South American Journal of Logic
Vol. 1, n. 2, pp. 447–459, 2015
ISSN: 2446-6719

Bringing AGM to Computer Science

Renata Wasserman and Márcio Moretto Ribeiro

Abstract

In the 80's AGM theory for Belief Revision was mainly developed by philosophers and logicians, while at the same time, computer scientists were facing concrete problems as database update and model-based diagnoses.

Recently, with the advances in computational power, it became more plausible to have implementations of the real theory. In this paper we explore two sides of the relationship between AGM and Computer Science: (i) using AGM for CS, as for example in proposals for AGM revision of system specification and (ii) using CS for AGM, by means of real experimentation with the theory.

Keywords: belief revision, computer science, AGM, Alchourrón

1 Introduction

The seminal paper by Alchourrón, Gärdenfors and Makinson [AGM85] introduced what is now known as the AGM theory of belief revision. The theory was was developed in the 80's as a means of describing how a rational agent should modify its beliefs when confronted with new incoming information. The AGM theory of belief revision was developed by two philosophers, Peter Gärdenfors and David Makinson, and a lawyer(!), Carlos Alchourrón, that devoted his brilliant carrier to the study of the logic of normative systems [AB71]. AGM theory quickly became a sort of a golden standard against which one should evaluate any new ideas.

Parallel to the development of the AGM theory, Fagin et al. [FKUV86] described the closely related problem of updating a database – the "view updating problem". The authors describe two main problems of view update: its lack of unique solution and the difficulty of guarantying integrity constraints after an update.

The development of belief revision theory helped the understanding of such problems. Katsuno and Mendelzon [KM91], for example, argued that there is a subtle, but very important difference between what they called revising and updating a theory. In

the first case, the agent acquired new information about a static world that it must incorporate in its beliefs. In the second, the world itself has evolved and, hence, the agent should incorporate some information about the new state of affairs. Since each of these types of change has is own characteristics, before dealing with view update in a database, one must understand its nature.

Furthermore, the problem of the multiple solutions is dealt elegantly in belief revision framework. Instead of proposing an unique, inevitably ad hoc, solution for a revision operation, AGM theory states that it must satisfy certain rationality postulates i.e. the theory describes constraints that a solution must satisfy.

Another classical example of application of Belief Revision in Computer Science is its use in Artificial Intelligence. Following Newell [New82], knowledge should be understood functionally in terms of what it does instead of how it is represented. Levesque [Lev84] gives a step further arguing that an agent's knowledge should be a kind of abstract datatype accessible only through a fixed number of operations such as `TELL`, `FORGET` and `ASK`. Belief revision theory follows this functional view of knowledge. The beliefs of an agent are represented as a closed set of formulas and the theory describes, from a very abstract and functional point of view, how it should incorporate inconsistent knowledge (`TELL`) and how it should abandon undesirable pieces of knowledge (`FORGET`).

Besides providing important applications to belief revision theory, the converse, i.e. using computers to model belief revision, is also possible and promising. According to Bynum and Moor, computing should be seen as a tool for testing and evaluating philosophical ideas:

> "Computing and related concepts significantly enhance philosophy by providing a kind of intellectual clay that philosophers can mold and shape and study. Through computing, abstract ideas – which philosophers like to manipulate – can be instantiated and investigated. There is nothing wrong with good armchair reflection *without* the aid of a computer, (...) But armchair reflection has its limitations. As sophisticated as our imaginations and reasoning skills may be, there are practical limits to how much complexity we can process without some assistance. Armchair recursion doesn't recur very many times. But when ideas are modeled on a computer, consequences, especially consequences that emerge after complex processing, are revealed in a way that would be completely overlooked without such computer processing. Models and methods can be made more precise, tested and refined. These philosophical results of computing can be shared with others who also can submit them to their own scrutinity and development."[BM98]

Even the theory of belief base revision [Han99], which seems to be closer to application needs, lacks empirical studies which show how it performs in practice.

This work, as a tribute to the work of Alchourrón, aims to present a short review of current usages of belief revision theory in computer science. Furthermore, it presents a concrete example on how computer science can be used to provide insights to belief revision theory itself.

Section 2 presents the background, i.e. introduces the notation and briefly presents belief revision theory. Sections 3 to 4 presents two important fields of CS that can benefit from belief revision techniques: software specification and ontology evolution. The following section describes a concrete example of how Computer Science can be used to increase our understanding of Belief Revision through empirical testing. The last section concludes the paper.

2 Preliminaries

Consider a logic as a pair $\langle \mathscr{L}, Cn \rangle$ such that $\mathscr{L}$ is a language, i.e. the set of formulas, and $Cn : 2^{\mathscr{L}} \to 2^{\mathscr{L}}$ a consequence operator.

In AGM theory, a belief set as a set of sentences closed under logical consequence i.e. $K = Cn(K)$. Three basic operations are defined over a belief set: expansion ($+$), contraction ($-$) and revision ($*$). The first is the simple addition of a new sentence in K and is defined as $K + \alpha = Cn(K \cup \{\alpha\})$. As mentioned in the introduction, the other operations do not have a single solution, hence, they are defined through sets of rationality postulates. In a few words, *revision postulates* state that $K*\alpha$ is a consistent belief set that contains α and *contraction postulates* state that $K-\alpha$ is a belief set that does not contain α.

Besides presenting rationality postulates that the operations must satisfy, AGM theory presents ways to construct such operation. One example of construction for contraction, called partial meet contraction, consists of the intersection of some of the maximal subsets of K that do not imply α. The choice of the maximal subsets of K to be intersected is an extra-logical element of the theory. The construction is represented as:

$$K - \alpha = \bigcap \gamma(K \perp \alpha)$$

where $K \perp \alpha$ is the set of maximal subsets of K not implying α and γ is a function that selects some of these subsets.

The main result in AGM theory is the *representation theorem* which proves, on one hand, that every partial meet contraction satisfies the AGM postulates for contraction and, on the other hand, that every AGM contraction can be constructed as a partial meet contraction.

Usually, revision is defined via contraction using the so called *Levi identity*:

$$K * \alpha = (K - \neg\alpha) + \alpha$$

Again it is proved that the revision operation constructed via the Levi identity and partial meet contraction satisfies AGM postulates.

One important limitation of AGM theory, from a computational point of view, is that closed sets are typically infinite, even if in some cases they can be finitely represented. In a series of papers and a book [Han99], Hansson proposes to generalize AGM theory to arbitrary sets of formulas called belief bases. Besides being computationally easier to handle, belief bases are more expressive than belief sets. For instance, there is only one inconsistent belief set while there are many different inconsistent belief bases. This is due to the fact that, differently from belief sets, belief bases distinguish explicit from implicit beliefs. One may argue, however, that belief bases are much more syntax dependent.

As in the AGM theory, the same basic three operations are defined. Expansion in belief bases is simply $B + \alpha = B \cup \{\alpha\}$ and the other operations are also defined via different sets of postulates and constructions, linked by representation theorems. Besides partial meet contraction, an important construction proposed by Hansson is that of kernel contraction [Han99]:

$$B - \alpha = B \setminus \sigma(B \perp\!\!\!\perp \alpha)$$

where $B \perp\!\!\!\perp \alpha$ is the set of minimal subsets of B implying α and σ is a function that selects at least one element of each minimal subset.

In the next two sections, we present two areas of Computer Science where we have recently applied belief revision operations.

3 Revision of Software Specification

Complex systems are usually maintained by different people. In the process of formally specifying a system, a software engineer (or a team of engineers) may have to deal with requirements coming from different users, which may be jointly inconsistent. The engineer may also misunderstand the users' requirements, leading to the need of a revision of the specification. The need to update often arises when an user changes some of the requirements which were modelled.

A popular and efficient method for automated verification of systems specification, *model checking* consists in verifying whether a given system model satisfies the required properties. In this paper, we will focus on problems where the system is described as a Kripke model and the required properties to be checked are formalized in the temporal logic CTL [CE81].

In the last decades, several tools for automated model checking have been proposed

(CadenceSMV[1], NuSMV[2], SPIN[3]). A common characteristic of these systems is that they provide a counterexample in the cases where the model does not satisfy a given property. However, the systems do not help to solve the problem. Belief revision can help with this task: given a counterexample, we would like to look for minimal changes in the specification so that the property is satisfied.

What we would like to add to the existent model checkers is the ability to suggest changes. As an example, de Sousa and Wassermann have implemented a belief revision layer on top of NuSMV [dSW07]. The system provides, instead of a counterexample, suggestions for the minimal changes needed in the model so that it satisfies the given properties. The changes are given in the form of addition or removal of states or transitions.

The implementation of BR-NuSMV was useful in order to test our initial ideas. However, there are still many interesting theoretical issues to be solved. The AGM theory and even the results for belief base change depend on the compactness of the underlying logic. Since CTL is not compact, the relation between constructions and rationality postulates is still not clear. A first step was done in [GW10], where a construction was defined and shown to satisfy a set of postulates. The proposed construction was based on minimal changes in the Kripke model, as was the case in BR-NuSMV. While in classical AGM theory changes in models and changes in sets of formulas are interchangeable, Guerra and Wassermann have shown that this is not the case in CTL [dTOGW13]. Recently, van Zee et al. [vZDDvdT15] have provided representation theorems for changes in temporal formulas, but restricting them to a certain time limit.

4 Revision of Ontologies

In the last decade, we have seen great interest in Computer Science for ontologies, by means of formal languages used to represent knowledge about a certain domain. The most widely used languages, the members of the OWL family [PH12], are based on Description Logics [BCM+03]. There are already many tools available for building and reasoning with ontologies, such as graphical editors (e.g. Protégé[4]), reasoners (e.g. Pellet[5], HermiT[6]) and APIs (e.g. Jena[7], OWL API[8]). Ontologies may be used together with databases in two different ways: the database system is used to store the data in

[1] http://www.kenmcmil.com/smv.html
[2] http://nusmv.fbk.eu/
[3] http://spinroot.com/spin/whatispin.html
[4] http://protege.stanford.edu/
[5] http://clarkparsia.com/pellet/
[6] http://hermit-reasoner.com/
[7] http://jena.apache.org/
[8] http://owlapi.sourceforge.net/

the ontology or the ontology is used in order to access the data in a database in a more flexible way.

More recently, we see a growing interest in the dynamics of ontologies, with the need to revise, update or deal with inconsistencies. There are a number of workshops dedicated to the matter (IWOD, later EvoDyn, ARCOE, ...). Most Description Logics, however, are not compliant with AGM theory in the sense that no operation that satisfies the AGM postulates can be constructed [FPA05]. More precisely, Flouris et. al. defined a property of a logic called *decomposability*. In decomposable logic $\langle \mathscr{L}, Cn \rangle$ for every $K, A \subseteq \mathscr{L}$ such that $\emptyset \neq A \subset K$ there is $X \subset K$ such that $Cn(X \cup A) = Cn(K)$ called the *complement* of A w.r.t. K. The authors proved first that, in order to be compliant with AGM postulates for contraction, a logic must be decomposable. Second, most DLs are not decomposable and, hence, not AGM compliant.

Makinson in [Mak87] proposes an operation called *withdrawal* which satisfies every AGM postulate except recovery. It can be proved that any (Tarskian) logic is compliant with the withdrawal postulates. Hence, The impossibility to construct AGM contraction in DLs seems to be related to the *recovery* postulate. Years before this finding, Hansson already criticized this postulate in [Han91] and suggested to exchange it to some postulate that directly guaranties the minimality of change. Two postulates were presented for this purpose: *relevance* and *core-retainment*. Both, however, were proved equivalent to recovery in the presence of the other AGM postulates for contraction which led the author to conclude the following:

> "[Recovery postulate should be accepted as an] emerging property, rather than as a fundamental postulate, of belief set contraction"

In [RWFA13] the authors proved two important results. First it was proved the representation theorem relating partial meet contraction to the AGM postulates when recovery is exchanged by relevance. The novelty of the result is that, differently from the original result in [AGM85], it holds for every logic which is Tarskian and compact. Hence, a much wider class of logics is compliant when relevance is used instead of recovery. Second, it was proved that for many logics, DLs in particular, the equivalence between relevance and recovery doesn't hold in general. Hence, in such cases, recovery fails to capture the concept of minimality.

Furthermore, most DLs are not closed under negation, which poses interesting questions for applying belief revision operators. The lack of negation of axioms in most DLs prevents the usage of Levy identity to construct revision. In [RW09a, RW10] we have provided a direct construction (based on the set of maximal subsets of a belief set K that is consistent with a sentence α), a set of postulates and a representation theorem for revision of belief sets. This results are applicable for only a fraction of DLs, those

we called *distributive*[9]. This work was complemented in [RW14] where the following postulate was presented:

(strong inclusion) $K * \alpha \subseteq (K \cap K * \alpha) + \alpha$

We proved that strong inclusion together with certain postulates from belief base literature characterizes a construction very similar to partial meet revision. This result hold not only to distributive logics, but for every (Tarskian) compact and explosive[10] logic.

In [RW09b] we presented a collection of sets of postulates and the respective construction that it characterizes. The strategy used to avoid negation was to first insert the new sentence α and the remove the inconsistencies taking care not to remove α in the process. To illustrate the theory developed in [RW09b], we have implemented a Protégé Plugin [RW08] which revises or contracts the ontology being edited, giving the user choices between the several different operators. The ReviewAndContractProtegePlugin [11] uses the OWL API and the reasoner Pellet.

Two hot topics in ontology management are conflict resolution (given a single inconsistent ontology, either solve or isolate the inconsistencies) and merging (given two or more ontologies, join them in a consistent way), both widely studied in the belief revision literature [BDP95, KP11].

5 Empirical Testing

Even if we restrict ourselves to propositional logic, partial meet and kernel constructions make extensive use of theorem provers, or SAT solvers. And the SAT problem for propositional logic is long known to be intractable. However, with the advances in computational power in the last years, we can now adventure ourselves in implementations that were not possible before. The recent SAT competitions[12] show that it is already possible to deal with a good part of the SAT problems. This means that we can empirically evaluate the different operators proposed in the literature and see how they behave in practice.

Around the time of the development of the AGM theory, Reiter has proposed his theory for consistency based diagnoses [Rei87], together with an algorithm for computing minimal *hitting sets*. Although developed with a very particular purpose in mind, Reiter's algorithm was later shown to be easily adaptable to be used for computing

[9] In a distributive logic every $A, B, C \subseteq \mathscr{L}$ satisfies the following: $Cn(A \cup (Cn(B) \cap Cn(C))) = Cn(A \cup B) \cap Cn(A \cup C)$

[10] A logic is explosive if there is a finite set $A \subseteq \mathscr{L}$ such that $Cn(A) = \mathscr{L}$

[11] http://code.google.com/p/review-and-contract/wiki/ReviewAndContractProtegePlugin

[12] http://www.satcompetition.org/

belief contraction operations [Was00]. This algorithm has served as a basis for many of the implementations described in what follows.

5.1 BContractor

For this purpose, we have been working on a framework for general testing, the BContractor[13], which allows us to compare the behavior of the operators and try to get new insights that can be used to refine the theory.

BContractor [LRW12] was developed in order to provide a simple and uniform ground for testing contraction operators. The idea was to have an architecture providing the common constructs used in the literature (remainders, kernels, incision functions, selection functions), so that new operators based on them could be quickly coded. As a result, translation from formulas and pseudo-code to real working code is direct. An operation of partial meet contraction is encoded by:

```
public ISet<S> contract(ISet<S> base, S sentence) {
    return intersection(this.selection(this.remainder(base, sentence)));
    }
```

An important feature of BContractor is that it also abstracts away the particular logic being used. One has to provide the preferred theorem prover for the logic chosen and for some constructions, a method for calculating the negation of a formula. The framework has been tested with propositional logic, using miniSAT[14] and SAT4J[15], and with description logics, using HermiT[16].

The framework also provides facilities for generating random tests and for measuring different parameters such as time, size of the answer, and number of calls to the theorem prover.

BContractor was used as a basis for different implementations and tests, such as merging ontologies [CRW13], multiple contraction [RRW14] and ontology repair [CW15].

5.2 Case Study - Kernel vs. Partial Meet

The initial motivation for the development of BContractor was to compare the performance of kernel and partial meet contraction. Theoretically, the two constructions have the same computational complexity. Moreover, it had been widely accepted that

[13] http://code.google.com/p/bcontractor/wiki/GettingStarted
[14] http://minisat.se/
[15] http://www.sat4j.org/
[16] http://hermit-reasoner.com/

partial-meet only made sense as a theoretical construction, and that for implementations, kernel should be used, since it deals with minimal sets implying a formula instead of maximal sets not implying the formula. Some preliminary and very naïve implementations have shown that usually, it takes longer to compute the kernel sets than the remainders.

It is clear that dealing with smaller sets does not bring any advantages in the performance. Extensive tests performed with BContractor have shown that indeed, computing the remainder sets is on average faster [LRW12].

When trying to understand why this happens, we have measured a number of parameters during the executions. From the results, we see that computing remainders usually involves less calls to the SAT-solver. There are also more kernel sets than remainders.

One possible explanation of the difficulty in computing kernels may be related to the phenomenon known as *Phase Transition* [CKT91]. For the SAT problem, this has been studied and is now reasonably well-understood. Although the problem is NP-complete, some instances are harder than others [SLM92]. The harder instances lie where the ratio between the number of clauses and the number of variables is around 4.3 [GW94].

6 Conclusion

In this paper we made a case for the fact that computer science and belief revision can both gain from interacting with each other. Distant fields from database updating to ontology evolution and software specification can all get important insights from belief revision. This has been illustrated by concrete examples.

In software specification belief revision has been used to provide the ability to suggest model changes. In ontology evolution, revision and contraction are operations that occur naturally. AGM solution, though, is not directly applicable in this framework. In adapting the theory important results were achieved.

Furthermore, computational power, as argued by Bynum and Moor, can and should be used to experiment in philosophical issues. In this direction we presented a concrete example on how to use computers to empirically test an hypothesis, namely, that kernel and partial meet contraction, although equivalent from a computational complexity point of view, present very distinct efficiency results.

References

[AB71] Carlos Alchourrón and Eugenio Bulygin. *Normative Systems*. Springer-Verlag, 1971.

[AGM85] Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

[BCM+03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BDP95] Salem Benferhat, Didier Dubois, and Henri Prade. How to infer from inconsisent beliefs without revising? In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 1449–1457. Morgan Kaufmann, 1995.

[BM98] Terrell Ward Bynum and James H. Moor. How computers are changing Philosophy. In Terrell Ward Bynum and James H. Moor, editors, *The Digital Phoenix*, pages 1–14. Blackwell, 1998.

[CE81] E. M. Clark and E. A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In D. Kozen, editor, *Logics of Programs*, LNCS 131, pages 52–71. Springer-Verlag, 1981.

[CKT91] Peter Cheeseman, Bob Kanefsky, and William M. Taylor. Where the really hard problems are. In John Mylopoulos and Raymond Reiter, editors, *IJCAI*, pages 331–340. Morgan Kaufmann, 1991.

[CRW13] Raphael Cóbe, Fillipe Resina, and Renata Wassermann. Merging ontologies via kernel contraction. In Marcello Peixoto Bax, Mauricio Barcellos Almeida, and Renata Wassermann, editors, *Proceedings of the 6th Seminar on Ontology Research in Brazil, Belo Horizonte, Brazil, September 23, 2013*, volume 1041 of *CEUR Workshop Proceedings*, pages 94–105. CEUR-WS.org, 2013.

[CW15] Raphael Cóbe and Renata Wassermann. Ontology repair through partial meet contraction. In Richard Booth, Giovanni Casini, Szymon Klarman, Gilles Richard, and Ivan José Varzinczak, editors, *Proceedings of the International Workshop on Defeasible and Ampliative Reasoning, DARe 2015, co-located with the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015), Buenos Aires, Argentina, July 27, 2015.*, volume 1423 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[dSW07] Thiago Carvalho de Sousa and Renata Wassermann. Handling Inconsistencies in CTL Model-Checking using Belief Revision. In *Proceedings of the Brazilian Symposium on Formal Methods (SBMF)*, 2007.

[dTOGW13] Paulo de Tarso O. Guerra and Renata Wassermann. On the difference between revising models and revising specification. In *Logical and Semantic Frameworks with Applications (LSFA)*, 2013.

[FKUV86] Ronald Fagin, Gabriel M. Kuper, Jeffrey D. Ullman, and Moshe Y. Vardi. Updating logical databases. In *Advances in Computing Research*, pages 1–18. JAI Press, 1986.

[FPA05] Giorgos Flouris, Dimitris Plexousakis, and Grigoris Antoniou. On applying the agm theory to dls and owl. In *In 4th International Semantic Web Conference (ISWC*, pages 216–231, 2005.

[GW94] Ian P. Gent and Toby Walsh. The sat phase transition. In *ECAI*, pages 105–109, 1994.

[GW10] P. Guerra and Renata Wassermann. Revision of CTL Models. In *Advances in Artificial Intelligence–IBERAMIA 2010*, pages 153–162. Springer, 2010.

[Han91] Sven Ove Hansson. Belief contraction without recovery. *Studia Logica*, 50(2):251–260, 1991.

[Han99] Sven Ove Hansson. *A textbook of belief dynamics*. Applied logic series. Kluwer, 1999.

[KM91] Hirofumi Katsuno and Alberto O. Mendelzon. On the difference between updating a knowledge base and revising it. In James F. Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991.*, pages 387–394. Morgan Kaufmann, 1991.

[KP11] Sébastien Konieczny and Ramón Pino Pérez. Logic based merging. *Journal of Philosophical Logic*, 40(2):239–270, 2011.

[Lev84] Hector J. Levesque. Foundations of a functional approach to knowledge representation. *Artificial Intelligence*, 23(2):155 – 212, 1984.

[LRW12] Renato U. Lundberg, Márcio Moretto Ribeiro, and Renata Wassermann. A framework for empirical evaluation of belief change operators. In Leliane Nunes de Barros, Marcelo Finger, Aurora T. R. Pozo, Gustavo Alberto Giménez Lugo, and Marcos A. Castilho, editors, *SBIA*, volume 7589 of *Lecture Notes in Computer Science*, pages 12–21. Springer, 2012.

[Mak87] David Makinson. On the status of recovery postulate. *Journal of Philosophical Logic*, 16(4):383–394, 1987.

[New82] Alan Newell. The knowledge level. *Artificial Intelligence*, 18(1):87–127, 1982.

[PH12] Bijan Parsia Peter F. Patel-Schneider Sebastian Rudolph Pascal Hitzler, Markus Krötzsch. Owl 2 web ontology language primer. http://www.w3.org/TR/2012/REC-owl2-primer-20121211/, december 2012.

[Rei87] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[RRW14] Fillipe Resina, Márcio Moretto Ribeiro, and Renata Wassermann. Algorithms for multiple contraction and an application to OWL ontologies. In *2014 Brazilian Conference on Intelligent Systems, BRACIS 2014, Sao Paulo, Brazil, October 18-22, 2014*, pages 366–371. IEEE, 2014.

[RW08] Márcio Moretto Ribeiro and Renata Wassermann. The Ontology Reviser Plug-In for Protégé. In *Workshop on Ontologies and Their Applications*, Salvador, 2008.

[RW09a] Márcio Moretto Ribeiro and Renata Wassermann. A{G}{M} Revision in Description Logics. In *Proceedings the IJCAI Workshop on Automated Reasoning about Context and Ontology Evolution (ARCOE)*, 2009.

[RW09b] Márcio M Ribeiro and Renata Wassermann. Base Revision for Ontology Debugging. *Journal of Logic and Computation*, 19(5):721–743, 2009.

[RW10] Márcio Moretto Ribeiro and Renata Wassermann. More About AGM Revision in Description Logics. In *Proceedings the ECAI Workshop on Automated Reasoning about Context and Ontology Evolution (ARCOE)*, 2010.

[RW14] Márcio Moretto Ribeiro and Renata Wassermann. Minimal change in AGM revision for non-classical logics. In Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014*. AAAI Press, 2014.

[RWFA13] Márcio M. Ribeiro, Renata Wassermann, Giorgos Flouris, and Grigoris Antoniou. Minimal change: Relevance and recovery revisited. *Artificial Intelligence*, 201:59 – 80, 2013.

[SLM92] Bart Selman, Hector J. Levesque, and David G. Mitchell. A new method for solving hard satisfiability problems. In William R. Swartout, editor, *AAAI*, pages 440–446. AAAI Press / The MIT Press, 1992.

[vZDDvdT15] Marc van Zee, Dragan Doder, Mehdi Dastani, and Leendert W. N. van der Torre. AGM revision of beliefs about action and time. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3250–3256, 2015.

[Was00] Renata Wassermann. An algorithm for belief revision. In *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000)*. Morgan Kaufmann, 2000.

Renata Wassermann
Department of Computer Science
State University of São Paulo (USP)
Rua do Matão, 1010 - CEP 05508-090 - Butantã, São Paulo - SP
E-mail: renata@ime.usp.br

Márcio Moretto Ribeiro
School of Arts, Sciences and Humanities
State University of São Paulo (USP)
Av. Arlindo Béttio, 1000 - CEP 03828-000 - Ermelino Matarazzo, São Paulo - SP
E-mail: marciomr@usp.br