Irrelevant Updates of Nonmonotonic Knowledge Bases

Ján Šefránek and Jozef Šiška[1]

1 INTRODUCTION

The second postulate of Katsuno and Mendelzon, 2[KM] hereafter, characterizes irrelevant updates: if ψ implies μ, then $\psi \diamond \mu$ is equivalent to ψ, where ψ is a knowledge base, μ is an update and $\diamond$ is an update operator [3].

We show that the postulate has to be modified, if nonmonotonic assumptions are considered. Our characterization of irrelevant updates is based on a dependency framework [5], which provides an alternative semantics of multidimensional dynamic logic programming (MDyLP). MDyLP [4] can be viewed as a logical idealization of nonmonotonic knowledge bases (NMKB).

Unwanted generation of new models caused by cyclic or tautological updates has been a serious problem of MDyLP for a time. Recently, the problem has been solved for dynamic logic programs in [1] and for the general MDyLP in [2]. The solution of [1] is based on Refined Extension Principle with ambition to express features essential for a "right" semantics of logic program updates. Unfortunately, the principle has not been extended to the general case of MDyLP and the trivial semantics assigning empty set of models to each dynamic logic program satisfies the principle, see [2].

We believe that the role of nonmonotonic assumptions (and dependencies on assumptions) is crucial for understanding the semantic problems of NMKBs and also for understanding unwanted generation of models.

2 DEPENDENCY FRAMEWORK

We assume a language $\mathcal{L}$ with a set of atoms $\mathcal{A}$. In general, a language of the dependency framework contains a set of assumptions. In this extended abstract are assumptions represented by a set of negative literals $\mathcal{D} = \{not\ A \mid A \in \mathcal{A}\}$, where not is a default negation. The set of literals is $Lit = \mathcal{A} \cup \mathcal{D}$.

Definition 1 A *dependency relation* is a set of pairs $\{(L, W) \mid L \in Lit, W \subseteq Lit, L \notin W\}$.

$Xs \subseteq \mathcal{D}$ is called a *sound set of assumptions* (SSOA) w.r.t. the dependency relation $\ll$ iff the set $Cn_{\ll}(Xs) = \{L \in Lit \mid L \ll Xs\} \cup Xs$ is non-empty and consistent.

It is said that a SSOA Xs is *total* (TSSOA) iff for each $A \in \mathcal{A}$ holds either $A \in Cn_{\ll}(Xs)$ or $not\ A \in Cn_{\ll}(Xs)$. The set of all (T)SSOAs w.r.t. $\ll$ is denoted by $(T)SSOA(\ll)$. □

Each NMKB consisting of rules can be mapped into the dependency framework. We will present a mapping of generalized logic programs (programs hereafter) and multidimensional dynamic logic programs (multiprograms hereafter) into the dependency framework.

[1] Comenius University, Bratislava, Slovakia, email: [sefranek,siska]@fmph.uniba.sk

A program consists of a set of rules of the form $L_0 \leftarrow L_1, \ldots, L_k$, where L_i is a literal. L_0 is denoted by $head(r)$ and $\{L_1, \ldots, L_k\}$ by $body(r)$, if a rule r is considered. A multiprogram is a set of programs together with a preference relation – a partial order on the set of programs. If there are some conflicts between programs, they are solved according to the preference relation – information from the less preferred programs is rejected.

Definition 2 A literal L *depends* on a set of literals W, $L \notin W$, *with respect to a program* P ($L \ll_P W$) iff there is a sequence of rules $\langle r_1, \ldots, r_k \rangle$, $k \geq 1$, and $head(r_k) = L$, $W \models body(r_1)$, for each i, $1 \leq i < k$, $W \cup \{head(r_1), \ldots, head(r_i)\} \models body(r_{i+1})$.

It is said that the dependency relation $\ll_P$ is *generated* by the program P. □

Notice that a literal cannot depend on itself (also in a context of other literals). Next theorem shows that a semantics of the dependency framework based on TSSOAs is equivalent to stable model (answer set) semantics of logic programs.

Theorem 3 *X is a TSSOA w.r.t. $\ll_P$ iff $Cn_{\ll_P}(X)$ is a stable model of P. Let S be a stable model of P. Then there is $X \subseteq \mathcal{D}$, a TSSOA w.r.t. $\ll_P$ such that $S = Cn_{\ll_P}(X)$.* □

Next example illustrates Theorem 3 and it also shows that dependencies in multiprograms are well defined. Only the most simple multiprograms of the form $\langle P, U \rangle$, where U is more preferred (updating) program are considered here because of the limited space. However, the results are extendable to the general case.

Example 4 $P = \{a \leftarrow not\ b; b \leftarrow not\ a\}$, $U = \{not\ b \leftarrow not\ a\}$.

There are two TSSOAs w.r.t. $\ll_P$: $Xs_1 = \{not\ b\}$, $Xs_2 = \{not\ a\}$. $Cn_{\ll_P}(Xs_1) = \{not\ b, a\}$, $Cn_{\ll_P}(Xs_2) = \{not\ a, b\}$. Note that both TSSOAs generate (all) stable models of P.

$P \cup U$ is a program and it generates a dependency relation. Observe that $a \ll_{P \cup U} \{not\ a\}$, but $(a, \{not\ a\}) \notin (\ll_P \cup \ll_U)$. □

Proposition 5 *Let $\langle P, U \rangle$ be a multiprogram. Then $\ll_{P \cup U}$ is well defined. It holds $(\ll_P \cup \ll_U) \subseteq \ll_{P \cup U}$, but the converse inclusion does not hold.*

Multiprograms contain in general conflicts. Example 6 illustrates notions connected to conflicts and solutions of conflicts.

Example 6 Dependency relation $\ll_{P \cup U}$ from Example 4 contains conflicts $C_1 = \{(b, \{not\ a\}), (not\ b, \{not\ a\})\}$ and $C_2 = \{(a, \{not\ a\})\}$.

A solution of a conflict is a set of dependencies which should be rejected (ignored) in order to remove the conflict. $D =$

$\{(not\ \ b, \{not\ \ a\})\}$ is a minimal solution of C_1. However, it is more suitable to reject less preferred dependencies. Hence, $D' = \{(b, \{not\ \ a\})$ is a more suitable solution of C_1 than D. A solution of a conflict is called a good solution of it if there is not a more suitable solution. D' is the good solution of C_1.

We intend to define the semantics of a multiprogram (with the dependency relation $\ll$) as a subset of $\ll$, which provides a coherent view on the multiprogram – i.e. there is a TSSOA w.r.t. the subset. In our example the set of assumptions $\{not\ a, not\ b\}$ is a TSSOA w.r.t. $View = \ll_{P\cup U} \setminus \{(b, not\ a)\}$ and $\{not\ b\}$ is a TSSOA w.r.t. $\ll_{P\cup U}$. □

Definition 7 (Semantics of multiprograms) A dependency relation $\ll$ is called *coherent* iff there is a TSSOA w.r.t. $\ll$.

Semantics of multiprograms (of the form $\langle P, U\rangle$) is a mapping Σ which assigns to $\langle P, U\rangle$ the set of all pairs of the form $(Z, View)$, where $View$ is a coherent subset of $\ll_{P\cup U}$ and Z is a TSSOA w.r.t. $View$.

3 IRRELEVANT UPDATES

2[KM] is adapted for multiprograms of the form $\langle P, U\rangle$ as follows: if $P \models_{SM} U$, then $TSSOA(P \cup U) = TSSOA(P)$, where $P \models_{SM} U$ means that U is satisfied in all stable models of P. Unfortunately, 2[KM] is not acceptable literally for multiprograms (and for NMKBs), see Examples 10 and 11. 2[KM] should be modified for NMKBs in order to catch the role of nonmonotonic assumptions.

Convention 8 Let $\langle P, U\rangle$ be a multiprogram. We would call U an *irrelevant update* of P iff $TSSOA(\ll_{P\cup U}) = TSSOA(\ll_P)$. □

We are aiming to find sufficient and necessary conditions of $TSSOA(\ll_{P\cup U}) = TSSOA(\ll_P)$ and to define irrelevant updates in terms of those conditions.

First, consider the condition $P \models_{SM} U$. It is a necessary condition of an irrelevant update. On the other hand, if $P \not\models_{SM} U$, then update U is a relevant one.

Proposition 9 *If* $P \not\models_{SM} U$, *then* $TSSOA(\ll_P) \not\subseteq TSSOA(\ll_{P\cup U})$. *If* $TSSOA(\ll_{P\cup U}) = TSSOA(\ll_P)$, *then* $P \models_{SM} U$. *If* $P \models_{SM} U$, *then* $TSSOA(\ll_P) \subseteq TSSOA(\ll_{P\cup U})$.

Unfortunately, Examples 10 and 11 show that $TSSOA(\ll_{P\cup U}) \subseteq TSSOA(\ll_P)$ does not hold in general, if $P \models_{SM} U$.

Example 10

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

$P \models_{SM} U$, but U introduces a new assumption $not\ a$, which is false in all stable models of P and generates a new stable model of $P \cup U$.

Example 11 $P = \{a_1 \leftarrow not\ b_1; b_1 \leftarrow not\ a_1; a_2 \leftarrow not\ b_2; not\ b_2 \leftarrow not\ a_2\}, U = \{b_2 \leftarrow not\ a_2\}$.

The set of assumptions $\{not\ \ a_2\}$ is a SSOA w.r.t. $\ll_U$ and $Cn_{\ll_P}(\{not\ \ a\}) \cup Cn_{\ll_U}(\{not\ \ a\})$ is inconsistent. The inconsistency can be overridden if we prefer $b_2 \ll_U \{not\ \ a_2\}$ over $not\ b_2 \ll_P \{not\ \ a_2\}$. Hence, $\{not\ \ a_2\}$ generates new stable models: $\{not\ a_2, not\ b_1\}$ and $\{not\ a_2, not\ a_1\}$ are TSSOAs w.r.t. $\ll_{P\cup U} \setminus \{(not\ b_2, not\ a_2)\}$. □

We proceed to the definition of irrelevant updates. An update U of P is irrelevant if $P \models_{SM} U$ and there is no set of assumptions, which is sound w.r.t. $\ll_U$, some literals are dependent on it, it is not falsified, not satisfied in P (in order to gain something new w.r.t. $P \models_{SM} U$), and which generates a new model (by extending the set of reasonable assumptions, see Example 10, or by solving a conflict, see Example 11).

Definition 12 An assumption $not\ \ A$, where $A \in \mathcal{A}$, is *falsified* in a dependency relation $\ll$ iff $A \ll \emptyset$, $not\ \ A \not\ll \emptyset$ and $\emptyset$ is a SSOA w.r.t. $\ll$.

A set of assumptions $Xs \subseteq \mathcal{D}$ is falsified in $\ll$ iff it contains a literal falsified in $\ll$. □

Definition 13 (Irrelevant update) Let $\langle P, U\rangle$ be a multiprogram, P be coherent and $P \models_{SM} U$.

It is said that U is an irrelevant update of P iff there is no $Xs \subseteq \mathcal{D}$ such that:

- $Xs \in SSOA(\ll_U)$ and $Cn_{\ll_U}(Xs) \setminus Xs \neq \emptyset$,
- Xs is not falsified in $\ll_{P\cup U}$,
- Xs is false in each stable model of P,
- there is $Bss \subseteq \mathcal{D}$ s.t. $Xs \subseteq Bss$ and $Bss \in TSSOA(\ll_{P\cup U})$ or
  - $Cn_{\ll_P}(Xs) \cup Cn_{\ll_U}(Xs)$ is inconsistent,
  - there is $View \subseteq \ll_{P\cup U}$ and $Bss \subseteq \mathcal{D}$ such that $Xs \subseteq Bss$ and $Bss \in TSSOA(View)$ □

Observe that the condition $Cn_{\ll_U}(Xs) \setminus Xs \neq \emptyset$ eliminates unsupported cyclic updates. A nondeterministic algorithm for computation of TSSOAs is presented in [5].

Theorem 14 *If U is an irrelevant update of P, then*

$$TSSOA(\ll_{P\cup U}) = TSSOA(\ll_P).$$

4 CONCLUSIONS

Main contributions of this work are as follows:

- analysis and definition of irrelevant updates,
- an adaptation of 2[KM] for updates of NMKB,
- we show that irrelevant updates do not generate new models and that models of the original program are preserved after an irrelevant update.

ACKNOWLEDGMENTS

This work was supported under grants APVV-20-P04805 and VEGA 1/3112/06. We are grateful to one of the anonymous reviewers for his comments.

REFERENCES

[1] Alferes, J.J., Banti, F., Brogi, A., Leite, J.A.: The refined extension principle for semantics of dynamic logic programming. Studia Logica **1** (2005)

[2] Banti, F., Alferes, J.J., Brogi, A., Hitzler, P.: The well supported semantics for multidimensional dynamic logic programs. LPNMR 2005, LNCS 3662, Springer, 356-368

[3] Katsuno, H., Mendelzon, A.O.: On the difference between updating a knowledge base and revising it. Proc. of KR'91

[4] Leite, J.A., Alferes, J.J., Pereira, L.M.: Multi-dimensional dynamic logic programming. In: Procs. of CLIMA'00. (2000) 17–26

[5] Šefránek, J.: Rethinking semantics of dynamic logic programming. To appear in Proc. of NMR 2006