Organon F 20 (Supplementary Issue 2) 2013: 216-234

From Deduction to Knowledge Representation

MICHAL VINCE

Department of Applied Informatics. Faculty of Mathematics, Physics and Informatics
Comenius University. Mlynská dolina. 842 48 Bratislava. Slovak Republic
michal.vince@fmph.uniba.sk

JÁN ŠEFRÁNEK

Department of Applied Informatics. Faculty of Mathematics, Physics and Informatics
Comenius University. Mlynská dolina. 842 48 Bratislava. Slovak Republic
sefranek@fmph.uniba.sk

RECEIVED: 11-12-2012 • ACCEPTED: 02-03-2013

Abstract: In this paper, we discuss why deduction is not sufficient for knowledge representation of programs with commonsense. Requirements of representation of incomplete, evolutive and conflicting knowledge led to a rise of alternative logic formalisms, dubbed nonmonotonic logics. Important features of nonmonotonic logic were discussed on the example of default logic – a role of assumptions in reasoning, use of fixpoint constructions as a formal tool for building a nonmonotonic semantics and, finally, computational aspects of nonmonotonic reasoning. This overview is completed by a presentation of our approach to updates. Updates are closely connected to nonmonotonic reasoning. We construct our approach for assumption based frameworks (and for default theories, as a consequence).

Keywords: Assumption-based framework – commonsense reasoning – default logic – knowledge representation – non-monotonic logic – update.

1. Introduction

The goals of this paper are twofold. First, we are aiming at a description of logical aspects of knowledge representation. Knowledge representation is

a field of artificial intelligence, its task is to study and construct languages, formalisms and tools for expressing knowledge of "intelligent" (knowledge-based) programs (agents) and for reasoning with the represented knowledge. Foundational aspects of this task involve

- semantic specification of features of a representational language, appropriate with respect to a represented domain and required reasoning tasks,
- investigation and proposal of syntactic systems and algorithms with good computational properties, obeying semantic specifications.

It is obvious that this task has a relevant logical content. According to our opinion, the core of the knowledge representation research field is logical.

Second, we intend to provide a brief presentation of a topic of our own research – of updates of some nonmonotonic knowledge bases.

We will start from a more general perspective, emphasizing the role of logic in computer science. Mathematical logic became rather one of many mathematical disciplines after the end of a dream about firm logical foundations of mathematics in thirties of the previous century. On the other hand, may be surprisingly, influence of logic on computer science is striking. It is possible to speak about unusual effectiveness of logic for computer science (Halpern et al. 2001). More in Section 2.

Particularly, logic played an important role in the history of artificial intelligence since its beginning until current days. John McCarthy (1959) presented a vision of logic-based intelligent programs already in fifties. More in Section 3. Pioneers of artificial intelligence were aware of close relationships between artificial intelligence and philosophical logic (McCarthy – Hayes 1969).

A primary field of artificial intelligence from the viewpoint of applications of logic is the field of knowledge representation and reasoning (KRR). More importantly, KRR is a field, which motivates an emergence of new logical systems and ways how to do logic (Makinson 2002). Deduction is not a sufficient reasoning mode for tasks inherent in knowledge representation. Representation of incomplete, evolving and conflicting knowledge and reasoning with such knowledge provide new, challenging stimuli for logical research.

The rest of this paper after Section 3 is structured as follows. An overview of default logic is discussed, emphasizing some general features of that

logic – use of nonmonotonic assumptions, fixpoint constructions for specifying semantics of nonmonotonic theories and computational aspects of nonmonotonic reasoning. After that, an original approach to updates of some nonmonotonic knowledge bases is presented.

2. Logic and computer science

During the last forty years logic has gained much more influence in computer science than it ever has in mathematics. In fact, concepts and methods of logic seize unmistakable place in computer science, therefore the logic has been called "the calculus of computer science" (Manna – Waldinger 1985).

Halpern et al. (2001) described the status of logic in computer science as follows: "*…logic has turned out to be significantly more effective in computer science than it has been in mathematics. This is quite remarkable, especially since much of the impetus for the development of logic during the past one hundred years came from mathematics.*"

Logic is used as a conceptual apparatus in many fields of computer science, e.g. in complexity theory, relational databases and query languages, programming language research, program specification, program and protocol verification, automated verification of hardware designs, reasoning about knowledge, distributed processes, multi-agent systems, knowledge representation, semantic web.

In the next few paragraphs we will briefly present an example of an effective use of logic in computer science, its use in databases. Relational data model, based on logic, led to a technological turn in database field, even if there was a big distrust against the used logical apparatus.[1] Clear, precise, but intuitive and easy to use relational query languages belong among important contributions of logic for computer science, but also for information technology.

[1] Codd, the author of the relational data model, said in his Turing award lecture Codd (1970): "Instead of welcoming a theoretical foundation as providing soundness, the attitude seems to be: if it's theoretical, it cannot be practical." Notice that the title of his article is symptomatic — the logical approach to databases has been not yet generally accepted in the year 1982.

Logic as a database query language. First-order logic (FO) forms the base of several modern database systems, and the standard query languages such as Structured Query Language (SQL) and Query-By-Example (QBE) are syntactic variants of FO. Immermann in Halpern et al. (2001) proposed several reasons why FO has turned out to be so successful as a query language, from which three main reasons are:

- FO has syntactic variants that are used to build practical languages (SQL, QDB),
- FO-based query languages can be efficiently implemented using relational algebra (Codd 1982). The algebra turns out to yield a crucial advantage when large amounts of data are concerned,
- in principle, FO queries can be evaluated in constant time, independent of the database size, when parallelism is available.

We close this section by a non-exhaustive enumeration of logics applied in different fields of computer science. Some of them are new formalisms for computer science goals, but many are "borrowed" from mathematical or philosophical logic, however fragments of those formalisms with better computational properties are sometimes constructed. Our list of logics is as follows: automated deduction, program logics, type theory, domain theory logic, equational logic, term rewriting, formal semantics of programming languages, linear logic, logic programming, constraint logic programming, inductive logic programming, adductive logic programming, epistemic and temporal logics, logics for spatial reasoning, nonmonotonic logics, description logics, logics of hybrid systems.

3. McCarthy's programs with commonsense

In 1950s researchers believed that programs will be able to solve problems using human-like intelligence (Reiter 1980).

In 1959 J. McCarthy proposed the *advice taker* – a program for solving problems by drawing conclusions by reasoning (McCarthy 1959). For this purpose, the *advice taker* should use formal language when manipulating the statements.

With such a design, McCarthy expected the program to be improvable by describing its symbolic environment to it and what is wanted from it.

These statements will not require any knowledge of the program or any a priori knowledge of the advice taker. The user of such a program can assume that he will be given an answer based on logical consequences of anything it was told before.

McCarthy likens this property with human's *commonsense*, as "*it automatically deduces for itself a sufficiently wide class of immediate consequences of anything it is told and what it already knows*".

The *advice taker* has a method for representing expressions (terms) in computer. Certain of these expressions may be regarded as declarative sentences in a certain logical system, others are the names of entities of various kinds (objects, individuals, functions and programs). Also, *immediate deduction routine* is part of the system. The program is intended to operate cyclically, drawing conclusion by the *immediate deduction routine* and obeying conclusions of the imperative form (routine *deduce and obey* may be obeyed too). Although McCarthy did not choose the particular formal system, he suggested that it will have a single rule of inference which will combine substitution for variables with modus ponens. The purpose of this was to avoid choking the machine with special cases of general propositions already deduced.

An airport scenario. To describe *advice taker's* abilities, McCarthy used simple real world problem. In this example, a man is sitting behind the desk in his home and there is a car in his garage. The man lives in a county with an airport. If the man decides to go to the airport the *advice taker* will, collecting right premises and using deduction to draw conclusions, advice the man to walk from the desk to the car and drive the car to the airport.

Bar-Hillel's criticism. The article McCarthy (1959) is published together with a discussion after the presentation of the advice taker. A critique expressed by Bar-Hillel is interesting and significant. His main objection is that deduction is not a proper reasoning mode for formalizing commonsense. The development of the attempts to formalize commonsense reasoning in next decades confirmed Bar-Hillel's objections. His critique considers incredible the machine to conclude proposed goal, i.e., "Walk from your desk to your car!", by sound deduction. Bar-Hillel argues that *such conclusion could not be drawn from the premise in any serious sense,* as there are varieties of other options, e.g., to call a taxi, to cancel a flight, etc. (McCarthy 1959).

To sum up, McCarthy's vision about implementing commonsense reasoning using deduction failed. An appropriate semantic description of commonsense reasoning in terms of deduction is not possible. Moreover, also computational point of view was against deduction. Undecidable and computationally demanding logical deduction has been considered as an inappropriate tool for computational modelling of commonsense reasoning.

4. Nonmonotonic logics

In seventies came McCarthy with a new vision, the vision about understanding commonsense reasoning as jumping to conclusions. A goal was to construct some new logics, logics which, unlike careful and stepwise proceeding deduction, jump to conclusions.

The new slogan about jumping to conclusions aimed to characterize quick and hypothetical reasoning with incomplete knowledge. This type of logics has been dubbed non-monotonic,[2] mainly within artificial intelligence community, or defeasible, mainly within philosophical logic community.

The year 1980 can be considered as a milestone in the development of nonmonotonic logics. A series of seminal papers has been published in a special volume of the *Artificial Intelligence Journal* (we emphasize McCarthy 1980, McDermott – Doyle 1980, Reiter 1977).

Researchers interested in formalizations of commonsense reasoning gained in two decades between McCarthy (1959) and McCarthy (1980) a rich experience. It became clear that attempts to implement commonsense reasoning using deduction from a knowledge base are not feasible. On the other hand, and more importantly, some positive insights have been reached.

Investigations of methods and patterns useful for a formalization of reasoning about actions and for representing relevant knowledge ran into some crucial problems. A cost-saving representation of a domain should not contain axioms about properties, which do not change as a result of a given action. The content of the frame problem is how to propose such

[2] If $A \subset B$ are sets of formulae then all consequences of A are not necessarily consequences of B. Commonsense reasoning is nonmonotonic in a sense that consequences drawn from (incomplete) knowledge may be rejected after an addition of new information.

rational representation. A solution is based on an idea of *inertia* – properties of objects are *usually* fixed, *except* some, which are affected by an action. We assume by *default* that actions do not change properties of object. If they are changed, it must be expressed explicitly. This is an intuitive basic idea, but formal solutions need to overcome some subtle problems. Representation of an action usually contains a representation of prerequisites and effects of the action.[3] An attempt to represent both aspects leads to problems closely connected to the frame problem. The content of the qualification problem is how to qualify normal circumstances (prerequisites) of an action. Usually we assume that no exceptional conditions occur, when an action is intended. Similarly, a representation of actions should be focused only on direct effects of an action under normal conditions. A need to abstract from possible indirect effects of actions is the content of the ramification problem.

A necessity to introduce and study ways how to reason about usual, normal, default properties or situations with possible exceptions was recognized also in other areas of reasoning research.

The development and experience briefly described above led to a conviction that another type of logic is needed. The earliest nonmonotonic formalisms, considered as classical, are circumscription (McCarthy 1980), default logic (Reiter 1980) and autoepistemic logic Moore (1985).

Some stimuli for studying nonmonotonic reasoning came also from the database and logic programming fields. Closed world reasoning (Reiter 1977) is appropriate for databases: if a sentence, say A, is not recorded in a database D, it is natural to assume that $\neg A$ holds in D. The same idea is applied in logic programming – according to the principle of negation as failure (Apt – Bol 1994) – *not* A[4] is assumed, if a proof of A fails in a finite time.

A noteworthy amount of research was devoted in eighties by logic programming community to deep semantic investigations of the negation as failure principle (Apt – Bol 1994). A new paradigm of logic programming, answer set programming (ASP) (Marek – Truszczynski 1998, Niemelä 1999), based on stable model and answer set semantics (Gelfond – Lifschitz 1988, 1990) resulted from this research period. Moreover, characterizations

[3] Logical representation of actions is in this volume presented in Čertický (2013).

[4] Notice that another symbol is used for the negation as failure.

of classical nonmonotonic formalisms in terms of semantics of logic programs has been provided.[5]

A characterization of some crucial features of nonmonotonic reasoning is given below on the example of default logic.

5. Default logic

We are aiming to show three important features of (some) formalizations of nonmonotonic reasoning on the example of default logic:

- the role of nonmonotonic assumptions in reasoning,
- use of fixpoint constructions,
- computational intractability, its causes and its relation to representational compactness.

5.1. Default rules and nonmonotonic assumptions

A default rule is of the form[6]

$$\frac{\alpha : \beta}{\gamma}$$

where α (called premise), β (justification) and γ (conclusion) are formulae[7] of a logical language L, we will say that the default rule is over L. Usually, a first order language is considered as a general option. Ordinary rules of deductive logic contain premises and conclusions. However, there are fundamental differences. Default rules are not inference rules, they are not structural (they do not specify operations on expressions of some form) and they are domain-dependent.

Let consider an example of a default rule (over a propositional language), a formal treatment of the presumption of innocence.

---

[5] More about logic programing as a computational and conceptual tool for implementing and studying nonmonotonic reasoning can be found in this volume in Šiška – Šimko (2013).

[6] We will use also notation $\alpha : \beta / \gamma$.

[7] Other, more general definitions of a default rule are possible.

$$\frac{\textit{accused} : \neg \textit{proven\_guilty}}{\textit{innocent}}$$

The intuitive idea behind a default rule is that its conclusion (somebody is innocent, in our example) is acceptable, if the prerequisite (she/he is accused) is true and there is no argument against the justification (it is not known that he/she is proven guilty). A default rule represents actually a proposition with a reference to a global context.[8]

As a consequence, a default rule is not applicable locally as usual inference rules. Default rules are context-dependent expressions, their truth/acceptability depends on a given global context. In some contexts justifications are supported, in other they are falsified. Hence, justifications may serve as nonmonotonic assumptions – if a new information is given, previously acceptable assumption may be rejected. Marek and Truszczynski characterize nonmonotonic reasoning as context-dependent reasoning (Marek – Truszczynski 1993).

An important point of view on nonmonotonic reasoning, emphasizing a role of nonmonotonic assumptions, was presented in Bondarenko et al. (1997), where nonmonotonic reasoning is interpreted as a deduction from nonmonotonic assumptions.

5.2. Extension, fixpoint constructions

First we define a default theory as a pair $T = (E, D)$, where E is a set of ordinary formulae of a logical language L and D is a set of default rules over L. Suppose (without loss of generality) that L is a first order language. We denote by Cn_{FOL} the consequence operator of the first-order logic.

The meaning of a default theory is characterized by its extension.

Let X be a set of formulae of L and $\Gamma(X)$ be a minimal (w.r.t. inclusion) set of formulae s.t.

- $E \subseteq \Gamma(X)$,
- $Cn_{FOL}(\Gamma(X)) = \Gamma(X)$,
- if $(\alpha : \beta / \gamma) \in D$, $\alpha \in \Gamma(X)$, $\neg\beta \notin X$, then $\gamma \in \Gamma(X)$.[9]

[8] Nothing in the relevant context, within the accessible knowledge, supports that she/he is proven guilty.

[9] If this condition is satisfied, it is said that $\alpha : \beta / \gamma$ is applicable w.r.t. X.

An extension is a set of formulae $\mathcal{E}$ s.t. $\mathcal{E} = \Gamma(\mathcal{E})$, i.e. an extension is a fixpoint of the operator Γ.

Different default theories may have no extension, exactly one extension or more extensions. The case of more extensions leads to two kinds of (default) reasoning – skeptical and credulous. If a formula holds in each extension of a default theory T, then it is a skeptical consequence of T. If it holds in an extension, then it is its credulous consequence.
A fixpoint as defined above enables to specify a set of formulae $\mathcal{E}$, which is

- supported, i.e., each formula of $\mathcal{E}$ is either a member of E or a consequence of an applicable default rule or a deductive consequence of them,
- saturated, i.e., all members of E, all consequences of applicable default rules and deductive consequences of both classes of formulae are contained in $\mathcal{E}$,
- coherent, i.e., no default rule with a justification denied by $\mathcal{E}$ is applicable.

The fixpoint construction used in the definition of extension is a typical conceptual tool used in formalizations of nonmonotonic reasoning and it enables to specify coherent, supported and saturated sets of formulae.

5.3. Default logic – computational aspects

Some negative results concerning computational aspects of default logic are presented in this subsection. Generally, nonmonotonic formalisms did not satisfy original expectations about efficient computations, about "jumping to conclusions".

First order language. Let denote the premise of a default rule d as $pre(d)$ and its justification as $just(d)$. Then $d \in D$ is applicable w.r.t. an extension $\mathcal{E}$ iff $\mathcal{E} \models pre(d)$ and $\mathcal{E} \not\models \neg just(d)$.

The relation $\models$ is not recursive, it is recursively enumerable, hence $\not\models$ is not recursively enumerable and the decision problem whether $\mathcal{E}$ is an extension of a default theory is not semi decidable.

Propositional language. The following basic types of problems are usually considered in investigations of computational complexity of propositional default theories.

Given is a propositional formula ϕ and a default theory $T = (E, D)$.

Q1: does ϕ belong to each extension of T?

Q2: does ϕ belong to some extension of T?

Q3: is $\mathcal{E}$ an extension of T?

The problems are more hard than NP-complete, they are complete on the second level of polynomial hierarchy (Gottlob 1995).

At first glance it seems that the computational complexity of default reasoning is caused by a need to test entailment or satisfiability.

More about causes of intractability. Kautz and Selman (1991) posed a question whether the computational complexity of problems connected to default logic is really caused (only) by satisfiability (entailment) testing.

They studied only simple disjunction-free propositional languages with default rules of the form $a_1 \wedge \ldots \wedge a_k : b_1 \wedge \ldots \wedge b_m \wedge c_1 \wedge \ldots \wedge c_n / b_1 \wedge \ldots \wedge b_m$ and a hierarchy of even more simple disjunction-free default theories up to default theories with default rules of the form $p : q/q$ and $p : \neg q/\neg q$. Membership in a set is tested in those theories instead of testing satisfiability / consequence.

Results reached by Kautz and Selman were pessimistic also for those simple default theories. Kautz and Selman analyze causes of those results as follows:

- there are conflicts between default rules and incoherent cycles,
- there is an exponential number of extensions in the worst case.

The causes mentioned above may be illustrated by this example: Let E be empty and $D = \{: \neg b_1/a_1, : \neg a_1/b_1, \ldots, : \neg b_n/a_n, : \neg a_n/b_n\}$.

Conflicts and incoherent cycles can be detected for each pair of rules $: \neg b_i/a_i$ and $: \neg a_i/b_i$. Each extension contains exactly n atoms – one from each pair a_i, b_i, it means there are 2^n extensions of this default theory.

Is intractability of nonmonotonic reasoning a real drawback? The title of this paragraph is identical with the title of a paper by Cadoli, Donini and Schaerf (1996). They studied some nonmonotonic formalisms, in which inference is not computable in polynomial time. They investigated compilations of a given nonmonotonic formalism into another formalism. The idea of this investigation is as follows. A formalism and a reasoning problem, which is not solvable in polynomial time, is given. The goal is to

study, whether it is possible to translate (compile) the problem into other formalism, where the problem is solvable in polynomial time. The time, required for the compilation, may be arbitrary. The result of their investigations was that for many formalisms and problems the compilation requires exponentially bigger space.

Their interpretation of this class of results is that nonmonotonic formalisms enable extremely compact representation of propositional knowledge. Thus, computational intractability is a price for having a compact representation.

6. Updates

Nonmonotonic reasoning is closely related to belief change. If a new formula is believed (inserted into a knowledge base) then the nonmonotony of the corresponding consequence relation may lead to a rejection of some beliefs (formulae from the set of consequences of the original knowledge base). A process started by insertion, acceptance of some new propositions and by subsequent solving some conflicts, by a rejection of some propositions is called an update.

Updates of nonmonotonic knowledge bases (sets of formulae with a nonmonotonic consequence operator) are analyzed in this section. A role of nonmonotonic assumptions determine some special features of updates of nonmonotonic knowledge bases, as we will see below.

An original result is presented in this section. Our goal is to continue with focusing on default theories and to investigate updates of default theories. According to our best knowledge there is no paper devoted to this topic. On the other hand, updates of logic programs were studied intensively in the last fifteen years. We will generalize our semantics of logic program updates (Šefránek 2011) to updates of assumption based frameworks (ABF) (Bondarenko et al. 1997). Updates of a specialization of ABF for logic programs were studied in (Šefránek 2012), but here updates of abstract, general ABF are studied directly (according to our best knowledge, no research was devoted to updates of ABF). Updates of ABF may be transferred to updates of default theories – an ABF view on default theories was presented in (Bondarenko et al. 1997).

We begin with a description of a problem with logic program updates and a proposal of its solution using a principle of inertia of the current

state. The problem occurs within all semantics, based on a causal rejection principle. We omit here technicalities connected to approaches based on the causal rejection principle (see, e.g., Leite 2003, Homola 2004). Only a discussion of an example will serve as a starting point for a sketchy presentation of the main idea of our approach.

Consider the following example, where P is an original program and U is an updating program.

Example 1 $P = \{a \leftarrow; b \leftarrow a\}$

$U = \{\neg a \leftarrow not\ b\}$

We get two distinguished models (dynamic stable models) of this update according to an arbitrary semantics, based on the causal rejection principle: $S_1 = \{a, b\}$, $S_2 = \{\neg a\}$.

The first one complies with natural intuitions – S_1 is a natural description of the world, given by P. U is vacuously true in S_1, because not b is false in S_1, hence nothing is supported by U w.r.t. S_1, i.e. w.r.t. the current state of affairs.

On the other hand, S_2 is counterintuitive. The rule $\neg a \leftarrow not\ b$ does not change the description of the world given by P. The consequence $\neg a$ could be accepted only if there is no evidence that b holds (if there is no such evidence, the assumption not b is justified). However, b holds according to P.

The acceptance of S_2 is the result of a too free selection of interpretations checked by a fixpoint condition used in definitions of dynamic semantics of logic program updates. Without going into details: let an operator Φ defined on interpretations is given. If Φ is applied to S_2, the fact $a \leftarrow$ of P is rejected because of preference of new information of U and because not b is satisfied in S_2. Hence, the residue of $P \cup U$ is $\{b \leftarrow a, \neg a \leftarrow not\ b\}$ and for an appropriately defined operator Φ holds $\Phi(S_2) = S_2$.

According to our view, a selection of a candidate for a semantic characterization of updates should be somehow restricted. The main idea is that we do not accept unjustified assumptions. In the example above S_1 was supported by an empty set of assumptions and S_2 by the set $\{not\ b\}$, which is a superset of the empty set. A kind of Occam's razor is used: a minimization of non-monotonic assumptions. We prefer the empty set of assumptions over the set $\{not\ b\}$ in the presented example. The empty set of assumptions is sufficient for a semantic characterization of the current state, hence we respect an inertia of the current state. A principle of inertia of the current state was introduced in (Šefránek 2011; 2012). The principle serves as a solution of a problem of irrelevant updates (Šefránek 2006). The example presented above represents an irrelevant update.

We will start with a slightly modified version of ABF in the next subsection. Then dynamic assumption based framework, a specification of updates of ABF, is introduced. Finally, the construction is applied to updates of default theories.

6.1. Assumption based framework

Assumption based framework is constructed over a deductive system. A deductive system is a pair (L, R), where

- L is a language, i.e., a countable set of sentences (formulae),
- R is a set of rules of the form

$$\frac{\alpha_1, \ldots, \alpha_n}{\gamma}$$

where $\alpha_i, \gamma \in L$, $n \geq 0$. We suppose that for each formula $f \in L$ is defined its negation $\neg f$, however no special properties of a negation operator are supposed on this abstract level.

An assumption based framework is a quadruple (L, R, A, c), where $A \subset L$ is a set of assumptions and c is a mapping defined for each assumption s.t. $c(a) \in L$ is called contrary of a. Negation $\neg$ and c may be different.

We illustrate the notion of ABF on the example of default logic. Suppose a default theory (E, D) and a deductive system (L_0, R_0) of first order logic. A default rule

$$\frac{\alpha : \beta}{\gamma}$$

can be rewritten as

$$\frac{\alpha, M\beta}{\gamma}$$

hence, default rules have the same form as rules of a deductive system. Thus, we define $L = L_0 \cup \{M\beta \mid \beta \in L_0\}$, $R = R_0 \cup D$, the set of assumptions A contains all $M\beta$ s.t. $M\beta$ occurs in a default rule. Finally, $c(M\beta) = \neg\beta$.

It was shown in Bondarenko et al. (1997) that default theories, logic programs and also other nonmonotonic formalisms may be understood and presented as assumption based frameworks and nonmonotonic reasoning

within those formalisms as a deduction from nonmonotonic assumptions. Similarly, nonmonotonic assumptions play a key role in our approach to updates of nonmonotonic theories (nonmonotonic knowledge bases).

We are returning back to a general ABF. A proof of a formula f of L is a sequence $\langle f_1, \ldots, f_n \rangle$, where f_1 is a set S of assumptions, each f_j where $j > 1$, is a consequence of a rule $r \in R$, premises of r are previous members of the sequence and $f_n = f$,[10] notation $S \vdash f$.

It is clear that in an ABF corresponding to a default theory holds $S \vdash M\beta$ iff $M\beta \in S$. In general, if for each set of assumptions S in an ABF holds that $S = \{a \in A \mid S \vdash a\}$, S is called closed. We consider only closed ABF in this paper.

A set of assumptions $S_1 \subseteq A$ undercuts in an ABF a set of assumptions $S_2 \subseteq A$, if for some $a \in S_2$ holds that $S_1 \vdash c(a)$.

S_1 rebuts in an ABF S_2, if $S_1 \vdash f$ and $S_2 \vdash \neg f$. If $\neg\neg f \equiv f$ in L then rebutting relation is symmetric.

S_1 attacks S_2 in an ABF iff it undercuts or rebuts it in the ABF.[11] We can introduce attacks of an assumption against a set of assumptions, of a set of assumptions against an assumption and between assumptions by identifying a singleton set of assumptions with an assumption.

Argumentation semantics, originally defined in Dung (1995) were applied to ABF in Bondarenko et al. (1997).

We define some argumentation semantics in the following paragraphs.

A set of assumptions is conflict-free in an ABF iff it does not attack itself in the ABF.

A conflict-free set of assumptions S is admissible in an ABF iff for each $a \in A$ holds that if a attacks S in the ABF then S attacks a in the ABF.

A conflict-free set of assumptions S is stable in an ABF iff it attacks each $a \in A \setminus S$ in the ABF.

6.2. Dynamic assumption based framework

A dynamic ABF is a pair of ABFs $\mathcal{A}_1 = (L, R_1, A_1, c)$ and $\mathcal{A}_2 = (L, R_2, A_2, c)$. $\mathcal{A}_1$ is an original ABF and $\mathcal{A}_2$ is an updating ABF. The pair represents an update operation. The updating ABF is more preferred than

[10] Notice that a rule may have 0 premises. As a consequence, we consider proofs from a set S of assumptions instead of a proof from $S \cup T$, where $T \subseteq L$ is a theory.

[11] Our attack relation differs from the attack relation of Bondarenko et al. (1997). Use of undercutting in this paper is forced by distinguishing $\neg$ and c.

the original ABF. Proof and the relation $\vdash$ are defined as before, the only difference is that rules from $R_1 \cup R_2$ are taken into account. We will sometimes use notation $\vdash_{R_1 \cup R_2}$ or a similar one, with an index denoting the set of used rules.

Conflicts solving and the inertia of the current state. We recognize two kinds of conflicts, corresponding to undercutting and rebutting.

Definition 1. *Let Δ be a set of assumptions. It is said that Δ contains a conflict w.r.t. a set of rules R iff*

- $\Delta \vdash_R f$, $\Delta \vdash_R \neg f$ *for some* $f \in L$,
- $a \in \Delta$ *and* $\Delta \vdash_R c(a)$.

Definition 2. *A* solution *of a conflict C contained in a set of assumptions Δ w.r.t. $R_1 \cup R_2$ is a minimal set of rules R s.t. Δ does not contain C w.r.t. $(R_1 \cup R_2) \setminus R$.*

A solution of all *conflicts contained in Δ w.r.t. $R_1 \cup R_2$ is a minimal set of rules R s.t. Δ contain no conflict w.r.t. $(R_1 \cup R_2) \setminus R$.*

Notice that also $\emptyset$ may be a solution of all conflicts.

Consequence 1. *Assume that an ABF $(L, R_1 \cup R_2, A_1 \cup A_2, c)$ is given. Let R be a solution of all conflicts in Δ w.r.t. $R_1 \cup R_2$.*

Then Δ is *conflict free in he ABF $(L, R, A_1 \cup A_2, c)$.*

Thus, conflict-free sets of assumptions are results of conflicts solving. However, simple solving of conflicts is not sufficient for a specification of updating. First, a preference of more recent updating ABF over the original ABF should be respected. We will define preferential conflict solving. Second, irrelevant updates, i.e., updates, which are not applicable to the current state of affairs should be ignored.

Let proceed to the preferential conflict solving. The preference of R_2 means that, if it is possible to solve a conflict by rejecting a rule from R_1 or by rejecting a rule from R_2, we prefer the first option. More formally:

Consider two rules r_1, $r_2 \in R_1 \cup R_2$. We say that r_2 is more preferred than r_1 iff $r_2 \in R_2$ and $r_1 \in R_1$ (notation: $r_1 \prec r_2$).

Definition 3. *Suppose that Q_1, $Q_2 \subseteq R_1 \cup R_2$.*

If $\exists r_1 \in Q_1 \setminus Q_2$ $\exists r_2 \in Q_2 \setminus Q_1$ $r_2 \prec r_1$ and $\neg(\exists r_3 \in Q_2 \setminus Q_1$ $\exists r_4 \in Q_1 \setminus Q_2$ $r_4 \prec r_3)$ then Q_1 is more preferred than Q_2.

Definition 4. *Let Q be a solution of all conflicts in Δ w.r.t. $R_1 \cup R_2$.*

Q is called a preferred solution if there is no set of rules $R \subseteq R_1 \cup R_2$ s.t. R is more preferred than Q and R is also a solution of all conflicts in Δ w.r.t. $R_1 \cup R_2$.

Conflict solving, based on a preference relation, is not sufficient for an intuitively satisfactory specification of update, remind Example 1. Our analysis of that example aimed to show that it is not reasonable to solve conflicts for an arbitrary set of assumptions. More specifically, it is not reasonable to accept more assumptions than is necessary w.r.t a given description of the current state.

Definition 5. *Let $(\mathcal{A}_1 = (L, R_1, A_1, c),\ \mathcal{A}_2 = (L, R_2, A_2, c))$ be a DABF and $\Delta \subset \Omega \subseteq A_1 \cup A_2$ be sets of assumptions.*

It is said that Δ defeats Ω, if Δ attacks Ω in $(L, R_1 \cup R_2, A_1 \cup A_2, c)$.

Suppose that Q_1, Q_2 are preferred solutions of all conflicts in Δ and Ω, respectively, w.r.t. $R_1 \cup R_2$.

Suppose that Ω is defeated by Δ and both are stable sets of assumptions w.r.t. subsets Q_1, Q_2 of $R_1 \cup R_2$, respectively.

Then the set $\{f \mid \Omega \vdash_{Q_1} f\}$ is an irrelevant update of $\mathcal{A}_1$ by $\mathcal{A}_2$.

Let Q_2 be a preferred solution of all conflicts in Δ w.r.t. $R_1 \cup R_2$. Then the set $S = \{f \mid \Delta \vdash_{Q_2} f\}$ is a stable update of $\mathcal{A}_1$ by $\mathcal{A}_2$ iff it is not an irrelevant update of $\mathcal{A}_1$ by $\mathcal{A}_2$.

Suppose that Δ is a stable set of assumptions w.r.t. ABF (L, R_1, A_1, c) and also w.r.t. $(L, R_1 \cup R_2, A_1 \cup A_2, c)$. If the set $\{f \mid \Omega \vdash_Q f\}$ is for each Ω s.t. $\Delta \subset \Omega$ and for each $Q \subset R_1 \cup R_2$ an irrelevant update of $\mathcal{A}_1$ by $\mathcal{A}_2$ then we can speak about an inertia of the current state (specified by Δ).

Finally, we notice that the presented approach to updates of ABF can be applied to updates of default theories. It was shown in Bondarenko et al. (1997) that default theories are a special case of ABF.

7. Conclusions

A view on some important features of logical aspects of knowledge representation is presented in the first part of this paper. Formalizations of reasoning with incomplete, dynamic and conflicting knowledge represent a challenge for logicians. We mention here the role of nonmonotonic assumptions in that kind of reasoning.

Our original result is presented in the second part of the paper. This part is connected to the first part by emphasizing nonmonotonic assumptions while specifying update of nonmonotonic knowledge bases. Key idea of our approach is a minimization of nonmonotonic assumptions and, consequently, a principle of inertia of the current state.

As regards open problems and future research, we will devote our attention to more detailed investigations of properties of the presented approach and for extension of that approach to admissible and complete extensions. We suppose that the principle of inertia of the current state is applicable to complete extensions, but not to admissible extensions.

References

Apt, K. R. – Bol, R. (1994): Logic programming and negation: A survey. *Journal of Logic Programming* 19, 9-71.

Bondarenko, A. – Dung, Phan Minh – Kowalski, R. A. – Toni, F. (1997): An abstract, argumentation-theoretic approach to default reasoning. *Artif. Intell.* 93, 63-101.

Cadoli, M. – Donini, F. M. – Schaerf, M. (1996): Is intractability of nonmonotonic reasoning a real drawback? *Artif. Intell.* 88, Nos. 1-2, 215-251.

Codd, E. F. (1970): A relational model of data for large shared data banks. *Communications of the ACM* 13, No. 6, 377-387.

Codd, E. F. (1982): Relational database: A practical foundation for productivity. *Commun. ACM* 25, No. 2, 109-117.

Čertický, M. (2013): Action models and their induction. *Organon F* 20, supplementary issue 2, 206-215.

Dung, Phan Minh (1995): On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.* 77, No. 2, 321-358.

Gelfond, M. – Lifschitz, V. (1988): The stable model semantics for logic programming. In: *ICLP/SLP*, 1070-1080.

Gelfond, M. – Lifschitz, V. (1990): Logic programs with classical negation. In: *ICLP*, 579-597.

Gottlob, G. (1995): The complexity of default reasoning under the stationary fixed point semantics. *Inf. Comput.* 121, No. 1, 81-92.

Halpern, J. Y. – Harper, R. – Immerman, N. – Kolaitis, P. G. – Vardi, M. Y. – Vianu, V. (2001): On the Unusual Effectiveness of Logic in Computer Science. *The Bulletin of Symbolic Logic* 7, No. 2, 213.

Homola, M. (2004): Dynamic logic programming: Various semantics are equal on acyclic programs. In: Leite, J. A. – Torroni, P. (eds.): *CLIMA, Computational Logic*

in Multi-Agent Systems, 5th International Workshop, CLIMA V, Lisbon, Portugal, September 29-30, 2004, 78-95.

KAUTZ, H. A. – SELMAN, B. (1991): Hard problems for simple default logics. *Artif. Intell.* 49, Nos. 1-3, 243-279.

LEITE, J. (2003): *Evolving Knowledge Bases: Specification and Semantics.* IOS Press.

MAKINSON, D. (2002): Ways of doing logic: what was different about agm 1985? *Journal of Logic and Computation* 12.

MANNA, Z. – WALDINGER, R. (1985): *The Logical Basis for Computer Programming, Vol. 1: Deductive Reasoning.* Addison-Wesley Professional.

MAREK, V. W. – TRUSZCZYNSKI, M. (1993): *Nonmonotonic Logic – Context-Dependent Reasoning.* Artificial intelligence. Springer.

MAREK, V. W. – TRUSZCZYNSKI, M. (1998): Stable models and an alternative logic programming paradigm. *CoRR,* cs.L0/9809032.

MCCARTHY, J. (1959): Programs with common sense. In: *Proceedings of the Symposium on the Mechanization of Thought Processes.* Teddington, England: National Physiology Lab, 1-15.

MCCARTHY, J. (1980): Circumscription, a form of non monotonic reasoning. *Artificial Intelligence* 13, 27-39.

MCCARTHY, J. – HAYES, P. J. (1969): Some philosophical problems from the standpoint of artificial intelligence. In: *Machine Intelligence*. Edinburgh University Press, 463-502.

MCDERMOTT, D. – DOYLE, J. (1980): Non-monotonic logic I. *Artificial Intelligence* 13, 41-72.

MOORE, R. C. (1985): Semantical considerations on nonmonotonic logic. *Artif. Intell.* 25, No. 1, 75-94.

TURING, A. M. (1950): Computing machinery and intelligence. *Mind* LIX, No. 236, 433-460.

NIEMELÄ, I. (1999): Logic programs with stable model semantics as a constraint programming paradigm. *Ann. Math. Artif. Intell.* 25, Nos. 3-4, 241-273.

REITER, R. (1977): On closed world data bases. In: *Logic and Data Bases*, 55-76.

REITER, R. (1980): A logic for default reasoning. *Artificial Intelligence* 13, 81-132.

ŠEFRÁNEK, J. (2006): Irrelevant updates and nonmonotonic assumptions. In: *Proceedings of the 10th European conference on Logics in Artificial Intelligence,* JELIA'06. Berlin – Heidelberg: Springer-Verlag, 426-438.

ŠEFRÁNEK, J. (2011): Static and dynamic semantics. Preliminary report. In: *MI- CAI 2011, Special Session, Mexican International Conference on Artificial Intelligence*.

ŠEFRÁNEK, J. (2012): Updates of argumentation frameworks. In: Rosati, R. – Woltran, S. (eds.): *Proceedings of the 14th International Workshop on Non-Monotonic Reasoning.* Rome.

ŠIMKO, A. – ŠIŠKA, J. (2013): Logic programming and interactive applications. *Organon F* 20, supplementary issue 2, 187-205.