftarrow\}$, U be $\{not\ a \leftarrow b\}$ and w be $\{a, b\}$. Then $(\emptyset, \{a, not\ \neg a\}) \in \rho^P$ is attacked by no edge in ρ^U, but the rule $a \leftarrow$ belongs to $rejected(\Pi, w)$ (Mariničová 2001).

It is a further argument in favour of the claim that rejection defined within the frame of Kripkean semantics is more sensitive (able to make a more fine distinguishing) than the causal rejection principle.

Consider edges of the form $(w, w_\perp) \in \rho^P$. There is no general relation between rejecting such edges and rejecting rules. The Example 21 shows a rejected edge to inconsistency, but the rule generated that edge is not rejected.

On the other hand, it is possible that an edge to inconsistency, generated by a rejected rule, is not rejected (because of blocking):

$$P = \{not\ c \leftarrow \qquad\qquad U = \{c \leftarrow b\}$$
$$not\ b \leftarrow not\ c\}$$

The rule *not* $c \leftarrow$ is rejected in dynamic logic programming paradigm w.r.t. each interpretation w such that $b \in w$, but the edge $(\{b, c\}, w_\perp) \in \rho^P$ is not rejected – the reason is that $(\{b\}, \{b, c\})$ is blocked. □

We now present a series of examples in order to illustrate that updates on Kripke structures are immune to some well known troubles connected to the approaches based on the causal rejection principle.

Example 43 ((Leite 2003)) Let P be $\{a \leftarrow\}$ and U be $\{not\ a \leftarrow not\ a\}$. Tautologies do not generate edges. Therefore the only good world in $\mathcal{K}^{U \oplus P}$ is $\{a\}$. □

Observation 44 *Tautologies do not exert influence on updates of Kripke structures.* □

Example 45 (Cyclic updates) Consider Example 8.

$$P = \{a \leftarrow \quad not\ a \leftarrow\} \qquad U = \{a \leftarrow b \quad b \leftarrow a\}$$

Edge $(\{a, b\}, w_\perp) \in \rho^P$ should not be rejected. Edge $(\{a\}, \{a, b\}) \in \rho^U$ is blocked by the edge $(\emptyset, \{not\ a\}) \in \rho^P$. There is no good world in $\mathcal{K}^{U \oplus P}$. □

The following proposition claims that cycles in a more preferred program U do not allow rejection of supported literals from a less preferred program P. A similar, but rather complicated, proposition holds for the general case.

Proposition 46 *Let w be a set of mutually dependent literals in $\mathcal{K}^U$, i.e. for all $L_1, L_2 \in w$ there is a ρ^U-path from a possible world w_0 to w and $L_1 \in w_0$, $L_2 \notin w_0$. Let w be not supported in $\mathcal{K}^U$ and there is no edge $(w, w') \in \rho^U$.*

If $(w, w_\perp) \in \rho^P$, $((w, w_\perp), L) \in label^P$ and L is supported in $\mathcal{K}^P$ then $(w, w_\perp) \notin Rejected_{\rho^U}(\rho^P)$. □

It means that Kripkean semantics of updates does not allow to remove inconsistencies in less preferred programs on the basis of cycles in more preferred programs.

Tautological and cyclic updates do not represent a problem for Kripkean semantics even for multiprograms.

Example 47 Consider Example 9: a multiprogram $\{P_i, P_j, P_k, P_l\}$ is given, where $P_i \prec P_j \prec P_k$ and $P_l \prec P_k$.

$P_k = \{b \leftarrow\}$, $P_l = \{not\ a \leftarrow\}$, $P_j = \{not\ a \leftarrow not\ a\}$, $P_i = \{a \leftarrow\}$.

There is no good world in $\oplus\mathcal{K}$. □

Assumptions from more preferred programs do not override facts from the less preferred programs:

Example 48 Consider Example 10. We have observed a conflict between nonmonotonic assumption *not* a in the root of a ρ^U-path and the literal a dependent on the empty interpretation in P. Overriding of $e_2 = (\{not\ a, not\ b\}, w_\perp) \in \rho^P$ by $e_1 = (\{not\ a\}, \{not\ a, not\ b\}) \in \rho^U$ has been blocked by the ρ^P-path $\langle \emptyset, \{b, not\ \neg b\}, \{b, not\ \neg b, a, not\ \neg a\}\rangle$. □

The proposition 50 claims that Kripkean semantics is immune to irrelevant (in a sense) updates.

Definition 49 Let Π be a multiprogram, S be an interpretation. Let $RejR(\Pi, S)$ be a set of rules rejected according to a semantics based on rejection of rules. If $RejR(\Pi, S) \neq \emptyset$ and there is $L_1 \in S$ carefully falsified by some L_2 then $RejR(\Pi, S)$ is an irrelevant update. □

An irrelevant update is caused by a rather free choice of an interpretation in the frame of a semantics based on rejection of rules. In Kripkean semantics each world w containing a carefully falsified literal is the source of an edge $(w, w_\perp)$.

Proposition 50 *Let w be a good world in $\mathcal{K}^j$. Let $L_1 \in w$ be carefully falsified by L_2. Then w is not a good world in $\oplus\mathcal{K}$.*

Example 21 illustrates the following observation.

Observation 51 *Kripkean semantics of updates enables to recognize some conflicts between programs that are not identifiable on the level of conflicts between rules. In such cases some edges are rejected, but there is no reason to reject a rule.* □

Finally, we refer to (Šefránek 2004) where a dependency theory is proposed, *true because of* is defined and postulates for solving conflicts between dependencies are introduced. Our Kripkean semantics is an instance of the dependency theory.

Theorem 52 *The update operation on Kripke structures satisfies the postulates of (Šefránek 2004) for dependencies rejection.* □

Conclusions

Summary of the results.

A significantly modified Kripkean semantics of logic program updates is presented. The semantics enables to solve the problem of cyclic and irrelevant updates. It is able to identify the conflicts not distinguishable by the causal rejection principle. The basic feature of the semantics is an inclusion of dependencies between beliefs and belief sets into the semantics.

Discussion.

A dependency theory introduced in (Šefránek 2004) provides a more abstract background for evaluation of our semantics. Some postulates for rational rejections of dependencies are presented in the dependency theory.

We believe that a semantic investigation of rejections is of fundamental importance for understanding of (nonmonotonic) reasoning. The dependency theory represents an attempt to proceed from an analysis of problematic examples to some fundamental principles which enable to evaluate solutions of such examples.

On the other hand, different formalization can serve for different goals. Some differences between formalizations may be explained by a clash of intuitions, but various intuitions behind various formalizations may be acceptable. Anyway, an explicit formulation of principles and postulates should help to clear the field.

Kripkean semantics of logic program updates. The Kripkean semantics presented in this paper represents a continuation of our research. Nevertheless, it is fundamentally modified. The most important new features are based on the concept of blocking. That feature enables to recognize conflicts not identifiable on the level of conflicts between rules. A discussion of the semantics from the viewpoint of the troubles connected to the approaches based on the causal rejection principle is also a new one.

An attempt to give a Kripkean semantics for logic program updates is given in (Šefránek 2000). The semantics was defined only for two programs. More importantly, features of blocking has been not understood and no comparison to the approaches based on causal rejection of rules has been given. A modified and general semantics was proposed in (Mariničová 2001). The notion of attacked edges was introduced. However, the feature of blocking was not understood and conflicts between belief sets were not recognized. No attention has been devoted to the troubles connected to semantics based on rejection of rules.

Open problems, future plans.

A more detailed comparison of the presented approach to the approaches based on rejection of rules. An elaboration of the dependency theory and investigation of the relation of the dependency theory to strong (and uniform) equivalence of logic programs. An ongoing research is devoted to an implementation and to investigation of computational aspects of the approach (Galbavý, to appear). Our results in dependency theory (Šefránek 2004) indicate possibilities of updated Kripke structures pruning.

Acknowledgments: I would like to thank Martin Homola, Martin Baláž, Jozef Šiška and Tomáš Galbavý for inspiration and for the remarks to earlier versions of this paper. I am also thankful to anonymous referees for valuable comments. This work was partially supported by grant VEGA 1/0173/03.

References

Alferes, J.J., Pereira,L.M. *Update-programs can update programs.* LNAI 11126, Springer 1996

Alferes, J.J., Leite, J.A., Pereira, L.M., Przymusinska, H., Przymusinski, T.C. *Dynamic Updates of Non-Monotonic Knowledge Bases.* The Journal of Logic Programming 45 (1-3):43-70, September/October 2000

Alferes, J., Banti, F., Brogi, A., Leite, J. *Semantics for dynamic logic programming: a principled based approach.* In V. Lifschitz and I. Niemel, (eds.), Procs. of the 7th International Conference on Logic Programming and Nonmonotonic Reasoning, Springer-Verlag, LNAI, 2004.

Eiter, T., Fink, M., Sabbatini, G., Tompits, H. *On properties of update sequences based on causal rejection.* Theory and Practice of Logic Programming 2(6), 711-767, 2002

Galbavý, T., Master Thesis, to be finished in 2005.

Gelfond, M., Lifschitz, V. *The Stable Model Semantics for Logic Programming.* Proc. of 5th ICLP, MIT Press, 1988, 1070-1080

Gelfond, M., Lifschitz, V. *Logic programs with classical negation.* Proc. of 7th ICLP, MIT Press, 1990, 579-597

Homola, M. *On Relations of the Various Semantic Approaches in Multidimensional Dynamic Logic Programming.* Mater Thesis, Comenius University, Bratislava

Leite, J. *Evolving Knowledge Bases.* IOT Press, 2003.

Leite, J., Pereira, L. *Generalizing Updates: from models to programs.* In LNAI 1471, 1997

Leite, J., Pereira, L. *Iterated Logic Programs Updates.* In Proc. of JICSLP98

Leite, J.A., Alferes, J.J., Pereira, L.M. *Multidimensional dynamic knowledge representation.* In: Eiter, T., Faber, W., Truszczynski (Eds.): Procs. of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning, 2001, Springer, 365-378

Marek, W., Truszczynski, M. *Revision Programming.* Theoretical Computer Science, 190 (1998), 241-277

Mariničová, E. *Semantic Characterization of Dynamic Logic Programming.* Diploma Thesis, Comenius University, Bratislava, 2001

Przymusinski, T., Turner, H. *Update by means of inference rules.* The Journal of Logic Programming, 30, 2, 125-143, 1997

Šefránek, J. *A Kripkean Semantics for Dynamic Logic Programing.* Logic for Programming and Automated Reasoning, Springer, 2000

Šefránek, J. *Logic program updates: a semantics of rejection.* http://www.ii.fmph.uniba.sk/ sefranek/online/rejects.ps

Šefránek, J. *A dependency theory for logic program updates*, submitted

Šiška, J. *Refined Semantics for Multidimensional Dynamic Logic Programming.* Master Thesis, Comenius University, Bratislava, 2004.