

A theory of modular and dynamic knowledge representation

Ján Šeřránek

Institute of Informatics, Comenius University, Mlynská dolina, 842 15 Bratislava, Slovakia, phone: (421-7) 6029 5436, e-mail: seřranek@fmph.uniba.sk

Paper Track

Conference Topics: AI Foundations of Knowledge Representation, Reasoning Models: Non-monotonic Reasoning, Belief Revision.

Keywords: knowledge representation, belief revision, nonmonotonic reasoning, nonmonotonic/dynamic/modular knowledge base, multidimensional dynamic logic programming, dynamic preferences, stable model semantics, Kripke structure

Abstract. A logic-based theory of modular and dynamic knowledge representation is presented in the paper. The theory has its origin in the paradigm of multidimensional dynamic logic programming. A knowledge base (KB) may be viewed as a multidimensional dynamic logic program (MDyLoP). MDyLoP consists of a set of generalized logic programs (modules) and of a preference relation (defined on the modules). Knowledge evolution is modeled according to a causal rejection principle (if there is a conflict between rules, then more preferred rules override those less preferred).

The paper is devoted to the semantic foundations of modular and dynamic knowledge representation. The semantics is focused on the conflicts between belief sets instead of the conflicts between rules. We do not intend to construct and study a syntactic variant of the modular knowledge base \mathcal{P} , which represents a changed situation, the change is represented on semantic level. The meaning of each module is represented by a Kripke structure. An update operation on Kripke structures is defined.

We assume that a modular KB is embedded in an environment. The events in the environment are reflected by a dynamics of the KB. An event determines the dominant module and exerts an influence on the preference relation. The meaning of the whole KB is changed, if the dominant module or the preference relation is changed. The changed meaning is specified by the update operation on Kripke structures. Question answering is based on the semantics.

1 Introduction

Logic programming has been proved as a powerful framework suitable for theoretical investigation of knowledge representation, see for example [4, 5]. However, first attempts to give a logical characterization of modularity and dynamics of knowledge representation (undoubtedly the features of crucial importance) appeared only recently. Multidimensional dynamic logic programming (MDyLoP) [3, 8] seems to be a promising candidate for a logic-based theory of modular and dynamic knowledge representation.

A knowledge base in the frame of MDyLoP may be seen as a set of modules $\mathcal{P} = \{P_i : i = 1, 2, \dots\}$. Moreover, a preference relation is defined on \mathcal{P} . Suppose that a unique dominant (current) module P_s is activated after an event (in a time moment).¹ The content of \mathcal{P} should be revised from the viewpoint of P_s . The revision is “implemented” in [3, 8] by a syntactic transformation and it respects the causal rejection principle, [7]: if there is a conflict between *rules*, then more preferred rules override those less preferred. The result of this transformation is denoted by $\oplus_s \mathcal{P}$.

Our approach is different. While the attention of MDyLoP is focused on the conflicts between rules (syntactic objects), our attention is focused on the conflicts between belief sets (semantic objects). A Kripkean semantics of dynamic knowledge bases has been proposed. It was introduced in [10], a modified version was presented in [11]. The semantics enables to express dependencies between sets of literals (belief sets). An update operation is defined on Kripke structures, it resolves the conflicts between belief sets and it is sensitive to the dependencies between belief sets. The semantics of [11] is improved and a also its general version is presented first time in this paper.

This paper is devoted to an application of this semantics to a characterization of modular and dynamic knowledge representation. We assume that a KB (a set of modules together with a preference relation) is embedded in an environment (similarly as in [2]). A dynamics of the modular KB reflects the events in the environment: an event determines the dominant module, moreover, the preference relation on modules may be modified, too. The dynamics of the KB is represented only on the semantic level: The meaning of the whole KB is changed if the dominant module or the preference relation is changed. The queries addressed to an evolving knowledge base are responded (in a given situation) with respect to the (changing) semantics.

It was shown in [11] that our approach handles the conflicts between rules correctly, moreover it is able to handle the conflicts not distinguishable by the traditional approach of MDyLoP.

The main contributions of this paper: a purely semantic characterization of modular and dynamic knowledge representation, an improved and generalized approach to updates on Kripke structures which enables to recognize the conflicts between belief sets and to deal with the dependencies between the belief sets.

¹ This decision may be modified: a set of dominant modules can be considered.

The paper is structured as follows. First, a model (an idealization) of modular and dynamic knowledge representation is presented in the Section 2. The Kripkean semantics of generalized logic programs is described in the Section 3. The updates on Kripke structures are presented in 4. Finally, the results of the paper are summarized and open problems are sketched in the Conclusions.

2 Modular and dynamic knowledge base

In this Section we outline (an idealization of) a modular and dynamic knowledge representation.

A set \mathcal{A} of propositional atoms is assumed. The set \mathcal{A}_{not} is defined as $\mathcal{A} \cup \{not A : A \in \mathcal{A}\}$. Each member of \mathcal{A}_{not} is called literal.

A *generalized clause* is a formula c of the form $L \leftarrow L_1, \dots, L_k$, where L, L_i are literals. We will denote L also by $head(c)$ and the conjunction L_1, \dots, L_k by $body(c)$. A non-empty set of generalized clauses is called a *generalized logic program*. In the following, whenever we use “clause” or “program” we mean “generalized clause” and “generalized logic program”, respectively.

The language \mathcal{L} is the set of all clauses (over \mathcal{A}).

Multidimensional dynamic logic program is defined as a set of generalized logic programs together with a preference relation on the programs [8]. The relation enables to represent dynamic aspects of knowledge.

Definition 1 ([8]) A *multidimensional dynamic logic program* is a pair (\mathcal{P}, G) , where $G = (V, E)$ is an acyclic digraph, $|V| \geq 2$, and $\mathcal{P} = \{P_v : v \in V\}$ is a set of generalized logic programs.

We denote by $v_i \prec v_j$ that there is a directed path from v_i to v_j and $v_i \preceq v_j$ means that $v_i \prec v_j$ or $i = j$. If $v_i \prec v_j$, we say that P_{v_j} is *more preferred* than P_{v_i} .

Remark 2 If $|V| = 2$ and $E = \{(v_1, v_2)\}$, we get an important basic case: P_{v_1} may be considered as an original program and P_{v_2} as an updating (new) program. Similarly, an idea of a temporal evolution of a knowledge base can be modeled by $|V| = k$ and $E = \{(v_1, v_2), \dots, (v_{k-1}, v_k)\}$.

An addition of new programs (modules) and a removal of some programs can be modeled by changing (“dynamic”) sets V and E . \square

Possible conflicts between the rules of different programs are solved according to a causal rejection principle – the rules from more preferred programs override those from less preferred, [7].

One of the programs, say P_s , from \mathcal{P} is dominant in a time moment, it is assumed that P_s represents the current viewpoint (or current situation).

Our goal is to specify – on a semantic level – the current meaning of \mathcal{P} with respect to P_s and to the preference relation E . In consequence, if the current viewpoint is changed or the preference relation is changed, the meaning of \mathcal{P} is changed.

A (modular) *knowledge base* may be represented by a multidimensional dynamic logic program.

It is assumed that a multidimensional logic program $KB = (\mathcal{P}, G)$ is situated in an environment. The environment is represented by a subset of \mathcal{L} , too. The clauses in the environment are of two types: events (of the form $L \leftarrow$) and queries (of the form $\leftarrow L_1 \dots, L_k$). We denote the set of events by \mathcal{E} and the set of queries by \mathcal{Q} .

The events do not persist by inertia, similarly as in [1]. The set of events \mathcal{E} is changing in a discrete and linear time. We assume that in each time moment $t \in \text{Nat}$, where Nat is the set of natural numbers, there is a non-empty set \mathcal{E}_t of events. Only the events recorded in \mathcal{E}_t holds in the time moment t . For each pair of subsequent time-moments $t, t + 1$ holds that $\mathcal{E}_t \neq \mathcal{E}_{t+1}$. And conversely, each modification of \mathcal{E} “increments the time”: if \mathcal{E}' is the immediate result of a modification of \mathcal{E} , then there is a time moment t such that $\mathcal{E}_t = \mathcal{E}$ and $\mathcal{E}_{t+1} = \mathcal{E}'$.

The dynamics of the environment exerts influence on the (dynamics of the) knowledge base:

- the dominant (current) module may be changed,
- the preference relation on the modules may be changed.

This influence may be modeled again by a program \mathcal{M} . The role of \mathcal{M} is to specify the dominant module P_s and to modify the set of edges E of the graph G (the relation of preference). \mathcal{M} is considered as a meta-knowledge base (a control mechanism) and predicates *current*/1 and *edge*/2 are defined in \mathcal{M} . Of course, \mathcal{M} contains the postulate

$$\text{not edge}(X, Y) \leftarrow \text{current}(X),$$

for each module Y : if X is a current (dominant) module, then X is not less preferred than Y .

The events are matched against the bodies of the rules in \mathcal{M} . Each possible event determines a current module. Moreover, the preference relation on the modules may be (re)defined when an event occurs. The graph $G = (V, E)$ is a dynamic one, the set of edges depends on the current event(s).

We do not intend to construct and study a syntactic variant of the modular knowledge base $KB = (\mathcal{P}, G)$, which represents the current situation. We are aiming to represent the change (of the current module and of the preference relation) on the semantic level. Similarly, the (correct) answers to the queries are specified on the semantic level. The queries (in \mathcal{Q}) are answered with respect to a (dynamic) semantics assigned to (\mathcal{P}, G) and P_s . We are now aiming to present the basic features of the semantics.

The meaning of $KB = (\mathcal{P}, G)$, where $G = (V, E)$, depends on the current program (module) P_s and the information from less preferred modules (programs) is inherited if it is not in a conflict with the information from more preferred modules. It is important to emphasize: when the current module is changed or the preference relation is changed, also the meaning of \mathcal{P} is changed.

We are going to specify the conditions for the semantic characterization of \mathcal{P} . First, some preliminaries: For each $A \in \mathcal{A}$, A and *not* A are called *conflicting literals*. There is a function (bijection) assigning to each literal L the conflicting literal: $\overline{A} = \text{not } A$ and $\overline{\text{not } A} = A$. A set of literals is *consistent*, if it does not contain a pair of conflicting literals. *Partial interpretation* (of a language \mathcal{L}) is a consistent subset of \mathcal{A}_{not} . *Total interpretation* is a partial interpretation I such that for each $A \in \mathcal{A}$ either $A \in I$ or $\text{not } A \in I$.

Let σ be an assignment of a set of total interpretations to a of programs \mathcal{P} , to corresponding preference relation E and to the current module P_s . We may consider σ as a semantic characterization of \mathcal{P} (w.r.t. P_s and E), slightly adapting the concept of [6]. The corresponding set of interpretations is denoted by $\sigma(\mathcal{P}, P_s, E)$.

The queries addressed to the KB (in a time-moment t) are responded with respect to $\sigma(\mathcal{P}, P_s, E)$. The set represents the current meaning of the knowledge base.

Let L be a literal, then $\sigma(\mathcal{P}, P_s, E) \models_{\text{cred}} L$ iff there is $I \in \sigma(\mathcal{P}, P_s, E)$ such that $L \in I$ and $\sigma(\mathcal{P}, P_s, E) \models_{\text{scept}} L$ iff for each $I \in \sigma(\mathcal{P}, P_s, E)$ holds that $L \in I$.

If we abstract from local changes² of modules, then the changing environment is mirrored by the changing semantic representation of the knowledge base. The semantics σ enables to respond in each state of the environment \mathcal{E}_i each query $Q = \leftarrow L_1, \dots, L_k$. We respond to Q by *true* w.r.t. $\sigma(\mathcal{P}, P_s, E)$ iff for each L_i holds $\sigma(\mathcal{P}, P_s, E) \models_{\text{scept}} L_i$, w.r.t. the skeptical attitude (similarly w.r.t. the credulous attitude).

A policy of defining σ is considered in the Section 4. The power of the policy is in the ability to record dependencies between belief states and, in consequence, in the ability to specify updates on a semantic level. The definition of \models and of query answering may be viewed only as an appendix of the proposed semantics.

3 Kripke structure associated with a program

We recap the basic concepts and results from [10] (with some modifications from [11]), in order to make this paper self-contained.

If w is a set of literals, then by \overline{w} we mean the set of literals $\{\overline{l} : l \in w\}$ and $w^- = \{\text{not } A : \text{not } A \in w\}$. The set $\{\text{not } A : A \in \mathcal{A}\}$ will be denoted by \mathcal{D} .

We accept a convention as follows: All programs considered below use only propositional symbols from \mathcal{A} . Similarly, we assume a common set of (partial) interpretations. We denote the set by $\text{Int}_{\mathcal{A}}$. It means, an interpretation of a program P may contain a propositional symbol not occurred in P . Our specification of updates uses the convention.

Let I be a partial interpretation. A literal L is *satisfied* in I iff $L \in I$. A conjunction of literals of the form L_1, \dots, L_k is satisfied in I iff each L_i is satisfied

² A (local) modification of modules $P_i \in \mathcal{P}$ is allowed. We do not devote an attention to this topic in this paper, however.

in I . A clause c of the form $L \leftarrow L_1, \dots, L_k$ is satisfied in I iff L is satisfied in I whenever each L_i is satisfied in I . Notation: $I \models L$, $I \models L_1, \dots, L_k$, $I \models c$. A partial interpretation I is a *model* of a program P iff for each clause $c \in P$ holds $I \models c$.

Notice that propositional generalized logic programs may be treated as Horn theories. The least model of the Horn theory H we denote by $least(H)$.

Definition 3 Let P be a program. A Kripke structure \mathcal{K}_P associated with P is a pair (W, ρ) , where:

- $W = Int_{\mathcal{A}} \cup \{w_{\perp}\}$, 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 $w \neq w'$, and it holds either $w' = w_{\perp}$ or $w' = w \cup u$, where u is a set of literals.
- $\rho = \rho_1 \cup \rho_2 \cup \rho_3$, where
 1. $(w, w') \in \rho_1$ iff $w' = w \cup \{head(c)\}$ for some $c \in P$ such that $w \models body(c)$,
 2. $(w, w') \in \rho_2$ iff w' is consistent, $w' = w \cup u$, where $\emptyset \neq u \subseteq \mathcal{D}$, and there is no rule $r \in P$ such that $w \models body(r)$, but there is a rule $r' \in P$ such that $w' \models body(r')$ and $u \subseteq body(r')$,
 3. $(w, w') \in \rho_3$ iff w is not a total interpretation, there is no pair $(w, v) \in \rho_1 \cup \rho_2$, $w' = w \cup u$, where $\emptyset \neq u \subseteq \mathcal{D}$ and w' is a total interpretation.

Three types of edges enable to distinguish three types of default negations:

1. the derived default negations (occurring in the heads of the rules),
2. the used default negations (occurring in the bodies of the rules), they may be called *what-if nonmonotonic assumptions*,
3. and default negations, which may be called *completion-based nonmonotonic assumptions*.

Remark 4 Suppose that $w \in W$, $w \neq w_{\perp}$, $r \in P$, $w \models body(r)$, and $\overline{head(r)} \in w$. Then $(w, w_{\perp}) \in \rho_1$.

Definition 5 Let τ be a binary relation. Then τ -path is a sequence σ of pairs $(w_0, w_1), \dots, (w_{n-1}, w_n)$ (denoted by $\langle w_0, \dots, w_n \rangle$), if each $(w_i, w_{i+1}) \in \tau$.

We say that this σ is *rooted* in w_0 . If there is no pair $(w_n, w) \in \tau$, we say that σ is *terminated* in w_n .

We are now ready to state (in terms of nodes and paths in \mathcal{K}_P) conditions of being a stable model of a program P .

Definition 6 Let P be a program, σ be a ρ -path $\langle w_0, w_1, \dots, w_n \rangle$ in \mathcal{K}_P . We say that σ is *correctly rooted*, if $w_0 = \emptyset$.

Definition 7 (Stable model, [3]) Let P be a generalized logic program and w be an interpretation of P . It is said that w is a stable model of P iff $w = least(P \cup w^-)$.

Definition 8 (Distinguished paths, good worlds) A correctly rooted ρ -path σ terminated in a total interpretation w is called a *distinguished* path and w is called a *good world*.

Theorem 9 Let P be a program, \mathcal{K}_P be the Kripke structure associated with P , $\sigma = \langle w_0, w_1, w_2, \dots, w_{n-1}, w_n \rangle$ be a distinguished ρ -path in \mathcal{K}_P .
Then (a good world) w_n is a stable model of P .

Theorem 10 Let S be a stable model of a generalized logic program P and \mathcal{K}_P be a Kripke structure associated with P .
There is a distinguished ρ -path $\sigma = \langle w_0, \dots, w_n, S \rangle$ in \mathcal{K}_P .

Theorem 11 Let P and \mathcal{K}_P be as in the Theorem 10. If w_n is a total interpretation and $(w_n, w_\perp) \notin \rho$, then w_n is a model of P .
If M is a model of P , then $(M, w_\perp) \notin \rho$.

4 Updated Kripke structures

This Section is based on [11], however some important modifications are presented. In what follows we suppose only programs with a stable model.

4.1 The basic case

It is assumed in this Subsection that $\mathcal{P} = \{P, U\}$, and that the Kripke structures, $\mathcal{K}_P = (W, \rho^P)$ and $\mathcal{K}_U = (W, \rho^U)$, associated with P and U , respectively, are given. Moreover, U (the updating program) is more preferred than P (the original program).

We intend to define an operation \oplus on Kripke structures. The resulting Kripke structure $\mathcal{K}^{U \oplus P} = \mathcal{K}^U \oplus \mathcal{K}^P = (W, \rho^{U \oplus P})$ should be based on \mathcal{K}^U while a reasonable part of \mathcal{K}^P is preserved. Notice that the set of nodes, W , remains unchanged, but some edges should be rejected. The main goal of the rest of the paper is to motivate and to explain the “strategy of rejections”.

A removal of some edges may be interpreted as overriding the corresponding dependencies between belief sets. On the other hand, connecting the edges from one Kripke structure to the edges from another may be interpreted as a (semantic) construction based on the compatible dependencies between belief sets.

Definition 12 (Attacked edges, [9]) Let $\tau_1, \tau_2 \subseteq W \times W$ be binary relations. Let $u, v, v' \in \text{Int}_{\mathcal{A}}$ and $e = (u, v) \in \tau_1$, $e' = (u, v') \in \tau_2$.

We say that e is *attacked* by e' iff $(v \setminus u) \cap (v' \setminus u) \neq \emptyset$.

Of course, there is a symmetry: if e is attacked by e' then e' is attacked by e , too. Nevertheless, we want prefer “one side”. We usually prefer ρ^U to ρ^P , however, a more detailed analysis is needed. First, it is suitable to extend the definition of attacked edges in order to catch also w_\perp :

Definition 13 If $e' = (w, w') \in \rho_1^U$, where $w' \neq w_\perp$, and $e = (w, w_\perp) \in \rho_1^P$, then e is *attacked* by e' .

Our (general) decision is to reject all ρ_3 -edges from both Kripke structures \mathcal{K}_U and \mathcal{K}_P . Intuitively, we don't preserve the (too strong) completion-based non-monotonic assumptions.

A strategy concerning rejections of the ρ_1 - and ρ_2 -edges may be expressed in terms of some postulates (for a rational $\rho^{U \oplus P}$). Clear cases are as follows (notice that a set of edges $Rejected_{\rho^U}(\rho_P)$ is defined as a side-effect):

Definition 14 (Postulates P1 – P3) Let L be a literal, w, w_i, u, v possible worlds.

- P1** $\rho_1^U \subseteq \rho_1^{U \oplus P}$,
- P2** if $(u, w) \in (\rho_1^P \setminus \rho_1^U)$, then $(u, w) \in Rejected_{\rho^U}(\rho_P)$ iff (u, w) is attacked by $(u, w') \in \rho_1^U$,
- P3** if $(u, w) \in (\rho_2^P \setminus (\rho_1^U \cup \rho_2^U))$, then $(u, w) \in Rejected_{\rho^U}(\rho_P)$ iff (u, w) is attacked by $(u, w') \in \rho_1^U$.

Complications arise when ρ_2^U -edges are in a conflict with ρ_1^P -edges. The complications are illustrated and analyzed in [11]: it is shown that the preference of ρ_2^U -edges to ρ_1^P -edges has to be formalized carefully.³ Unfortunately, according to our current knowledge, a kind of global view is needed. The global view will be expressed in terms of independent literals.

Definition 15 (Dependent literal) A literal L is *dependent* on a literal L' in \mathcal{K}_U iff for each $\rho_1^U \cup \rho_2^U$ -path $\sigma = \langle \emptyset, \dots, w_j, \dots, w_n \rangle$ terminated in $w_n \neq w_\perp$ such that $L \in w_j$ and $L \notin w_{j-1}$ holds $L' \in w_{j-1}$.

Definition 16 (Complementary literals) Atoms A and A' are *complementary* in a Kripke structure \mathcal{K} iff both A is dependent on the nonmonotonic assumption $\overline{A'}$ and A' is dependent on the nonmonotonic assumption \overline{A} in \mathcal{K} .

We propose to handle the conflicts between nonmonotonic assumptions in \mathcal{K}_U and atoms in \mathcal{K}_P as follows: a ρ_2^U -edge justifying the nonmonotonic assumption *not* A is preferred to a ρ_1^P -edge justifying the atom A if there is an atom B complementary to A in \mathcal{K}_U , but B and A are compatible in \mathcal{K}_P .

Definition 17 (Preference on ρ_2^U - and ρ_1^P -edges) Let A be an atom, $e_P = (w, w \cup \{A\}) \in \rho_1^P$ and $e_U = (w, w \cup u) \in \rho_2^U$, where $\overline{A} \in u$. Then e_U is *preferred* to e_P if there is an atom B such that A and B are complementary in \mathcal{K}_U , but there is a ρ_1^P -path in \mathcal{K}_P from $w \cup \{A\}$ to a possible world v such that $B \in v$.

Definition 18 (Postulate P4) Let A be an atom.

³ It seems, there is a space for a variety of semantic decisions.

P4 if $e_P = (w, w \cup \{A\}) \in \rho_1^P$, $e_U = (w, w \cup u) \in \rho_2^U$, $\bar{A} \in u$, then $(w, w \cup \{A\}) \in \text{Rejected}_{\rho^U}(\rho^P)$ iff e_U is preferred to e_P .

We are going to define the update on Kripke structures. It should be (again) emphasized that there is a space for a variety of semantic decisions. The conflicts between ρ_2^U -edges and ρ_1^P -edges represent the critical point. The set $\text{Rejected}_{\rho^U}(\rho^P)$ contains only ρ_1^P -edges (according to the decision accepted in this paper).

Definition 19 (Update on Kripke structures)

$$\begin{aligned}\rho_1^{U \oplus P} &= \rho_1^U \cup (\rho_1^P \setminus \text{Rejected}_{\rho^U}(\rho^P)) \\ \rho_2^{U \oplus P} &= \rho_2^U \cup (\rho_2^P \setminus \text{Rejected}_{\rho^U}(\rho^P)) \\ \rho^{U \oplus P} &= \rho_1^{U \oplus P} \cup \rho_2^{U \oplus P} \cup \rho_3^{U \oplus P} \\ \mathcal{K}^U \oplus \mathcal{K}^P &= \mathcal{K}^{U \oplus P} = (W, \rho^{U \oplus P})\end{aligned}$$

The relation $\rho_3^{U \oplus P}$ is defined according to the definition 3. The definition of $\mathcal{K}^{U \oplus P}$ is sound in a sense – special cases of the facts 22, 23, 24 from the Subsection 4.2 shows that

- the rejected rules (according to the causal rule rejection principle) do not “generate” edges in the updated Kripke structures,
- the information of U is represented in the good worlds of the updated Kripke structure $\mathcal{K}_{U \oplus P}$,
- finally, an update on “reasonable” Kripke structures provides a reasonable Kripke structure.

4.2 The general case

We present now the definition of the Kripke structure $\mathcal{K}_{\oplus_s \mathcal{P}}$, which is the result of the update operation on Kripke structures associated with programs in a multidimensional dynamic logic program (\mathcal{P}, G) , where $G = (V, E)$. We denote for each $P_i \in \mathcal{P}$ the associated Kripke structure by $\mathcal{K}_{P_i} = (W, \rho_{P_i})$. A current (dominant) program is denoted by P_s .

Definition 20 Let $P_u \in \mathcal{P}$. An accessibility relation $\text{below}(\rho_i^{P_u})$, where $i = 1, 2$, is defined as follows:

- \emptyset , if there is no v such that $(v, u) \in E$,
- $\bigcup_v (\rho_i^{P_v} \setminus \text{Rejected}_{\rho_{P_u}}(\rho_{P_v})) \cup \text{below}(\rho_i^{P_v})$, where $v \in V$, $(v, u) \in E$.

Definition 21 (Update on \mathcal{P} w.r.t. P_s)

$$\begin{aligned}\rho_i^{\oplus_s \mathcal{P}} &= \rho_i^{P_s} \cup \text{below}(\rho_i^{P_s}), i = 1, 2 \\ \rho^{\oplus_s \mathcal{P}} &= \rho_1^{\oplus_s \mathcal{P}} \cup \rho_2^{\oplus_s \mathcal{P}} \cup \rho_3^{\oplus_s \mathcal{P}} \\ \mathcal{K}_{\oplus_s \mathcal{P}} &= (W, \rho^{\oplus_s \mathcal{P}}).\end{aligned}$$

Notice (again) that $\rho_3^{\oplus_s \mathcal{P}}$ is defined by the definition 3.

Rejected rules according to the causal rejection principle do not generate edges in $\rho^{\oplus_s \mathcal{P}}$:

Fact 22 *Let $w \models \text{body}(r)$, $\text{head}(r) \notin w$, where $r \in P_j$. Suppose that there is $r' \in P_i$, where $i \prec j$ such that $\text{head}(r) = \overline{\text{head}(r')}$, $w \models \text{body}(r')$, $\text{head}(r) \notin w$. Then $(w, w \cup \{\text{head}(r')\}) \in \text{Rejected}_{\rho_{P_j}}(\rho^{P_i})$.*

Good worlds from $\mathcal{K}_{\oplus_s \mathcal{P}}$ respect the information of the dominant program:

Fact 23 *Let P_s be the dominant program in \mathcal{P} . If w is a good world in $\mathcal{K}_{\oplus_s \mathcal{P}}$, then w is a model of P_s .*

An update on “reasonable” Kripke structures provides a reasonable Kripke structure:

Fact 24 *Let P_s be the dominant program in \mathcal{P} and it has a stable model. If each P_i accessible by E from P_s has a stable model, then there is a good world in $\mathcal{K}_{\oplus_s \mathcal{P}}$.*

Query answering based on a modular and dynamic knowledge base is specified as follows.

Definition 25 Let L be a literal.

$\mathcal{K}_{\oplus_s \mathcal{P}} \models_{\text{cred}} L$ iff there is a good world w in $\mathcal{K}_{\oplus_s \mathcal{P}}$ and $L \in w$.

$\mathcal{K}_{\oplus_s \mathcal{P}} \models_{\text{scept}} L$ iff for each good world w in $\mathcal{K}_{\oplus_s \mathcal{P}}$ holds that $L \in w$.

5 Conclusions

A semantic theory of a modular and dynamic knowledge representation was presented in this paper. The theory has its origin in the paradigm of multidimensional dynamic logic programming. This paradigm focused the attention to the logical foundations of crucial problems of knowledge representation - to the problems of dynamics.

The contributions of the paper are as follows: The dynamic aspects of knowledge representation are represented on the semantic level. The syntactic form of a knowledge base is (relatively) stable and the dynamics is mirrored by the changing meaning of the knowledge base. The semantic representation of a KB is changed when the current module is changed or the preference relation on modules is changed. It is important that the attention has been shifted from the conflicts between rules to the conflicts between belief sets. A semantic characterization of dependencies between literals and between belief sets has been introduced. The dependencies are used as a basis for a semantic specification of updates. A set of postulates characterizing reasonable updates has been specified. The queries addressed to the KB are responded (in a given situation) with respect to the (changed) semantics.

Some goals for the future research: An elaboration of the postulates on more abstract level. Computational aspects of this approach. Computation of updates via compilation. Extensions to the well-founded semantics and to the extended logic programs. There is a variety of semantic decisions concerning the updates of Kripke structures in the frame of stable model semantics, therefore the well-founded semantics could introduce a minimal common basis for a description of the variety. A more deep study of the variety of various possible semantic characterizations of the updates of Kripke structures. Applications of the approach to a multi-agent theory of quick, but erroneous reasoning combined with a sound and complete reasoning, when it is needed. The reasoning specified by our semantics has been characterized in a draft of this paper as prioritized default reasoning. This characterization has been omitted because of the limited size of the paper. It will be presented in a future paper.

\mathcal{M} (the metaknowledge base, the control mechanism) is in this paper a single program. The current semantic representation of KB depends on the current state of the environment. If the control mechanism itself is a multidimensional dynamic logic program, then the current semantic representation of KB may depend on the history of the environment.

Acknowledgment: I would like thank Martin Baláz for valuable comments.

References

1. Alferes, J.J., Brogi, A., Leite, J.A., Pereira, L.M. *Evolving logic programs*. 2002
2. Alferes, J.J., Brogi, A., Leite, J.A., Pereira, L.M. *Environment-aware computations via program updates*. LOPSTR'01
3. Alferes, J.J., Leite, J.A., Pereira, L.M., Przymusinska, H., Przymusinski, T.C. *Dynamic Logic Programming*. Proc. KR'98, 1998
4. Baral, C., Gelfond, M. *Logic Programming and Knowledge Representation*. Journal of Logic Programming 1994:19, 20: 73-148
5. Dix, J., Brewka, G. *Knowledge Representation with Logic Programs*. Universität Koblenz-Landau, 1996
6. Dix, J. *A classification theory of semantics of normal logic programs I: Strong properties*. Fundamenta informaticae, XXII (3):227-255, 1995
7. Eiter, T., Fink, M., Sabbatini, G., Tompits, H. *On properties of update sequences based on causal rejection*. 2001
8. Leite, J.A., Alferes, J.J., Pereira, L.M. *Multi-dimensional dynamic knowledge representation*. In: Eiter, T., Faber, W., Truszczyński (Eds.): LPNMR 2001, Springer, 365-378
9. Mariničová, E. *Semantic Characterization of Dynamic Logic Programming*. Diploma Thesis, Comenius University, Bratislava, 2001
10. Šeřfránek, J. *A Kripkean Semantics for Dynamic Logic Programming*. Logic for Programming and Automated Reasoning, Springer, 2000
11. Šeřfránek, J. *Considerations on dynamic knowledge bases*. <http://www.ii.fmph.uniba.sk/~sefranek/recent/conSem.ps>