A Category-Theoretic Perspective on Approximation Fixpoint Theory*

Samuele Pollaci

Vrije Universiteit Brussel
Brussels, Belgium

Katholieke Universiteit Leuven
Leuven, Belgium

`Samuele.Pollaci@vub.be`

Approximation Fixpoint Theory (AFT) was founded in the early 2000s by Denecker, Marek, and Truszczyński as an abstract algebraic framework to study the semantics of non-monotonic logics. Since its early successes, the potential of AFT as a unifying semantic framework has become widely recognised, and the interest in AFT has gradually increased, with applications now ranging from foundations of database theory to abstract argumentation. The non-monotonic constructive processes that occur in many more areas of computer science, together with their associated semantic structures, can be successfully studied using AFT, which greatly simplifies their characterizations. The goal of my research is to take a step towards the lifting of AFT into a more general framework for constructive knowledge.

1   Introduction and Related Work

Approximation Fixpoint Theory (AFT) was founded in the 2000s by Denecker, Marek, and Truszczyński [14] as an extension of Tarski's fixpoint theory to study the semantics of non-monotonic logics, like default logic (DL), autoepistemic logic (AEL) and logic programming (LP). In recent years, interest in AFT has gradually increased, with applications now ranging from foundations of database theory to abstract argumentation. Motivated by the success of AFT in this wide range of applications, this project aims at laying the foundation for an extention of AFT from a useful tool in the area non-monotonic logics into a general algebraic theory of constructive knowledge.

In the 1980s and 90s, the area of non-monotonic reasoning (NMR) saw fierce debates about formal semantics. In the subareas of DL, AEL and LP, researchers sought to formalize common-sense intuitions about knowledge of introspective agents. The main contribution of AFT was to demonstrate that, by moving to an algebraic setting, the common principles behind the concepts in these languages can be isolated and studied in a general way. This breakthrough allowed results that were achieved in the context of one of these languages to be easily transferred to another [27, 32].

The core ideas of AFT are relatively simple: we are interested in fixpoints of an operator O on a given lattice $\langle L, \leq \rangle$. For monotonic operators, Tarski's theory guarantees the existence of a least fixpoint, which is of interest in many applications. For non-monotonic operators, the existence of fixpoints is not guaranteed; and even if fixpoints exist, it is not clear which would be "good" fixpoints. AFT generalizes Tarki's theory for monotonic operators by making use of a so-called *approximating operator*; this is an operator $A : L^2 \to L^2$, that operates on L^2, and that is monotonic with respect to the precision order $\leq_p$ (defined by $(x, y) \leq_p (u, v)$ if $x \leq u$ and $v \leq y$)). The intuition is that elements of L^2 approximate

---

elements of L: $(x, y) \in L^2$ approximates z if $x \leq z \leq y$, i.e. when $x \leq y$, the tuple (x, y) can be thought of as an interval in L. Given such an approximator, AFT defines several types of fixpoints (supported fixpoints, a Kripke-Kleene fixpoint, stable fixpoints, a well-founded fixpoint) of interest. In several fields of non-monotonic reasoning, it is relatively straightforward to define an approximating operator and it turns out that the different types of fixpoints then correspond to existing semantics. In this way, AFT clarifies on the one hand how different semantics in a single domain relate, and on the other hand what the relation is between different (non-monotonic) logics.

Since its early successes, the potential of AFT as a unifying semantic framework has become widely recognised. It has been applied to a multi-agent extension of AEL [29], to Dung's abstract argumentation frameworks (AFs), and abstract dialectical frameworks (ADFs) [25]. It has also been applied to extensions of LP with aggregates [16], with HEX-atoms [1], with higher-order functions [11], and with description logics [22]. It served as the foundation of the causal logic FO(C) [8] and was used to characterize active integrity constraints [4]. In all these domains, the power of AFT allows a variety of complex semantic structures to be characterized by surprisingly simple approximating operators. In comparison to a direct definition of the semantic structures, the AFT approach therefore greatly reduces the risk of error, while significantly simplifying the mathematical study of these structures. Interest in these application areas has also driven the theoretical development of AFT in new directions [22, 11, 5, 7, 17].

The recent broad interest in AFT and the resulting research output provide a unique opportunity to put AFT on the map as a general algebraic theory of constructive knowledge. Throughout computer science, Tarski's fixpoint theory is used because it allows a potentially complex object (the least fixpoint of a monotone operator) to be constructed as the limit of a simple iteration process. In many applications, however, this simple monotone construction process does not suffice. In such cases, it is sometimes possible to devise other, derived operators that do not operate on the basic semantic space but on a more complex space. The core of AFT lies in the study of the general principles that underlie such approximations. Such non-monotone construction processes occur in many more areas of computer science, including formal verification, functional programming, and database theory. The goal of this project is to develop a single unified framework to study them, thereby bringing two important benefits. First, a general theory provides confidence in the correct characterization of the processes and resulting semantics. Non-monotone construction processes can be highly complex and developing a new formalization of such a process from scratch is a difficult, time-consuming and error-prone task. This advantage of AFT has already been convincingly demonstrated for logic programs with aggregates, where direct definitions of the desired semantic structures are highly complex, while the use of AFT requires nothing more than the definition of a (three-valued) truth evaluation function for the aggregates. Second, once the constructive process has been correctly characterized, AFT offers a powerful set of algebraic tools to analyze the process and its limit. For instance, a general toolset for the study of modularity properties was developed, generalizing several known concepts from AEL, DL and LP [28, 30, 31, 32]. As a second example, Truszczyński [28] developed tools for the study of strong equivalence, which immediately transfer to other fields, where such notions are now studied independently [23]. AFT greatly simplifies the characterization and subsequent study of constructive processes and their associated semantic structures. By lifting AFT into a more general framework for constructive knowledge, we bring these benefits to a wide range of application areas in computer science.

2 Scientific Research Goals

My PhD research activity fits into the framework of the AFTACK project (project G0B2221N) supported by Fonds Wetenschappelijk Onderzoek – Vlaanderen. The global goal of this project is to take the next big leap forward for AFT by lifting the results obtained for various specific application domains into a general framework for constructive knowledge. To achieve this, three major types of advancements are needed.

A. General approximation spaces: A construction process consists of a series of approximative objects. In AFT, these are elements (x, y) of the bilattice L^2, which correspond to intervals $[x, y] = \{z \in L \mid x \leq z \leq y\}$ in the original lattice. In various scenarios, intervals are not refined enough and a more general approximation space is needed.

B. General processes: AFT was built for processes typically found in non monotonic reasoning. When moving beyond this scope, the basic concepts behind constructive processes remain the same, but certain key differences nevertheless arise. For instance, while processes in AFT typically construct relations over a given set, domain theory [24] considers processes that simultaneously construct the relations and the set over which they are defined.

C. Analysis of processes: The success of AFT is for a large part due to the rich algebraic toolkit it offers to analyze the defined processes and semantics. In parallel with extending the range of spaces and processes, this toolkit needs to be extended as well.

These three goals are materialized in five concrete research objectives. In my research, I am going to focus primarily on three of them:

1. Approximation Spaces (A). Develop a generalization of AFT where the approximations can be complex mathematical structures, instead of simple intervals of lattice elements. In several applications, the limitation to intervals was recognized as a key limiting factor [11, 22, 6, 3]. The generalisation should be general enough to cover these domains.

2. Recursively defined domains and higher-order functions (A,B). Develop extensions of AFT and domain theory suitable to define recursive higher-order functions and predicates. Preliminary experiments have shown that AFT is unsuitable for recursive definitions of higher-order functions [12]. For monotonic recursively defined functions, a solution is provided in domain theory. The objective is to extend domain theory [24] with the fixpoint notions of AFT to handle non-monotonically defined recursive functions and predicates.

3. Explanations for AFT (C). In many domains, it is not only important to reach the right conclusions, but also to explain why they are correct. For instance, in causality, this is the question of actual causation [20]. In the context of constructive knowledge, this question can be posed as "Why does a property hold in the constructed object?". Achieving explainability is especially important in the light of the EU General Data Protection Regulation, article 22 of which requires that all AI with an impact on human lives needs to be accountable. Therefore, we want to obtain a principled approach towards explanations in AFT, which will immediately be applicable to all logics captured by AFT.

The impact of this project is spread over many different domains. In fact, within each new application domain, valuable lessons can be learnt from AFT, as has been witnessed before, for instance in the context of default logic, weighted argumentation, and active integrity constraints. The largest short-term impact is probably found in Objective 2 (Recursively defined domains and higher-order functions). If

successful, this project will on the one hand be a bridge between functional programming and non-monotonic reasoning, and on the other hand, will provide the semantic foundations for a new class of function definitions by lifting the restriction that function definitions need to be monotonic in the definedness order. Finally, explainability is an important topic in many of the application domains of AFT right now. By studying this once, algebraically, we lay the foundations for work on explanations in each of these research areas.

3   Research Methodology

In the following, I break down the three objective presented above into smaller work packages (WP). This subdivision follows the one proposed for the AFTACK project. In particular, WP2 will be the main core of my research activity. At the same time, I will collaborate with other members of the research group on (parts of) the other work packages. More details are provided in Section 4.

3.1   WP1 Approximation Spaces

To develop a notion of an approximation space of a given lattice $\langle L, \leq \rangle$, we need a mathematical structure equipped with a truth order $\leq$ and a precision order $\leq_p$, on which we define a generalization of an approximating operator. It is important that the space is equipped with enough structure to allow the construction of key concepts of AFT, such as the stable operator, whose fixpoints determine the partial stable model semantics and the well-founded semantics [14]. We first tackle this problem for two specific structures. We expect the lessons learned there to prove valuable when finally developing approximation spaces in WP1.3.

WP1.1. We first consider the use of the powerset of L as an approximations space. Based on a preliminary analysis, we conjecture that it is impossible to define a suitable truth order on the set of all subsets S of L. Therefore, we focus on sets "without holes", formally, sets S such that whenever $x, z \in S$ and $x < y < z$, also $y \in S$. The most important challenge is defining a stable operator in this context, and then studying the resulting semantics.

WP1.2. Weighted Abstract Dialectical Frameworks (wADFs) [10] were originally defined in the context of an arbitrary set with a precision order (a complete partial order) but no truth order. In this work package, we research (i) how to add a truth order to wADFs, and (ii) which restrictions on the respective orders we need in order to generalize stable and well-founded semantics to wADFs.

WP1.3. Now, we are concerned with formally defining the notion of an approximation space. This should be a mathematical structure $\langle L, \leq, \leq_p \rangle$ of approximations of a lattice $\langle L, \leq$ such that (i) there is a $\leq$-preserving injection from L into L; (ii) we can generalize the definition of an approximating operator $A: L^2 \to L^2$ on the bilattice L^2 to approximating operators $A: L \to L$ on the approximation space L; (iii) we can generalize the key concepts of AFT (such as the stable operator) to L; and (iv) the generalizations cover at least the approximations from WP1.1, and WP1.2, and possibly other extensions of AFT [11, 22].

WP1.4. We investigate what it means for an operator on an approximation space to be stratified or modular. We also investigate which types of fixpoints of the generalized AFT behave well under such stratification [30].

3.2 WP2 Recursively defined domains and higher-order functions

Defining semantics for definitions of higher-order objects is challenging for typed languages and even more so for untyped ones. Semantics for typed and untyped monotone higher-order function definitions are developed in denotational semantics for lambda calculus and functional programming languages [26]; one approach is domain theory [24]. Domain theory was the first approach to provide semantics for the untyped lambda calculus and allows for more complex semantic spaces than AFT, with rich classes of approximate objects, suitable for inductive and coinductive constructions of higher-order objects and infinite data structures. It simultaneously constructs approximations of these complex objects and of the functions operating on them, resulting in a space of self-applicable functions including a fixpoint operator through which recursive definitions are given meaning. In another sense, domain theory is more limited than AFT: it only considers continuous (hence, monotone) operators and definitions. Here, we combine the strengths of both approaches.

WP2.1. We study approximation spaces in a category-theoretic setting. In particular, we investigate which classes of approximation spaces form a Cartesian-closed category. One reason we are interested in this is that a Cartesian-closed category provides a foundation for applying AFT to higher-order definitions, as it provides a systematic construction of approximation domains for higher-order concepts from those of base concepts. We undertake this investigation for different notions of approximations.

WP2.2. Here, we exploit the framework of WP2.1 to develop languages and semantics of non-monotone definitions of higher-order sets and functions. The framework supports this (i) by providing construction of approximation domains for higher-order concepts; (ii) by imposing abstract conditions on language constructs that ensure that they are well-behaved (being morphisms in the category); (iii) by being generic in the underlying order, allowing to combine different orders (e.g., simultaneous definition of sets and functions).The range of notions of approximation space in the framework should offer an enriched spectrum with different trade-offs between precision, mathematical complexity, and computational complexity.

WP2.3. In AFT, the approximator is strictly higher in the set-theoretical hierarchy than the objects on which it operates. A powerful property of domain theory is that this is not the case: the objects it constructs can be applied as functions on all constructed objects, including itself. This property is key to defining semantics of untyped lambda calculus. Dana Scott's construction of such a domain is one of the great achievements in the theory of programming languages, but depends on the $\leq_d$-continuity (and hence, monotonicity) of operators of definitions, a condition we seek to relinquish. In this final work package, we examine whether it is possible to partially lift the limitation of $\leq_d$-monotonicity of domain theory and, at the same time, lift AFT's limitation that prevents its use for self-application.

3.3 WP3 Explanations for AFT

To bring explainability to AFT, we start from the theory of justifications [13]. Like AFT, this is a unifying theory that can characterize semantics of various formalisms. However, it is more specific, in the sense that it does not consider operators on an arbitrary lattice, but is focused on the special case of a powerset lattice. This allows justification theory to build detailed explanations of why each element of a set $X \in L$ belongs to X. Nevertheless, there are strong correspondences between AFT and justification theory, including similar notions of duality. The exact relationship remains largely unexplored.

WP3.1. Instead of starting from an operator, justification theory starts from a "justification frame", from which an operator can be derived. These frames can be nested, resulting in "nested justification systems". In this work package, we research the relationship between the concepts of non-nested justification theory and those of AFT. In particular, we verify the conjecture that for all main "branch evaluations" (a concept from justification theory), the semantics induced by justification theory coincides with the equally named fixpoint(s) in AFT. In other words, this WP provides an embedding of non-nested justification theory into AFT. While this result is valuable in itself, the more interesting question is how to generalize AFT using notions from justification theory, which is what we do next.

WP3.2. In WP3.1, we consider only the operators on powerset lattices induced by justification theory. In this work package, we research (i) how to generalize the notion of justification to work for an approximator over an arbitrary complete lattice, or more general, approximation space, (ii) how to automatically obtain justifications from operators (or rather, from approximators), and (iii) how these obtained justifications translate back to justification theory. While the main focus of this work package is on extending approximation fixpoint theory, and bringing justifications to its various fields of application, the results achieved here also have a significant impact on the theory of justifications itself. Indeed, one consequence of these results would be that we now also obtain justifications for so-called ultimate semantics [15].

WP3.3. As mentioned, justification theory allows justification frames to be nested. This results in an elegant way of capturing the semantics of, for instance, nested least and greatest fixpoint definitions. Now, we research how to achieve the same in AFT. In other words, we develop a suitable notion of nested approximations, where a single "step" of a high-level approximation may include an entire fixpoint construction of a lower-level approximation. In addition, it should be possible to choose, for each of the different levels independently which are the fixpoints of interest.

WP3.4. In the context of justification theory, we recently defined notions of duality, one of which is induced by inverting the truth order. In general, we expect that such notions also show up in AFT and that many properties have interesting dual variants, for instance so-called symmetric approximators would be self-dual. In this work package we perform a complete study of duality in AFT in general: we identify dual properties and study which new results we obtain by exploiting them.

The research goals of the AFTACK project in which my research activity fits are both fundamental and ambitious. It is therefore probable that we will not be able to meet every single one of them. However, this need not be a problem, since each of the individual steps towards the desired goals is in itself innovative and will lead to publications in high-impact conferences and journals. Moreover, the WPs are structured in such a way that there is little interdependence between them. The most central WP is that on approximation spaces (WP1). The research line on explanations (WP3) has a minor dependency on WP1, in the sense that it needs to be studied in the general context of approximation spaces in order to have maximum impact. However, even in the unlikely worst case that the WP on approximation spaces should fail completely, studying the topics in the context of the bilattice of standard AFT still provides novel and interesting results. The research line on recursive functions has the strongest dependency on approximation spaces, in the sense that we expect that the bilattice will not suffice for tackling this topic. Nevertheless, even if we cannot construct approximation spaces that are general enough to meet all of our stated goals for that WP, we can still develop a less ambitious extension of the bilattice that will be enough to allow the work on recursive functions to move forward.

4 Current Status and Future Plan

So far, I have been mainly working on WP2 for AFT for higher-order definitions, specifically WP2.1. and WP2.2.. In particular, a paper on the stable semantics for higher-order logic programming has been accepted for ICLP2024, and another paper on the generalization of approximation spaces for higher-order objects using the tools of Category Theory is under review for LPNMR2024. The latter paper is also contributing to WP1.1 and WP1.3 as it proposes a novel, general notion of possible approximation spaces. Continuing this line of work, I am currently studying the relation between our new research outputs and previous approaches, like [11], both from a theoretic perspective and from a more computational one, regarding the complexity in finding the models of a logic program. As next step, I plan to work towards the completion of both WP1 and WP2, in particular by tackling WP1.2. on wADFs, WP1.4., and WP2.3.. The study of AFT for wADFs may lead the way for the application of AFT to bound-founded ASP [2], assumption-based argumentation [9], weighted argumentation [19], probabilistic argumentation [18], and social argumentation [21].

References

[1] Christian Antic, Thomas Eiter & Michael Fink (2013): *Hex Semantics via Approximation Fixpoint Theory*. In: *Logic Programming and Nonmonotonic Reasoning, 12th International Conference, LPNMR 2013, Corunna, Spain, September 15-19, 2013. Proceedings*, pp. 102–115, doi:10.1007/978-3-642-40564-8_11.

[2] Rehan Abdul Aziz, Geoffrey Chu & Peter J. Stuckey (2013): *Stable model semantics for founded bounds*. *Theory Pract. Log. Program.* 13(4-5), pp. 517–532, doi:10.1017/S147106841300032X.

[3] Bart Bogaerts (2019): *Weighted Abstract Dialectical Frameworks through the Lens of Approximation Fixpoint Theory*. In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 2686–2693, doi:10.1609/aaai.v33i01.33012686.

[4] Bart Bogaerts & Luís Cruz-Filipe (2018): *Fixpoint semantics for active integrity constraints*. *Artif. Intell.* 255, pp. 43–70, doi:10.1016/j.artint.2017.11.003.

[5] Bart Bogaerts, Joost Vennekens & Marc Denecker (2015): *Grounded fixpoints and their applications in knowledge representation*. *Artif. Intell.* 224, pp. 51–71, doi:10.1016/j.artint.2015.03.006.

[6] Bart Bogaerts, Joost Vennekens & Marc Denecker (2016): *On Well-Founded Set-Inductions and Locally Monotone Operators*. *ACM Trans. Comput. Log.* 17(4), p. 27, doi:10.1145/2963096.

[7] Bart Bogaerts, Joost Vennekens & Marc Denecker (2018): *Safe inductions and their applications in knowledge representation*. *Artif. Intell.* 259, pp. 167–185, doi:10.1016/j.artint.2018.03.008.

[8] Bart Bogaerts, Joost Vennekens, Marc Denecker & Jan Van den Bussche (2014): *FO(C): A Knowledge Representation Language of Causality*. *Theory Pract. Log. Program.* 14(4–5-Online-Supplement), pp. 60–69, doi:10.48550/arXiv.1405.1833.

[9] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski & Francesca Toni (1997): *An Abstract, Argumentation-Theoretic Approach to Default Reasoning*. *Artif. Intell.* 93, pp. 63–101, doi:10.1016/S0004-3702(97)00015-5.

[10] Gerhard Brewka, Hannes Strass, Johannes Peter Wallner & Stefan Woltran (2018): *Weighted Abstract Dialectical Frameworks*. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 1779–1786, doi:10.1609/aaai.v32i1.11545.

[11] Angelos Charalambidis, Panos Rondogiannis & Ioanna Symeonidou (2018): *Approximation Fixpoint Theory and the Well-Founded Semantics of Higher-Order Logic Programs*. Theory Pract. Log. Program. 18(3-4), pp. 421–437, doi:10.1017/S1471068418000108.

[12] Ingmar Dasseville, Matthias van der Hallen, Bart Bogaerts, Gerda Janssens & Marc Denecker (2016): *A Compositional Typed Higher-Order Logic with Definitions*. In: Technical Communications of the 32nd International Conference on Logic Programming, ICLP 2016 TCs, October 16-21, 2016, New York City, USA, pp. 14:1–14:13, doi:10.4230/OASIcs.ICLP.2016.14.

[13] Marc Denecker, Gerhard Brewka & Hannes Strass (2015): *A Formal Theory of Justifications*. In: Logic Programming and Nonmonotonic Reasoning - 13th International Conference, LPNMR 2015, Lexington, KY, USA, September 27-30, 2015. Proceedings, pp. 250–264, doi:10.1007/978-3-319-23264-5_22.

[14] Marc Denecker, Victor Marek & Mirosław Truszczyński (2000): *Approximations, Stable Operators, Well-Founded Fixpoints and Applications in Nonmonotonic Reasoning*. In Jack Minker, editor: Logic-Based Artificial Intelligence, The Springer International Series in Engineering and Computer Science 597, Springer US, pp. 127–144, doi:10.1007/978-1-4615-1567-8_6.

[15] Marc Denecker, Victor W. Marek & Miroslaw Truszczynski (2004): *Ultimate approximation and its application in nonmonotonic knowledge representation systems*. Inf. Comput. 192(1), pp. 84–121, doi:10.1016/j.ic.2004.02.004.

[16] Marc Denecker, Nikolay Pelov & Maurice Bruynooghe (2001): *Ultimate Well-Founded and Stable Semantics for Logic Programs with Aggregates*. In: Logic Programming, 17th International Conference, ICLP 2001, Paphos, Cyprus, November 26 - December 1, 2001, Proceedings, pp. 212–226, doi:10.1007/3-540-45635-X_22.

[17] Marc Denecker & Joost Vennekens (2007): *Well-Founded Semantics and the Algebraic Theory of Non-monotone Inductive Definitions*. In: Logic Programming and Nonmonotonic Reasoning, 9th International Conference, LPNMR 2007, Tempe, AZ, USA, May 15-17, 2007, Proceedings, pp. 84–96, doi:10.1007/978-3-540-72200-7_9.

[18] Phan Minh Dung & Phan Minh Thang (2010): *Towards (Probabilistic) Argumentation for Jury-based Dispute Resolution*. In: Computational Models of Argument: Proceedings of COMMA 2010, Desenzano del Garda, Italy, September 8-10, 2010, pp. 171–182, doi:10.3233/978-1-60750-619-5-171.

[19] Paul E. Dunne, Anthony Hunter, Peter McBurney, Simon Parsons & Michael J. Wooldridge (2011): *Weighted argument systems: Basic definitions, algorithms, and complexity results*. Artif. Intell. 175(2), pp. 457–486, doi:10.1016/j.artint.2010.09.005.

[20] Joseph Y. Halpern (2016): *Appropriate Causal Models and the stability of Causation*. Rev. Symb. Log. 9(1), pp. 76–102, doi:10.1017/S1755020315000246.

[21] João Leite & João G. Martins (2011): *Social Abstract Argumentation*. In: IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011, pp. 2287–2292. Available at http://ijcai.org/Proceedings/11/Papers/381.pdf.

[22] Fangfang Liu, Yi Bi, Md. Solimul Chowdhury, Jia-Huai You & Zhiyong Feng (2016): *Flexible Approximators for Approximating Fixpoint Theory*. In: Advances in Artificial Intelligence - 29th Canadian Conference on Artificial Intelligence, Canadian AI 2016, Victoria, BC, Canada, May 31 - June 3, 2016. Proceedings, pp. 224–236, doi:10.1007/978-3-319-34111-8_28.

[23] Emilia Oikarinen & Stefan Woltran (2011): *Characterizing strong equivalence for argumentation frameworks*. Artif. Intell. 175(14-15), pp. 1985–2009, doi:10.1016/j.artint.2011.06.003.

[24] Dana Scott (1972): *Continuous lattices*. In E. Lawvere, editor: Toposes, Algebraic Geometry and Logic, Lecture Notes in Mathematics 274, Springer Verlag, pp. 97–136, doi:10.1007/BFb0073967.

[25] Hannes Strass (2013): *Approximating operators and semantics for abstract dialectical frameworks*. Artif. Intell. 205, pp. 39–70, doi:10.1016/J.ARTINT.2013.09.004.

[26] Robert D. Tennent (1991): *Semantics of programming languages*. Prentice Hall International Series in Computer Science, Prentice Hall.

[27] Miroslaw Truszczynski (2006): *Strong and uniform equivalence of nonmonotonic theories - an algebraic approach*. Ann. Math. Artif. Intell. 48(3-4), pp. 245–265, doi:10.1007/S10472-007-9049-2.

[28] Miroslaw Truszczynski (2006): *Strong and uniform equivalence of nonmonotonic theories - an algebraic approach*. Ann. Math. Artif. Intell. 48(3-4), pp. 245–265, doi:10.1007/s10472-007-9049-2.

[29] Pieter Van Hertum, Marcos Cramer, Bart Bogaerts & Marc Denecker (2016): *Distributed Autoepistemic Logic and its Application to Access Control*. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016, pp. 1286–1292. Available at http://www.ijcai.org/Abstract/16/186.

[30] Joost Vennekens, David Gilis & Marc Denecker (2006): *Splitting an operator: Algebraic modularity results for logics with fixpoint semantics*. ACM Trans. Comput. Log. 7(4), pp. 765–797, doi:10.1145/1183278.1183284.

[31] Joost Vennekens, Johan Wittocx, Maarten Mariën & Marc Denecker (2007): *Predicate Introduction for Logics with a Fixpoint Semantics. Part I: Logic Programming*. Fundam. Informaticae 79(1-2), pp. 187–208, doi:10.3233/FUN-2007-791-209.

[32] Joost Vennekens, Johan Wittocx, Maarten Mariën & Marc Denecker (2007): *Predicate Introduction for Logics with Fixpoint Semantics. Part II: Autoepistemic Logic*. Fundam. Informaticae 79(1-2), pp. 209–227, doi:10.3233/FUN-2007-791-209.