Artificial Intelligence 259 (2018) 167–185

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Safe inductions and their applications in knowledge representation ☆

Bart Bogaerts [a,*], Joost Vennekens [a,b], Marc Denecker [a]

[a] Department of Computer Science, KU Leuven, 3001 Heverlee, Belgium
[b] Department of Computer Science, KU Leuven, Campus De Nayer, 2860 Sint-Katelijne-Waver, Belgium

ARTICLE INFO

Article history:
Received 1 December 2017
Received in revised form 13 March 2018
Accepted 25 March 2018
Available online 28 March 2018

Keywords:
Approximation fixpoint theory
Lattice operator
Inductive definitions
Induction process
Construction
Well-founded semantics
Groundedness
Logic programming
Autoepistemic logic
Abstract argumentation

ABSTRACT

In many knowledge representation formalisms, a constructive semantics is defined based on sequential applications of rules or of a semantic operator. These constructions often share the property that rule applications must be delayed until it is *safe* to do so: until it is known that the condition that triggers the rule will continue to hold. This intuition occurs for instance in the well-founded semantics of logic programs and in autoepistemic logic. In this paper, we formally define the safety criterion algebraically. We study properties of so-called *safe inductions* and apply our theory to logic programming and autoepistemic logic. For the latter, we show that safe inductions manage to capture the intended meaning of a class of theories on which all classical constructive semantics fail.

1. Introduction

In many fields of computational logic, natural forms of *induction* show up. Such an induction can be seen as a sequence of semantic structures obtained by iterative applications of rules or a semantic operator. For instance, in logic programming, it is natural to think of sequences of interpretations where at each stage a number of rules whose bodies are satisfied are triggered (i.e., their head is added to the current interpretation). For positive logic programs, all such sequences converge to the minimal model. For non-positive programs, this strategy may yield meaningless results. For instance, for the program

$$\mathcal{P} = \left\{ \begin{array}{l} a \\ b \leftarrow \neg a \end{array} \right\},$$

one such sequence is

☆ A short version of this paper was published in the proceedings of the IJCAI'17 conference [6]. This paper extends the previous work with more theoretical results, examples, proofs of all propositions and applications of the work to argumentation frameworks.

* Corresponding author.
E-mail addresses: bart.bogaerts@cs.kuleuven.be (B. Bogaerts), joost.vennekens@cs.kuleuven.be (J. Vennekens), marc.denecker@cs.kuleuven.be (M. Denecker).

https://doi.org/10.1016/j.artint.2018.03.008
0004-3702/

$$\mathcal{N}_1 = \emptyset, \{b\}, \{b, a\},$$

the limit of which is not even a supported model of the logic program. On the other hand, the sequence

$$\mathcal{N}_2 = \emptyset, \{a\}$$

is another such sequence that *does* end in the intended model of $\mathcal{P}$, namely its perfect model. Intuitively, what is wrong with $\mathcal{N}_1$ is that the rule $b \leftarrow \neg a$ is applied too soon, before the value of a is established. For stratified programs, like $\mathcal{P}$, this problem has been resolved, e.g., by Apt et al. [2]. For the general case, the well-founded semantics [33] offers a solution that uses three-valued interpretations instead of two-valued interpretations.

In recent work, the notions of *natural* and *safe* inductions for inductive definitions were introduced [15,16]. It was argued that this kind of process forms the essence of our understanding of inductive definitions.

In this paper, we lift those ideas of safe and natural inductions to a more general setting: we provide a principled study of such inductions in the context of *approximation fixpoint theory (AFT)* (Denecker, Marek and Truszczyński (DMT) [10]), an algebraic theory that provides a unifying framework of semantics of nonmonotonic logics. We show convergence of safe inductions in this general setting and study the relationship between (algebraic) safe inductions and various fixpoints defined in approximation fixpoint theory.

By presenting our theory in AFT, our results are broadly applicable. DMT [10] originally developed AFT to unify semantics of logic programs [32], autoepistemic logic [26] and default logic [28]. Later, it was also used to define semantics of extensions of logic programs, such as HEX logic programs [1] and an integration of logic programs with description logics [23]. Strass [30] showed that many semantics for Dung's argumentation frameworks (AFs) [17] and abstract dialectical frameworks (ADFs) [7] can be obtained by direct application of AFT. Bogaerts and Cruz-Filipe [3] showed that AFT has applications in database theory, for defining semantics of active integrity constraints [19].

The theory we present in this paper induces for each of the above logics notions of (safe) inductions and a *safe semantics*. Our complexity results are obtained for general operators and hence can also be transferred to various logics of interest. Throughout the paper, we give examples from logic programming.

In Section 7, we apply our theory to autoepistemic logic. There we show that safe inductions induce a constructive semantics that captures the intended semantics of a class of theories for which classical constructive semantics fail. This failure was recently exposed and solved using a notion of *set-inductions* which is based on sets of lattice elements instead of intervals (which are standard in AFT) [5]. We show that safe inductions provide an alternative solution to this problem. Our solution is more direct: in contrast to set-inductions or well-founded inductions [14], safe inductions do not require any form of approximation; they are sequences in the original lattice. For logic programming, this means that they are sequences of interpretations such that some atoms are derived in each step. For AEL, this means that they are sequences of possible world structures such that additional knowledge is derived in each step.

In Section 8, we apply our theory to Dung's argumentation frameworks [17], where we show the surprising result that two different operators that exist for a given argumentation framework have *the same* safely defined point. Furthermore, this point corresponds to an existing semantics: it is the so-called *grounded extension*.

The rest of this paper is structured as follows. In Section 2, we give preliminaries regarding lattices and operators. In Section 3, we define (safe) inductions and provide some basic results. We continue by studying complexity of some inference problems related to safe inductions in Section 4. In Section 5, we recall the basics of AFT; we use this in Section 6 to study how (safe) inductions relate to various fixpoints studied in AFT. Afterwards, in Sections 7 and 8, we apply our general theory to autoepistemic logic and argumentation frameworks respectively. We conclude in Section 9.

2. Preliminaries: lattices and operators

A *partially ordered set (poset)* $\langle L, \leq \rangle$ is a set L equipped with a partial order $\leq$, i.e., a reflexive, antisymmetric, transitive relation. We write $x < y$ for $x \leq y \wedge x \neq y$. If S is a subset of L, then x is an *upper bound*, respectively a *lower bound* of S if for every $s \in S$, it holds that $s \leq x$, respectively $x \leq s$. An element x is a *least upper bound*, respectively *greatest lower bound* of S if it is an upper bound that is smaller than every other upper bound, respectively a lower bound that is greater than every other lower bound. If S has a least upper bound, respectively a greatest lower bound, we denote it $lub(S)$, respectively $glb(S)$. As is custom, we sometimes call a greatest lower bound a *meet*, and a least upper bound a *join* and use the related notations $\bigwedge S = glb(S)$, $x \wedge y = glb(\{x, y\})$, $\bigvee S = lub(S)$ and $x \vee y = lub(\{x, y\})$. We call $\langle L, \leq \rangle$ a *complete lattice* if every subset S of L has a least upper bound and a greatest lower bound. A complete lattice has a least element $\bot$ and a greatest element $\top$.

An operator $O : L \to L$ is *monotone* if $x \leq y$ implies that $O(x) \leq O(y)$ and *anti-monotone* if $x \leq y$ implies that $O(y) \leq O(x)$. An element $x \in L$ is a *prefixpoint*, a *fixpoint*, a *postfixpoint* of O if $O(x) \leq x$, respectively $O(x) = x$, $x \leq O(x)$. Every monotone operator O in a complete lattice has a least fixpoint [31], denoted $\mathrm{lfp}(O)$, which is also O's least prefixpoint and the limit of any terminal *monotone induction* of O, defined below.

Definition 2.1. A *monotone induction* of a lattice operator $O : L \to L$ is an increasing sequence (for some ordinal β) $(x_i)_{i \leq \beta}$ of elements $x_i \in L$ satisfying