Agents with incomplete and evolving knowledge

Ján Šefránek
Comenius University, Bratislava, Slovakia,
sefranek@fmph.uniba.sk
- knowledge representation and reasoning research

- reasoning with evolving and incomplete knowledge

- a modular approach to development of intelligent agents

- logic programming languages and specification of agents with incomplete and evolving knowledge

- KR&R research at our group
KR & R research
a dominating research stream in AI
(f.ex. 10 of 40 sessions at ECAI 2006 dedicated to KR & R)

the leading paradigm: formal models of some aspects of reasoning; a kind of logic is used in a large part of papers

a changed attitude to AI:
we are attempting to **understand** important features of reasoning (instead of aiming at “machines with common sense” etc.)
KR & R:
important research topics and a new methodology:
computer science

new points of view and essentially new kinds of results
thanks to the computational paradigm
Some of main topics:

- exception tolerant and inconsistency-tolerant reasoning
- temporal reasoning, spatial reasoning, causal reasoning, explanations
- reasoning about actions
- reasoning under uncertainty
- representations of vagueness
• information change, belief revision, update

• description logics, ontologies

• reasoning about preference

• learning, discovering and acquiring knowledge, summarization, categorization
reasoning with evolving and incomplete knowledge
a short historical overview (selected points):

John Mc Carthy’s visions

non-monotonic logics (80th)

logic programming, deductive databases and nonmonotonic reasoning (the end of 80th, 90th)

answer set programming
nonmonotonic logics: expressive power

Cadoli et al.: Is intractability of nonmonotonic reasoning a real drawback?

succinct, compact representations
non-monotony of reasoning:

$$\exists A, B \ (A \subset B \land Cn(A) \not\subset Cn(B))$$

van Benthem: non-monotony is a symptom, not the essence of non-standard reasoning

$$\exists \phi, \psi \ (\phi \in (B \setminus A) \land \psi \in (Cn(A) \setminus Cn(B)))$$

a delete enforced by an insert (by a completion of incomplete knowledge)

updates and revisions – evolving and incomplete knowledge
a modular approach to development of intelligent agents
agent (here:-)
a metaphor for modelling of (a simple) intelligent behaviour based on reasoning and knowledge

how to specify an intelligent behaviour?

the level of request that an entity can handle – a benchmark of intelligence
a spectrum of possibilities

from a fully imperative specification

to a fully declarative specification

the declarative paradigm: the agent is able to reason with incomplete (evolving, sometimes inconsistent) knowledge or to fill gaps in his knowledge
a modular BDI architecture (Peter Novák)

**encapsulated** modules: beliefs, desires, intentions (and also capabilities)

the only allowed interface to the modules: **queries** and **updates**

the specification of behaviour: interaction rules of the form

\[ Q \rightarrow U \]
the modules can be implemented in different languages

an example:

an interaction rule:

\[ Q_D(make_{\text{espresso}}) \& Q_B(ready) \rightarrow U_I(push(grindboilpourclean)) \]

a possible implementation of modules:
beliefs, desires - Prolog, intentions - Lisp, capabilities - C
logic programming representation of evolving and incomplete knowledge
logic program as a knowledge base (the language as a knowledge representation language)

\[ r = L \leftarrow L_1, \ldots, L_k \]

\[ \text{head}(r) = L, \text{body}(r) = \{L_1, \ldots, L_k\} \]

default negation \textit{not}, explicit negation \neg

generalized extended logic programs

disjunctive (epistemic, prioritized, nested) programs
non-monotony of default negation

negation as finite failure

\[ P = \]

\[ a \leftarrow \]

\[ b \leftarrow c \]

? not b – yes

\[ P' = P \cup \{c \leftarrow\} \] ? not b – no
stable model (answer set) semantics

negation as finite failure does not cover the idea of default negation

\[ P = \{ a \leftarrow \neg b \\ b \leftarrow \neg a \} \]

models: \( M_1 = \{ a, b \} \), \( M_2 = \{ a, \neg b \} \), \( M_3 = \{ b, \neg a \} \)

\[ O(P, M) = \{ \text{head}(r) \mid r \in P, M \models \text{body}(r) \} \]

stable model (answer set) of \( P \): a fixpoint of \( O \)

\[ O(P, M_1) \neq M_1, O(P, M_2) = M_2, O(P, M_3) = M_3 \]

\( M_1 \) is not a stable model of \( P \), \( M_2 \) and \( M_3 \) are stable models of \( P \), \( SM(P) = \{ M_2, M_3 \} \)
(M)DyLoP

dynamic logic programming – a sequence of logic programs (a flow of time, a knowledge evolution)
a generalization: the preference relation is determined also by other aspects as time

multidimensional dynamic logic programming
EVOLP

the language of dynamic logic programming + atoms of the form \textit{assert}(r), where \( r \) is a rule

the idea – a self-evolution of an agent is encoded in the program

beliefs, desires, intentions can be modelled by an EVOLP programs

action languages
KR&R research at our group
Jozef Šiška

Dynamic Logic Programming and World State Evaluation in Computer Games.
WLP 2006, Vienna

Logic Programming and Knowledge Representation in Computer Games
Martin Homola

Dynamic Logic Programming: Various Semantics are Equal on Acyclic Programs.
In: Computational Logic in Multi-Agent Systems. LNAI 3487, Springer 2005
Martin Baláž

Well-Supported Semantics for Multidimensional Dynamic Logic Programs
Peter Drábk, Michal Malý, Peter Klimo, Martin Slota – ongoing work on Master Theses

modelling BDI agents in EVOLP
translation of EVOLP programs into ordinary logic programs and evaluation using answer set programming solvers
reductio ad-absurdum via an adaptation of stable model semantics
non-normal modal logics
Ján Šefránek


Rethinking semantics of dynamic logic programming. Proc. of NMR 2006, Lake District, UK

Irrelevant updates of nonmonotonic knowledge bases. Proc. of ECAI 2006 (together with J.Šiška)

Nonmonotonic integrity constraints. WLP, Vienna 2006

Semantic considerations on rejection. Proc. of NMR 2004, Whistler, BC, Canada
standard semantics of MDyLoP do not satisfy postulates for updates by Katsuno and Mendelzon

if \( \psi \) and \( \mu \) are satisfiable, then \( \psi \diamond \mu \) is satisfiable

\[
P = \{a \leftarrow b\} \quad U = \{b \leftarrow \text{not } a\}
\]

similarly, the second postulate is not satisfied:

if \( \psi \models \mu \) then \( \psi \diamond \mu \equiv \psi \)
dependency framework

dependency relation \( \ll \):
\[
\{(L, W) \mid L \in Lit, W \subseteq Lit, L \not\in W\}.
\]

a dependency relation can be assigned to each non-monotonic knowledge base consisting of rules (with one literal in the head and a set of literals in the body)

semantics based on assumptions and dependencies

a coherent dependency relation

goal: a coherent view on an incoherent \( \ll \)
logic programs with preferences: there are programs with (standard) answer sets, but with no preferred answer sets

ongoing research: if there is a non-empty set of standard answer sets of a program, then a preferred answer set is selected
Peter Novák, Juergen Dix: Modular BDI Architecture
Proceedings of Fifth International Joint Conference on
Autonomous Agents and Multiagent Systems AAMAS’06
Ján Senko

Eiter, Erdem, Fink, Senko: Resolving Conflicts in Action Descriptions. ECAI 2006

Eiter, Erdem, Fink, Senko: Comparing Action Descriptions Based on Semantic Preferences. JELIA 2006

Eiter, Fink, Senko: A Tool for Answering Queries on Action Descriptions. JELIA 2006

Eiter, Erdem, Fink, Senko: Updating Action Domain Descriptions. IJCAI 2005

Eiter, Fink, Senko: KMonitor - A Tool for Monitoring Plan Execution in Action Theories. LPNMR 2005