An introduction to approximation fixpoint theory

Tutorial at KR 2024

Jesse Heyninck[1] and Hannes Strass[2]

[1]Open Universiteit, the Netherlands
[2]Technische Universität Dresden, Germany

May 3, 2024

Abstract

[A two-sentence description for the brochure:] Approximation fixpoint theory is a general framework that gives constructive techniques to approximate fixpoints of potentially non-monotonic operators. It has been shown to capture and unify the main semantics of a wide variety of formalisms in knowledge representation, such as logic programming, formal argumentation and default logic, which we use as illustrations of the general theory.

1 Tutorial Description

Semantics of various formalisms in fields of artificial intelligence, such as knowledge representation, can be described by fixpoints of corresponding operators. For example, in many logics (e.g. propositional or modal logics) theories are fixpoints of the underlying consequence operator [**?**]. Likewise, in logic programming, default logic, or formal argumentation, all the major semantics can be formulated as different types of fixpoints of the same operator [**?**, **?**]. Such operators are often non-monotonic, and so their fixpoints cannot be guaranteed to exist, or be defined constructively. In order to deal with this elusive nature of the fixpoints, Denecker, Marek and Truszczyński [**?**] introduced a method for *approximating* each value z of the underlying operator by a pair of elements (x,y). These elements represent lower and upper bounds on z, and so a corresponding *approximation operator* for the original, non-monotonic operator, is constructed. A natural requirement is that the approximating operator is precision-monotonic, intuitively meaning that more precise inputs of the operator give rise to more precise outputs. This requirement ensures (by Tarski and Knaster's Fixpoint Theorem) that the approximating operator has least (and greatest) fixpoints that consequently approximate the fixpoints of the approximated operator (whenever such fixpoints exist). Many existing formalisms in knowledge representation were shown to make implicit use of approximation fixpoint theory (e.g. logic programming [**?**], default logic [**?**], autoepistemic logic [**?**], abstract argumentation frameworks [**?**], hybrid MKNF [**?**], SHACL [**?**], and active integrity constraints [**?**]), and the semantics of several new formalisms were obtained by a straightforward application of approximation fixpoint theory (e.g. extensions of logic programs [**?**, **?**, **?**], abstract dialectical frameworks (ADFs, [**?**]) as well as their *weighted* counterparts [**?**], access control policies [**?**] and second-order logic extended with non-monotone inductive definitions [**?**]).

The goal of this tutorial is to introduce approximation theory to the community of AI researchers. The emphasis of this tutorial will be on the motivation and applications of approximation fixpoint theory. This emphasis will be achieved by a detailed study of some of the formalisms to which the application of approximation fixpoint theory has received, historically, the most attention, namely logic programming and abstract dialectical frameworks. In more detail, we will use this formalisms to highlight the challenges that occur when developing a semantics for a KR-language, and how approximation fixpoint theory allows to surmount this challenges. In doing so, we will organically pass along the main AFT-based semantics (e.g. the stable, well-founded and Kripke-Kleene semantics) and constructive techniques for obtaining approximation operators (e.g. the trivial and ultimate approximation operator).

2 Potential target audience, prerequisite knowledge, and learning goals

The tutorial is open for any researcher in knowledge representation, regardless of whether they have a background in knowledge representation or algebra. All necessary mathematical background will be introduced in the tutorial, as well as formalisms from knowledge representation that serve as guiding example, such as logic programming.

The goal of the tutorial is to introduce the basic ideas underlying approximation fixpoint theory as a tool for the definition and study of semantics of knowledge representation formalisms, as well as the language-independent study of concepts in knowledge representation. The learning outcomes of this tutorial are the following:

- Understand the role of operators and fixpoints in knowledge representation.
- Understand the concept of an approximation in knowledge representation, and its connection with three- and four-valued logics.
- Understand the idea of an approximation of a potentially non-monotonic operator, and how this allows to approximate fixpoints of the original operator.
- Understand how the stable approximator is constructed, and how this allows to define the well-founded fixpoint.
- Realize the benefit of the algebraic approach to knowledge representation underlying approximation fixpoint theory, and how this allows to give a language-independent account of important concepts occurring in different sub-fields of knowledge representation.

3 Motivation

Approximation fixpoint theory is arguably on of the main unifying approaches in knowledge representation. Due to its language-independent nature, it allows for a top-down methodology that forms a viable and fruitful alternative to the bottom-up, language-dependent methodology that is still the standard in a lot of work on knowledge representation. Furthermore, it encompasses a set of techniques that allow to straightforwardly define semantics with proven quality guarantees for many KR-formalisms. The goal of this tutorial is to educate non-experts on this established but specialized AI methodology, as we believe the perspective offered by approximation fixpoint theory has many more potential applications within and beyond knowledge representation.

4 Tutorial Outline

The proposed length of the tutorial is half a day (two 1:45 hour slots). The tutorial is conceived as follows:

- Motivation and history: a guided tour through the history of logic programming (75 minutes)
  - Syntax of normal logic programs.
  - First semantics: completion, minimal and perfect models.
  - Well-founded and stable semantics, immediate consequence operator.
  - Four-valued interpretations and approximation operators.
  - Stable approximations.
  - Extending logic programming syntax: aggregates.
- An abstract view: from logic programming to (general) AFT (45 minutes)
  - Lattices, complete lattices and operators.
  - Fixpoints of operators and (Knaster and) Tarski's fixpoint theorem.
  - Bilattices and approximation operators, Kripke-Kleene semantics.
  - Stable operators and well-founded semantics.
  - Ultimate approximations.
- A second application: (weighted) abstract dialectical frameworks (45 minutes)
  - Boolean ADFs
  - Weighted ADFs
- More general insights (45 minutes)
  - Modularity
  - Groundedness
  - Complexity