A Commonsense-Driven Model of Belief Revision for Dynamic Domains using the Event Calculus

Nikoleta Tsampanaki

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Prof. *Dimitris Plexousakis*

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

University of Crete
Computer Science Department

A Commonsense-Driven Model of Belief Revision for Dynamic Domains using the Event Calculus

Thesis submitted by
Nikoleta Tsampanaki
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author: _____
Nikoleta Tsampanaki

Committee approvals: _____
Dimitris Plexousakis
Professor, Thesis Supervisor

_____
Antonios Argyros
Professor, Committee Member

_____
Giorgos Flouris
Principal Researcher, Committee Member

Departmental approval: _____
Antonios Argyros
Professor, Director of Graduate Studies

Heraklion, November 2017

A Commonsense-Driven Model of Belief Revision for Dynamic Domains using the Event Calculus

Abstract

The fields of Artificial Intelligence (AI) and Robotics were strongly connected in the early days of AI, but have since diverged as practitioners of AI focused on problems and algorithms abstracted from the real world, while roboticists, building on their background in mechanical and electrical engineering, concentrated on sensory-motor functions. With advancements in both fields, there is now a renewed interest in bringing the two disciplines closer together. Robots, or any other autonomous entity inhabiting real-world domains, need to deal with incomplete information and uncertainty at various levels of abstraction, from low-level sensory data to high-level knowledge, such as action preconditions and effects. This thesis concentrates on the latter, aiming to keep the Knowledge Base (KB) of an agent both up-to-date and consistent, while performing world-changing or observation actions.

Action theories are well-established logical theories, based on classical logic, for reasoning about domains involving dynamically changing environments. Thus, they can inherently deal with change caused by actions. One of the most prominent action languages is the Event Calculus (EC), which incorporates certain useful features for representing causal and narrative information that differentiate it from other similar formalisms. The EC explicitly represents temporal knowledge, enabling reasoning about the effects of a narrative of events along a time line. Given that the logical theory stored in a KB is not always correct, there is also a need to revise KBs as new information is received. The area of belief revision addresses such a change to a KB. In the well-known AGM postulates, belief revision emerges when one has a knowledge base K and a formula α, and the issue is how to consistently incorporate α in K to obtain a new KB K'. This means that some of the beliefs in the original KB must be retracted, but not all of them, since this would be an unnecessary loss of valuable information. What makes things more complicated is that beliefs in a knowledge base have logical consequences, so when giving up a belief one has to decide as well which of the consequences to retain and which to retract. Thus, belief revision is non-trivial as several different ways for performing this operation may be possible. From the EC perspective, there has been extensive work on epistemic extensions of action languages, in general, as well as, on the main EC formalism. However, little attention has been paid to the problem of automatically revising (correcting) a KB in the EC when an observation contradicts the already inferred knowledge, despite mature work on the belief revision field.

As the current trend in related research is to identify efficient ways to couple high-level task planning with low-level task execution or feasibility checking, the current work aims to empower such combinations, through the delivery of a more generic high-level formalism that lifts some of the unrealistic assumptions

of existing solutions. We propose a generic framework in the context of the EC, along with Answer Set Programming (ASP) encodings of the revision algorithm, accommodating belief revision on top of EC axiomatizations. We consider both the epistemic and non-epistemic case, relying on the possible-worlds representation to give formal semantics to an agent's belief state. We formalize notions of commonsense revisions that take into consideration different knowledge states, such as factual (or observed) knowledge, default, inferred and also unknown knowledge. We present a methodology and an ASP encoding that can implement the formalism, in which we adapt the existing powerful action theories in a more realistic setting. We also present an optimization algorithm aiming to improve the efficiency of the implementation. Finally, we discuss possible future expansions and improvements of our framework and how this work can form the substrate for further extensions concerning a richer set of commonsense features, along with formal results showing that it is generic enough to be applied to different EC dialects.

Ένα μοντέλο κοινής λογικής για την αναθεώρηση πεποιθήσεων σε δυναμικά περιβάλλοντα με χρήση Λογισμού Συμβάντων

Περίληψη

Τα ερευνητικά πεδία της Τεχνητής Νοημοσύνης (ΤΝ) και της Ρομποτικής ήταν στενά συνδεδεμένα κατά τα πρώτα βήματα της ΤΝ, όμως αυτό έπαψε αργότερα να ισχύει, καθώς ο τομέας της ΤΝ επικεντρώθηκε σε θεωρητικά προβλήματα και αλγορίθμους που προήλθαν αφαιρετικά από τον πραγματικό κόσμο, ενώ οι ερευνητές στον τομέα της Ρομποτικής, βασιζόμενοι στο υπόβαθρο τους στη μηχανική και την ηλεκτρολογία, επικεντρώθηκαν στην ανάπτυξη μηχανοκίνητων και αισθητήριων συστημάτων. Όμως , με την συνεχή πρόοδο στους δύο τομείς, υπάρχει πλέον μια ερευνητική τάση για την εκ νέου συνένωση τους. Γενικά, ένα ρομπότ ή οποιοδήποτε άλλο ευφυές αυτόνομο σύστημα που λειτουργεί σε πραγματικό περιβάλλον θα πρέπει να έχει την ικανότητα να αντιμετωπίζει την έλλειψη πληροφορίας και την αβεβαιότητα σε πολλαπλά επίπεδα αφαίρεσης, από δεδομένα αισθητήρων χαμηλού επιπέδου μέχρι υψηλού επιπέδου γνώση, όπως για παράδειγμα προϋποθέσεις και επιδράσεις ενεργειών. Η διπλωματική αυτή εργασία επικεντρώνεται στο τελευταίο και σκοπό έχει να διατηρήσει την βάσης γνώσης (ΒΓ) ενός ευφυούς πράκτορα, τόσο ενημερωμένη όσο και συνεπή, ενώ εκείνος παρατηρεί ή εκτελεί ενέργειες στο περιβάλλον στο οποίο βρίσκεται.

Οι Θεωρίες Ενεργειών είναι εδραιωμένες λογικές θεωρίες, βασισμένες στην κλασική λογική, ανεπτυγμένες για την συλλογιστική διαδικασία πάνω σε τομείς που αφορούν δυναμικά μεταβαλλόμενα περιβάλλοντα. Έτσι, έχουν την δυνατότητα να αντιμετωπίσουν εγγενώς τις αλλαγές που προκαλούνται από ενέργειες στα περιβάλλοντα αυτά. Μια από τις εξέχουσες γλώσσες ενεργειών είναι ο Λογισμός Συμβάντων (ΛΣ), ο οποίος έχει αρκετά χρήσιμα χαρακτηριστικά για την αναπαράσταση πληροφορίας αιτιότητας και ακολουθιών ενεργειών, χαρακτηριστικά που τον διαφοροποιούν από άλλους παρόμοιους φορμαλισμούς. Ο ΛΣ αναπαριστά λεπτομερώς την συνεχώς μεταβαλλόμενη γνώση, επιτρέποντας στη διαδικασία συλλογιστικής να υπολογίσει τις επιδράσεις μιας ακολουθίας γεγονότων, κατά την πάροδο του χρόνου, στο περιβάλλον. Δεδομένου ότι η αποθηκευμένη πληροφορία σε μία ΒΓ δεν είναι πάντοτε σωστή, υπάρχει επίσης ανάγκη για ένα μηχανισμό αναθεώρησης πεποιθήσεων καθώς λαμβάνονται νέες πληροφορίες. Ο τομέας της Αναθεώρησης Πεποιθήσεων (ΑΠ) σκοπό έχει να αντιμετωπίσει τέτοιες αλλαγές σε μία ΒΓ. Στα πιο γνωστά της περιοχής AGM αξιώματα, προκύπτει αναθεώρηση πεποιθήσεων αν, έχοντας μια βάση γνώσεων K και μία πληροφορία α, ενσωματώνουμε αυτή την πληροφορία σε μια νέα βάση γνώσης K′ χωρίς να δημιουργηθεί ασυνέπεια σε αυτή. Αυτό σημαίνει ότι ορισμένες από τις πεποιθήσεις στην αρχική βάση γνώσης θα πρέπει να αναθεωρηθούν, αλλά όχι όλες, καθώς θα προκληθεί άσκοπη απώλεια πολύτιμων πληροφοριών. Αυτό που κάνει τα πράγματα ακόμα πιο περίπλοκα είναι ότι οι πεποιθήσεις σε μια ΒΓ έχουν λογικές συνέπειες, οπότε όταν αναθεωρηθεί μια πεποίθηση, πρέπει να αποφασισθεί και ποιες από τις συνέπειες της θα πρέπει να διατηρηθούν, αλλά και ποιες θα πρέπει να αφαιρεθούν. Έτσι, η αναθεώρηση γνώσης δεν μπορεί να θεωρηθεί μια τετριμμένη διαδικασία, καθώς είναι πιθανόν να υπάρχουν

περισσότεροι από ένας τρόποι για την εκτέλεση της. Από την σκοπιά του ΛΣ, έχουν γίνει εκτενείς εργασίες για την επιστημική επέκταση των γλωσσών ενεργειών, καθώς και του κύριου φορμαλισμού του ΛΣ. Ωστόσο, ελάχιστη προσοχή έχει δοθεί στο πρόβλημα της αυτόματης αναθεώρησης (διόρθωσης) μιας ΒΓ στον ΛΣ όταν μια παρατήρηση αντιβαίνει την ήδη εξαγόμενη γνώση, παρά την εκτενή έρευνα στον τομέα της ΑΠ.

Δεδομένου ότι η τρέχουσα τάση στον συναφή ερευνητικό τομέα είναι να εντοπιστούν αποτελεσματικοί τρόποι συνδυασμού της εύρεσης πλάνων εργασιών υψηλού επιπέδου με την εκτέλεση εργασιών χαμηλού επιπέδου ή τον έλεγχο εφαρμοσιμότητας, η παρούσα εργασία στοχεύει στην ενδυνάμωση τέτοιων συνδυασμών, μέσω ενός γενικού φορμαλισμού, υψηλού επιπέδου, που αναιρεί ορισμένες από τις μη ρεαλιστικές υποθέσεις υπαρχόντων λύσεων. Προτείνουμε ένα πλαίσιο εργασίας από την σκοπιά του ΛΣ, μαζί με μία Answer Set Programming (ASP) κωδικοποίηση του ΑΠ αλγορίθμου που μας επιτρέπει αναθεώρηση πεποιθήσεων βασισμένη στην αξιωματοποίηση του ΛΣ. Λαμβάνουμε υπόψη τόσο την επιστημική όσο και την μη επιστημική περίπτωση, βασιζόμενοι στην αναπαράσταση Πιθανών Κόσμων (ΠΚ), για να μπορέσουμε να ορίσουμε σημασιολογικά την κατάσταση των πεποιθήσεων ενός ευφυούς πράκτορα. Ορίζουμε τυπικά την έννοια της αναθεώρησης πεποιθήσεων κοινής λογικής που λαμβάνει υπόψη διαφορετικα είδη γνώσης, όπως η πραγματική γνώση (ή εκείνη που έχουμε παρατηρήσει στο περιβάλλον), η προκαθορισμένη γνώση, η εξαγόμενη γνώση αλλά και η γνώση που είναι προς το παρόν άγνωστη σε εμάς. Παρουσιάζουμε μία μεθοδολογία και μία ASP κωδικοποίηση που μπορεί να υλοποιήσει τον παραπάνω φορμαλισμό, πάνω στην οποία προσαρμόζουμε τις υπάρχουσες ισχυρές θεωρίες ενεργειών σε ένα πιο ρεαλιστικό πλαίσιο. Παρουσιάζουμε επίσης έναν αλγόριθμο βελτιστοποίησης που στοχεύει στη βελτίωση της αποτελεσματικότητας της υλοποίησης μας. Τέλος, αναφέρουμε πιθανές μελλοντικές επεκτάσεις και βελτιώσεις του συστήματος μας, καθώς και πως το έργο αυτό μπορεί να αποτελέσει τη βάση για περαιτέρω επεκτάσεις που θα μπορούν να υποστηρίξουν ένα πλουσιότερο σύνολο χαρακτηριστικών κοινής λογικής.

Acknowledgements

I would like to thank everyone who helped and supported me to carry through my master thesis.

First of all, I am greatly indebted to my supervisor, Professor Dimitris Plexousakis, for giving me the chance to deal with a very interesting topic of my choice.

I would like to thank my co-supervisors, Theodore Patkos, post-doctoral researcher at ICS-FORTH, and Giorgos Flouris, principle researcher at ICS-FORTH, for their continuous guidance, cooperation and support in the -unknown to me- area of Action theories and Belief revision. Despite their workload, they were always available to answer my questions, and I assure you there was a ton of them. For all the above, I am really grateful.

This work was supported by a post-graduate scholarship from the Institute of Computer Science (ICS) of the Foundation for Research and Technology Hellas (FORTH), which I want to acknowledge for providing financial assistance. In addition, I wish to acknowledge the help and support from the secretariat of ICS-FORTH, Information Systems Laboratory (ISL), Ms. Maria Moutsaki.

Special reference should be made to Antonis Krithinakis for putting up with my stress and productive mess. Thank you for your support and patience. I would also like to extend my thanks to my friends and colleagues at FORTH, with whom I have shared hours of work and joy. Petro, Roula, Makis, Nina, Dimitra, Georgia, Vassili, thank you!

Last but not least, I would like to deeply thank my parents, Markos and Popi, along with my sister Paulina, for their continuous support and love throughout all these years.

How to Write a Master's Thesis

To my parents, Markos and Popi
Στους γονείς μου, Μάρκο και Πόπη

Contents

List of Figures

Chapter 1

Introduction

Robots, or any other autonomous entity inhabiting real-world domains, need to deal with incomplete information and uncertainty at various levels of abstraction, from low-level sensory data to high-level knowledge, such as action preconditions and effects.

Action languages are well-established logical theories for reasoning about the dynamics of changing worlds, aiming at "*formally characterizing the relationship between the knowledge, the perception and the action of autonomous agents*" [52]. One of the most prominent action languages is the Event Calculus(EC) [21, 28], which incorporates certain useful features for representing causal and narrative information that differentiate it from other similar formalisms. The EC explicitly represents temporal knowledge, enabling reasoning about the effects of a narrative of events along a time line. It also relies on a non-monotonic treatment of events, in the sense that by default there are no unexpected effects or event occurrences.

Powerful extensions of the main formalism have been developed to accommodate, for instance, epistemic extensions [27, 26, 34], probabilistic uncertainty [45, 4] or knowledge derivations with non-binary-valued fluents [27]. Moreover, progress in generalizing the stable model semantics used in Answer Set Programming (ASP) has opened the way for the reformulation of EC axiomatizations into logic programs that can be executed with efficient ASP solvers [8]. This allowed for exploiting state-of-the-art tools that outperform previous SAT- or logic programming-based solvers in almost all classes of problems related to practical applications [22].

However, to the best of our knowledge, little work has been done on supporting *belief change* in the EC, in cases when the new information contradicts the already inferred knowledge. Specifically, the existing non-epistemic extensions accommodate *belief update*, which concerns beliefs that change as the result of the realization that the world has changed through some action. The epistemic extensions further focus on modeling the notions of knowledge, thus supporting *belief expansion*, where newly acquired information can enrich the belief state of agents about aspects that were previously considered unknown. Yet, the ability to accommodate, through proper *revisions*, sensed information that contradicts existing

beliefs is not supported. This problem is more general than belief expansion or even than diagnosis, as it not only intends to identify the reasons that explain the contradictions, but also to suggest proper modifications of the belief state of the agent under certain, potentially domain-dependent, criteria [1].

This thesis concentrates on keeping the Knowledge Base (KB) of an agent both up-to-date and consistent, while performing world-changing or observation actions. We present a generic framework that:

- axiomatizes belief revision on top of EC theories, in order to deal with contradictory information
- accommodates both epistemic and non-epistemic cases
- takes into consideration factual (or observed) knowledge, default, inferred and also unknown knowledge
- offers a parameterized cost function making the algorithm's preference relation easily adaptive to any domain specificity
- is implemented using the state-of-the-art logic programming language ASP

This work can form the substrate for further extensions concerning a richer set of commonsense features, along with formal results showing that is generic enough to be applied to different EC dialects.

Example: Consider the classical Yale Shooting scenario, where a loaded gun is fired against a living, walking turkey. An observer may believe that, after the shot, the turkey is dead. If future observations contradict her beliefs, e.g., by noticing that the turkey is still walking, the observer will need to assess different potential revisions of her belief state: can it be that she was so mistaken and the shooter did not fire the gun in the first place? Or is it just that the initial, default belief about the gun being loaded was not accurate? Moreover, how would the revisions be affected if the initial state of the gun is unknown?

Although simplistic, this setting of the Yale Shooting scenario can be generalized to account for different levels of commonsense inferences, some of which may be domain-independent, e.g., revising aspects that were initially unknown rather than aspects that have already been observed, while others may be domain-dependent, e.g., considering certain observations as being less reliable than others. For such types of domains, we develop in the sequel a formal methodology for revising the belief state of an agent, taking into consideration commonsense and epistemic notions, and we also present a more advanced scenario of a medical assistant agent.

The rest of the thesis is organized as follows. In Chapter 2, a review of the basics of the two main research fields is provided, namely belief revision and event calculus, the intersection of which constitute our domain of interest. A description of the two vital topics to our methodology and implementation, namely possible worlds semantics in Event Calculus and answer set programming, is also provided.

In Chapter 3, we focus on relevant literature and discuss other proposed formalizations and frameworks on the problem of automatically revising a Knowledge Base, when a observation contradicts predictions regarding the world. Additionally, we discuss how existing autonomous intelligent systems reason about action and time and how they deal with incomplete information and uncertainty. In Chapter 4, we lay the theoretical underpinnings of our methodology for the commonsense-driven model of belief revision for dynamic domains. Chapter 5 provides a complete description of our framework's implementation and architecture diagram, along with explanatory use case scenarios. Conclusions and possible future expansions of our framework are provided in Chapter 6.

A preliminary version of the proposed framework has been published in the proceedings of the Thirteenth International Symposium on Commonsense Reasoning (Commonsense-2017) [51].

Chapter 2

Preliminaries

The present chapter provides background material necessary to follow the main concepts and theories of this thesis. It comprises two main topics, the intersection of which constitute the domain of interest: belief revision and event calculus, as well as, two more topics vital to our methodology and implementation: possible worlds semantics in EC and ASP.

2.1 Belief Revision

Belief revision (BR) is the process that occurs when a new piece of information that is inconsistent with the present belief system (or database) is added in such a way that the result is a new consistent belief system [11]. This means that some of the beliefs in the belief system must be retracted, but not all of them, since this would be an unnecessary loss of valuable information. What makes things more complicated is that beliefs in a belief system have logical consequences, so when giving up a belief one has to decide as well which of the consequences to retain and which to retract. Thus, BR is non-trivial as several different ways for performing this operation may be possible.

Two kinds of changes are usually distinguished, update and revision. Update the new information is about the situation at present, while the old beliefs refer to the past. It is the operation of changing the old beliefs to take into account the change. In revision both the old beliefs and the new information refer to the same situation. An inconsistency between the new and old information is explained by the possibility of old information being less reliable than the new one. Revision is the process of inserting the new information into the set of old beliefs without generating an inconsistency. In other words, BR is appropriate for modeling static environments about which the agent has only partial and possibly incorrect information. New information is used to fill in gaps and correct errors, but the environment itself does not undergo change. Belief update, on the other hand, is intended for situations in which the environment itself is changing due to the performing of actions.

The main assumption of BR is that of minimal change: the knowledge before and after the change should be as similar as possible. In the case of update, this principle formalizes the assumption of inertia. In the case of revision, this principle enforces as much information as possible to be preserved by the change. The Principle of Consistency maintenance [2] requires that the result of revision should be consistent, while the Principle of Primacy of New Information [2] states that the new observation should always be entailed after the revision.

2.1.1 AGM Postulates

In the well-known AGM postulates [1], BR emerges when one has a knowledge base K and a formula α, and the issue is how to consistently incorporate α in K to obtain a new knowledge base K'. The revision is performed by a binary operator * that takes as its operands the current beliefs and the new information and produces as a result a belief base representing the result of the revision K*α. For completeness, we briefly list the AGM postulates for BR along with a short description, adopted from [35].

K * 1 *$K^*\alpha$ is deductively closed*

K * 2 $\alpha \in K^*\alpha$

K * 3 $K^*\alpha \subseteq K{+}\alpha$ [1]

K * 4 *If* $\neg\alpha \notin K$, *then* $K{+}\alpha \subseteq K^*\alpha$

K * 5 $K^* \alpha = \mathcal{L}$ *iff* $\models \neg\alpha$

K * 6 *If* $\models \alpha \equiv \beta$, *then* $K^*\alpha = K^*\beta$

K * 7 $K^*(\alpha \wedge \beta) \subseteq (K^*\alpha){+}\beta$

K * 8 *If* $\neg\beta \notin K^*\alpha$, *then* $(K^*\alpha){+}\beta \subseteq K^*(\alpha \wedge \beta)$

Postulate (K*1) says that the agent, being an ideal reasoner, remains logically omniscient after she revises her beliefs. Postulate (K*2) says that the new information α should always be included in the new belief set. (K*2) places enormous faith on the reliability of α. The new information is perceived to be so reliable that it prevails over all previous conflicting beliefs, no matter what these beliefs might be. Postulates (K*3) and (K*4) viewed together state that whenever the new information α does not contradict the initial belief set K, there is no reason to remove any of the original beliefs at all; the new belief state K*α will contain the whole of K, the new information α, and whatever follows from the logical closure of K and α (and nothing more). Essentially, (K*3) and (K*4) express the notion

[1] *The + operator denotes expansion: $K{+}\alpha$ is the deductive closure of $K \cup \{\alpha\}$*

of minimal change in the limiting case where the new information is consistent with the initial beliefs. (K*5) says that the agent should aim for consistency at any cost; the only case where it is "acceptable" for the agent to fail is when the new information in itself is inconsistent (in which case, because of (K*2), the agent cannot do anything about it). (K*6) is known as the irrelevance of syntax postulate. It says that the syntax of the new information has no effect on the revision process; all that matters is its content (i.e. the proposition it represents). Hence, logically equivalent sentences α and β change a theory K in the same way. Finally, postulates (K*7) and (K*8) are best understood taken together. They say that for any two sentences α and β, if in revising the initial belief set K by α one is lucky enough to reach a belief set K*α that is consistent with β, then to produce K*($\alpha \wedge \beta$) all that one needs to do is to expand K*α with β; in symbols K*($\alpha \wedge \beta$) = (K*α)+ β. The motivation for (K*7) and (K*8) comes again from the principle of minimal change. The rationale is as follows: K*α is a minimal change of K to include α and therefore there is no way to arrive at K * $\alpha \wedge \beta$ from K with "less change". In fact, because K*($\alpha \wedge \beta$) also includes β one might have to make further changes apart from those needed to include α. If however β is consistent with K*α, these further changes can be limited to simply adding β to K*α and closing under logical implications (no further withdrawals are necessary).

2.2 Event Calculus

Commonsense reasoning can be seen as the process of taking information about certain aspects of a scenario in the world and making inferences about other aspects of the scenario based on a human's commonsense knowledge, or otherwise how the world works. Commonsense reasoning is essential to intelligent behavior and thought. It allows us to fill in the blanks, to reconstruct missing portions of a scenario, to figure out what happened, and to predict what might happen next. Any methodology for automated commonsense reasoning must provide the ability to represent any scenario in the world, as well as, any knowledge about the world and any common sense entities, namely objects, agents, time-varying properties, events, and time. The method must represent and reason about time, space, and mental states, while addressing commonsense phenomena, such as the commonsense law of inertia, preconditions, ramifications, and triggered events. It must also specify processes for reasoning using representations of scenarios and commonsense knowledge [32].

The *EC* is based on first-order logic, which consists of a syntax, semantics and a proof theory, and addresses key issues of commonsense reasoning, such as the ones mentioned above. The basic notions of the EC are events, fluents and timepoints. An event represents an action or event that may occur in the world, such as a person picking up a glass. We use the words *event* and *action* interchangeably. A fluent represents a time-varying property of the world, such as the location of a physical object, while a timepoint represents an instant of time, such as 9:30 am.

An event may occur or happen at a timepoint. A fluent has a truth value at a timepoint or over a timepoint interval; the possible truth values are true and false. After an event occurs, the truth values of the fluents may change. We have commonsense knowledge about the effects of events on fluents. Specifically, we have knowledge about events that initiate fluents and events that terminate fluents. For example, we know that the event of picking up a glass initiates the fluent of holding the glass and that the event of setting down a glass terminates the fluent of holding the glass. We represent these notions by using the EC domain-independent predicates [2].

- $HoldsAt(F, T)$ represents that fluent F is true at timepoint T.
- $Happens(E, T)$ represents that event E occurs at timepoint T.
- $Initiates(E, F, T)$ represents that, if event E occurs at timepoint T, then fluent F will be true after T.
- $Terminates(E, F, T)$ represents that, if event E occurs at timepoint T, then fluent F will be false after T.
- $ReleasedAt(F, T)$ represents that fluent F is released from the commonsense law of inertia at timepoint T and its truth value can fluctuate [3].

Example (cont.) Returning to the Yale shooting example described before, the observer's KB, stating that the gun is loaded at timepoint 0 and fired at timepoint 1, while the turkey is alive and living, can be now described as follows:
$HoldsAt(Alive, 0)$
$HoldsAt(Loaded, 0)$
$Happens(Shoot, 1)$
We also know that the event of loading the gun initiates the fluent of the loaded gun, while the event of shooting the gun terminates the fluent of the loaded gun and the fluent of the living turkey, and are expressed as follows:
$Initiates(Load, Loaded, t)$
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Loaded, t)$
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Alive, t)$

2.2.1 DEC Axioms

Many dialects for the EC have been proposed, dealing with continuous or discrete time, binary or functional fluents etc. For our purposes, we rely on the Discrete time EC, in which timepoints are restricted to the integers. The domain independent set of axioms captures the notions of cause, effect and inertia. We list the

[2] In this section, variables start with a lowercase letter, while constants start with an uppercase letter.

[3] We make use of this predicate only when expressing the commonsense law of inertia.

$\mathcal{DEC}$ axioms used in the proposed methodology, along with a short description, adopted from [32].

Axiom DEC1.
HoldsAt(f,t) ∧ ¬ReleasedAt(f,t+1) ∧ ¬∃ e (Happens(e,t) ∧ Terminates(e,f,t)) ⇒ HoldsAt(f,t+1)

Axiom DEC2.
¬HoldsAt(f,t) ∧ ¬ReleasedAt(f,t+1) ∧ ¬∃ e (Happens(e,t) ∧ Initiates(e,f,t)) ⇒ ¬HoldsAt(f,t+1)

Axiom DEC3.
ReleasedAt(f,t) ∧ ¬∃ e (Happens(e,t) ∧ (Initiates(e,f,t) ∨ Terminates(e,f,t))) ⇒ ReleasedAt(f,t+1)

Axiom DEC4.
¬ReleasedAt(f,t) ∧ ¬∃ e (Happens(e,t) ∧ Releases(e,f,t)) ⇒ ¬ReleasedAt(f,t+1)

Axiom DEC5.
Happens(e,t) ∧ Initiates(e,f,t) ⇒ HoldsAt(f,t+1)

Axiom DEC6.
Happens(e,t) ∧ Terminates(e,f,t) ⇒ ¬HoldsAt(f,t+1)

Axiom DEC1 says that if a fluent is true at timepoint t, the fluent is not released from the commonsense law of inertia at t+1, and the fluent is not terminated by any event that occurs at t, then the fluent is true at t+1. Axiom DEC2 says that if a fluent is false at timepoint t, the fluent is not released from the commonsense law of inertia at t+1, and the fluent is not initiated by any event that occurs at t, then the fluent is false at t+1. Axioms DEC3 and DEC4 express that if a fluent is released from the commonsense law of inertia at timepoint t and the fluent is neither initiated nor terminated by any event that occurs at t, then the fluent is released from the commonsense law of inertia at t+1, and vice versa. The rest of the axioms express how event occurrences influence the states of fluents. Axiom DEC5 says that if a fluent is initiated by some event that occurs at timepoint t, then the fluent is true at t+1. Likewise, axiom DEC6 says that if a fluent is terminated by some event that occurs at timepoint t, then the fluent is false at t+1.

2.3 Possible Worlds in EC

The idea of applying the possible worlds semantics to model knowledge and belief was originally due to Hintikka [16]. The intuitive idea of this paradigm is to acknowledge in the semantics that things might have happened differently from the way they did in fact happen. Thus, besides the true state of affairs (actual world), there are other possible states. Under this interpretation, an agent is said to believe a fact if this is true in all the states that it considers possible. The language that is employed to formalize these ideas is typically some variation of propositional modal logic [7].

If an agent is believed to have complete world knowledge and thus there is one world state considered, then we consider to be in a non-epistemic case in the sequel of this thesis. However, the high demands that are imposed on autonomous systems in real domains have led to variations of Event Calculus theories that can support reasoning with partial world knowledge. Such epistemic extensions can accommodate both known and unknown fluents, using a special type of "sense" actions to acquire new knowledge, which by definition only affect the belief state of the agent, causing no effect to the state of the domain. These cases are considered as epistemic in the sequel of this thesis. The $\mathcal{EFEC}$ [27] dialect is a recently introduced approach that implemented an adaptation of the possible worlds model to give formal semantics to belief predicates.

In $\mathcal{EFEC}$, the function $<>$: $\mathcal{W} \times \mathcal{I} \rightarrow \mathcal{T}$ is introduced to map world/instant pairs to timepoints. Timepoint $< W, I >$ represents instant I in possible world W, where:

$\forall t \exists w, i.t =< w, i >$ (DOX1)

The time lines believed to be accessible at any given moment are captured by the relation $K \subseteq \mathcal{W} \times \mathcal{W}$, which represents the accessibility relation between possible worlds, as in modal logics. As ordinary, we formally define belief of some fluent f at some timepoint as the fact that this fluent has the same truth value in all worlds that are accessible from the actual world W_a:

$Bel(f, < W_a, i >) \equiv$ (DOX2)
$\forall w K(w, W_a) \rightarrow HoldsAt(f, < w, i >)$

$BelNot(f, < W_a, i >) \equiv$ (DOX3)
$\forall w K(w, W_a) \rightarrow \neg HoldsAt(f, < w, i >)$

$BelWh(f, < W_a, i >) \equiv$ (DOX4)
$Bel(f, < W_a, i >) \vee BelNot(f, < W_a, i >)$

We formally define $BelWh()$ as the belief that some fluent f at some timepoint has the same truth value in all worlds that are accessible from the actual world W_a, but we do not define whether this truth value is true or false. Based on DOX4, we can now define $\neg BelWh()$ as the belief that some fluent f at some timepoint is unknown, as it has various truth values among the worlds that are accessible from the actual world W_a.

2.4 Answer Set Programming

The syntax of Answer set programming derives from the Prolog language. We use the syntax of the ASP-Core-2 standard. The semantics of answer set programs is defined by the stable model semantics introduced by Michael Gelfond and Vladimir Lifschitz [13, 24]. Intuitively, according to these semantics, a conclusion is inferred only if there is explicit evidence to support it.

An answer set program consists of a set of rules of the form:
$\alpha : -\beta.$ which represents that α, the head of the rule, is true if β, the body of the

rule, is true. Here is an answer set program:

```
p.
r :- p, not q.
```

The first rule "p." is called a fact. It has an empty body and is written without the ":-" (if) connective. The symbol indicates conjunction ($\wedge$). The token *not* refers to negation as failure and is different from classical negation ($\neg$). The expression *not q* represents that q is not found to be true. We can perform automated reasoning on this program by placing it in a file and running the answer set grounder and solver clingo on the file. The clingo program is a combination of the answer set grounder gringo and the answer set solver clasp. A grounder converts an answer set program containing variables into an equivalent program not containing any variables. It operates by replacing variables with ground terms [12].

The syntax of answer set programs is defined as follows.
A *signature* σ consists of the following disjoint sets:

- A set of constants.
- For every n $\in \{1, 2, 3, \ldots\}$, a set of n-ary function symbols.
- For every n $\in \{0, 1, 2, \ldots\}$, a set of n-ary predicate symbols.

Given a signature σ and a set of variables disjoint from the signature, we define answer set programs as follows.
A term is defined inductively as follows:

- A constant is a term.
- A variable is a term.
- If τ_1 and τ_2 are terms, then -τ , τ_1 + τ_2, τ_1 - τ_2, τ_1 * τ_2, and τ_1 / τ_2 are terms. The symbols +, -, *, and / are the *arithmetic symbols.*
- If ϕ is an n-ary function symbol and $\tau_1, \ldots, \tau_n$ are terms, then $\phi(\tau_1,...,\tau_n)$ is a term.
- Nothing else is a term.

A *ground term* is a term containing no variables and no arithmetic symbols.
An *atom* is defined inductively as follows:

- If ρ is an n-ary predicate symbol and $\tau_1,...,\tau_n$ are terms, then $\rho(\tau_1,...,\tau_n)$ is an atom.
- If ρ is a 0-ary predicate symbol, then ρ is an atom.
- If τ_1 and τ_2 are terms, then $\tau_1 < \tau_2$, $\tau_1 <= \tau_2$, $\tau_1 = \tau_2$, τ_1 != τ_2, $\tau_1 > \tau_2$, and $\tau_1 >= \tau_2$ are atoms. The symbols <, <=, =, !=, >, and >= are the *comparative predicates.*

- Nothing else is an atom.

A *ground atom* is an atom containing no variables, no arithmetic symbols, and no comparative predicates.
A *rule* is
α_1 | ... | α_k :- β_1,..., β_m, not γ_1,..., not γ_n.
where α_1,...,α_k, β_1,..., β_m, γ_1,..., γ_n are atoms. α_1 | ... | α_k is the head of the rule, and β_1,..., β_m, not γ_1,..., not γ_n is its body.

A *fact* is a rule whose body is empty (m = 0 and n = 0).
A *constraint* is a rule whose head is empty (k = 0).
A *ground rule* is a rule containing no variables, no arithmetic symbols, and no comparative predicates.
A *logic program, answer set program*, or *program* is a set of rules.
A *traditional rule* is a rule whose head contains a single atom (k=1).
A *traditional program* is a set of traditional rules.
A *ground program* is a program containing no variables, no arithmetic symbols, and no comparative predicates.
Answer set programming languages, such as those of lparse, gringo, and DLV, and the standard ASP-Core-2, further specify the following:

- Constants are integers or start with a lowercase letter.
- Variables start with an uppercase letter.
- Function symbols start with a lowercase letter.
- Predicate symbols start with a lowercase letter.

The stable models semantics of ground programs are defined as follows.
An interpretation I is a set of ground atoms. If α is a ground atom, then $\alpha \in$ I represents that α is true, whereas $\alpha \notin$ I represents that α is false. An interpretation I satisfies a rule
α_1 | ... | α_k :- β_1,..., β_m, not γ_1,..., not γ_n.
if and only if
$\{\alpha_1,...,\alpha_k\} \cap \mathrm{I} \neq \varnothing$ or
$\{\beta_1,..., \beta_m\} \nsubseteq \mathrm{I}$ or
$\{\gamma_1,...,\gamma_n\} \cap \mathrm{I} \neq \varnothing$
That is, a rule is satisfied by an interpretation if and only if the head of the rule is satisfied by the interpretation if the body of the rule is satisfied by the interpretation.
An interpretation I is a *model* of a program Π if and only if I satisfies r for every r $\in \Pi$.
The *reduct* of a program Π relative to an interpretation I, written Π^I is the set of all rules
α_1 | ... | α_k :- β_1,..., β_m

such that
α_1 | ... | α_k :- β_1,..., β_m, not γ_1,..., not γ_n.
is an element of Π, and $\{\gamma_1,...,\gamma_n\} \cap I = \varnothing$.
An interpretation I is a *stable model* of a program Π if and only if I is a minimal model of Π^I relative to set inclusion.
A choice rule is
$\{\alpha_1;...;\alpha_k\}$:- $\beta_1,...,\beta_m$, not γ_1,..., not γ_n.
where $\alpha_1,...,\alpha_k$, $\beta_1,...,\beta_m$, γ_1,..., γ_n are atoms. This specifies that, if the body is satisfied, then any subset of the atoms in the head can be true.

Chapter 3

Related Work

The previous chapter described the main theories necessary for the comprehension of this thesis; namely, event calculus and belief revision. As the present thesis is targeted on the integration and expansion of these two fields into a unified generic framework, this chapter studies in more detail other formalizations and proposed frameworks on the problem of automatically revising a Knowledge Base when an observation contradicts predictions regarding the world. Despite mature work on the related belief revision field, adapting such results for the case of action theories is non-trivial. Furthermore, this chapter studies how existing autonomous intelligent systems reason about action and time, as well as, how they deal with incomplete information and uncertainty in real world domains. We will discuss what they offer, their reasoning and/or inference mechanisms along with the way they represent their knowledge. Although there are works taking uncertainty into account, they rather restrict conflicting states than revise their knowledge base to accommodate the new observation. Ultimately, there are other works handling contradictions and unexpected observations by using mechanisms inherent in the definition of active logics [33]. However, such approaches are not studied in detail, as our work uses a meta-level approach.

3.1 Belief Revision and Action Theories

Belief change (also known as *belief revision*) is a mature field of study dealing with the adaptation of a KB in the face of new information [1]. Most works in the classical belief change literature are dealing with the so-called *classical logics* [9], which have certain nice properties, both in terms of semantics (monotonicity, compactness, inclusion of the classical tautological implication, etc) and in terms of syntax (closed with respect to the usual operators $\wedge, \vee, \neg$, etc.). Extensions of these theories to apply for ontological languages [9, 37], or compact and monotonic logics in general [38] have been considered as well. However, to the best of our knowledge, no such study exists for non-monotonic formalisms, partly because many non-monotonic formalisms (most notably, defeasible logic, default logic and

paraconsistent logics) have inherent ways to reason under inconsistency without trivializing inference. Thus, technical results from the related literature are not directly applicable in our setting.

Studies that account for epistemic considerations of the Event Calculus are more closely related to ours. More specifically, the $\mathcal{EFEC}$ variant introduced in [27, 26] is the first to rely on the possible worlds semantics to reason about knowledge. $\mathcal{EFEC}$ supports a multitude of features, such as reasoning about the future and past, or dealing with non-determinism and concurrency. Our work utilizes the same underlying structures to formalize the treatment of epistemic notions, extending them with the ability to revise contradicting knowledge, although it is currently significantly limited in the set of supporting features. In [34], a different epistemic extension of discrete-time Event Calculus theories is presented, using a deduction-oriented rather than possible-worlds based model of knowledge.

Beyond the Event Calculus, possible-worlds based epistemic extensions for reasoning about actions and knowledge have been developed in the context of other calculi. The first approach that inspired this direction of research is owed to [29], who presented a Kripke-like formulation of epistemic notions of modal logic in action languages by reifying possible worlds as situations. [40] adapted this framework in the Situation Calculus, using possible situations to specify how the mental state of an agent should change with ordinary and sense actions, providing also a solution to the frame problem for knowledge. Other studies introduced further features: [50] adapted the model in the context of the Fluent Calculus, [41] covered concurrent actions, while [19] introduced epistemic modalities for groups of agents. Non-possible-worlds based epistemic action frameworks include [30, 5, 46, 36, 54, 25]. In all these frameworks, knowledge is assumed to be always correct and observations that contradict inferred knowledge will lead to inconsistency.

The ability to deal with belief changes has lately started to gain interest within other action languages, as in [43] and [42] in the Situation Calculus, but without taking time into account. [53] developed a new action formalism for revision of temporal belief bases; even though related to our work, [53] do not directly address the problem of revising theories in Event Calculus, but instead define a new logic of action that is closer to propositional logic, thereby allowing technical results from the belief change literature to be directly applicable in their framework.

3.2 Robotics and Autonomous Intelligent Systems

Robots, or any other autonomous intelligent entity inhabiting real-world domains, need to deal with incomplete information and uncertainty at various levels of abstraction, from low-level sensory data to high-level knowledge, such as action preconditions and effects. A prevalent knowledge processing system for robots is the KnowRob system [47, 48], that combines knowledge representation and reasoning methods with techniques for acquiring knowledge and for grounding the knowledge

in a physical system. It can serve as a common semantic framework for integrating information from different sources, as it combines static encyclopedic knowledge, common-sense knowledge, task descriptions, environment models, object information and information about observed actions that has been acquired from various sources (manually axiomatized, derived from observations, or imported from the web). It supports different deterministic and probabilistic reasoning mechanisms, clustering, classification and segmentation methods, and includes query interfaces as well as visualization tools.

Several European research projects use KnowRob for representing knowledge to be exchanged between robots (RoboEarth [1]), for integrating information from the Web with task demonstrations given by humans (RoboHow [2]), for elderly-care robots (SRS [3]), for assembly tasks in industrial applications (SMErobotics [4]), for reasoning about safe human-robot cooperation (SAPHARI [5]), and for multi-robot search-and-rescue tasks in alpine disaster scenarios (SHERPA [6]). Other applications include underwater robotics and multi-player on-line games that use the spatio-temporal object representation for virtual scenes.

KnowRob is based on SWI Prolog [56] and the knowledge about classes of objects and actions is represented in the form of an ontology, implemented in the Web Ontology Language (OWL) [31], that provides the vocabulary for describing knowledge about actions, events, objects, spatial and temporal information. The OWL ontology is loaded into the system using SWI Prolog's Semantic Web library [55], representing their triple structure as Prolog predicates. In [49], it is argued that instead of abstracting the data into a purely symbolic knowledge base, KnowRob shall rather keep the original, high-resolution, continuous data and only compute a "symbolic view" on it as needed. This on-demand abstraction can always provide the appropriate level of detail: Robotics algorithms can operate on the original data, logical inferences can be computed on a more abstract symbolic level. Queries can combine inferences made at all levels of abstraction.

An open question about KnowRob is the integration of the results of different inference mechanisms, especially if they produce conflicting results. Namely, if the robot knows the locations of some objects, it can compute locations for novel object types based on their semantic similarity to the known ones [6]. If it does not have such information, it can use abstract rules such as "a refrigerator is the storage place for perishable item". By combining knowledge about the properties of objects (to infer whether the object at hand is perishable) with the semantic environment map (to locate an instance of a refrigerator), the knowledge processing system can compute where to search for objects. While these different approaches can co-exist in the knowledge base and independently generate location hypotheses,

[1] http://www.roboearth.org
[2] http://www.robohow.eu
[3] http://www.srs-project.eu
[4] http://www.smerobotics.org
[5] http://saphari.eu
[6] http://sherpa-project.eu

the integration of the results is an open issue.

Similar integrated knowledge processing systems for robots that have been developed over the past few years are also discussed. The ORO ontology by Lemaignan et al. [23], is focused on human-robot interaction and on resolving ambiguities in dialog situations. This capability was for example described by Ros et al. [39], where the robot inferred, based on its knowledge about the objects in the environment and their properties, which queries it should ask to disambiguate a command. ORO uses OWL as a representation format and a standard DL reasoner for inference. An underlying three-dimensional geometrical environment representation by Simeon et al.[44], serves for computing spatial information and for updating the internal belief state about the positions of objects.

Another project that deals with representing knowledge and reasoning with it under strong uncertainty is [15]. They present a mobile robot platform along with a theory of how it can plan information gathering tasks and explain failure in environments that have uncertain sensing, uncertain actions, incomplete information and epistemic goals. Central to their ideas is how the robot's knowledge is organized as well as how the robot should represent what it knows or believes. [17] proposes reasoning methods that take uncertainty into account. It presents a methodology for allowing flexibility in task execution using qualitative approaches, which support representation of spatial and temporal flexibility with respect to tasks. However, to the best of our knowledge, in all these works discussed, they try to avoid inconsistent states altogether while executing dynamically generated plans. Otherwise, in case of unexpected failure, they try to explain and reason about this failure, rather than try to revise the state that led them to inconsistency.

Chapter 4

Methodology

In this chapter we develop the methodology of our commonsense-driven model of belief revision for dynamic domains. Our account of change and causality is based on the discrete time Event Calculus (DEC) axiomatization, while the modeling of possible worlds for representing epistemic notions is inspired by the epistemic extension of the Functional Event Calculus ($\mathcal{EFEC}$) [27], (see Chapter 2).

4.1 Notation

In the sequel of this thesis, predicate symbols, function symbols, and constants start with an uppercase letter, while variables start with a lowercase letter. Wherever not explicitly stated, variables are assumed to be universally quantified. In ASP code quotation, predicate symbols, function symbols, and constants start with a lowercase letter, while variables start with an uppercase letter.

4.2 Revision Setting and Principles

The representation of a dynamic domain for reasoning with the Event Calculus requires the coupling of the domain-independent axioms with definitions of domain-dependent axioms that describe the dynamics of the world we are interested in modeling. The domain dependent part involves the predicates $Happens()$, $Initiates()$ and $Terminates()$ and their logical formulas for representing the effects of actions, triggered events etc. The notions of cause, effect and inertia are captured in the $\mathcal{DEC}$ domain independent set of axioms (see Subsection 2.2.1). As we restrict our considerations to deterministic domains for the time being, we do not axiomatize fluents that are released from the law inertia, therefore we make use of the *ReseasedAt* and *Releases* predicates of $\mathcal{DEC}$ only to express the commonsense law of inertia. Overall, the knowledge base (KB) of an agent is described formally as follows:

Definition 4.2.1 *A Knowledge Base (KB) capturing a dynamic domain is defined as* $\Phi = \mathcal{DEC} \wedge \Sigma \wedge \Gamma^0 \wedge \Delta \wedge \Omega$ *where*

- $\mathcal{DEC}$ *is the conjunction of the Discrete Event Calculus domain-independent axioms,*
- Σ *is the conjunction of the domain-dependent axioms,*
- Γ^0 *is the initial knowledge, i.e., a conjunction of ground* $(\neg)HoldsAt(F_i, 0)$ *axioms at timepoint 0,*
- Δ *is the narrative of actions, i.e., a conjunction of ground* $Happens(E_i, T_j)$ *axioms,*
- Ω *is a conjunction of unique name axioms.*

Domain axioms in Δ can be partially defined and then minimized to address the Frame Problem and related issues. Γ^0 axioms cannot be partially defined, as we assume complete world knowledge initially (this assumption will be lifted as detailed below). We denote by $\Phi \models \phi$ the fact that Φ implies ϕ. We assume that from time to time we observe some part of the world, i.e., we obtain the truth value of certain fluents. Our current assumption is that observations can only contain a conjunction of $(\neg)HoldsAt()$ statements. We denote by Γ^T an observation obtained at timepoint T.

Example (cont.) Returning to the Yale shooting example described before, the observer's KB can be expressed based on the following axiomatization:

$Initiates(Load, Loaded, t)$ (1.1)
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Loaded, t)$ (1.2)
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Alive, t)$ (1.3)
$HoldsAt(Alive, 0)$ (1.4)
$HoldsAt(Loaded, 0)$ (1.5)
$Happens(Shoot, 1)$ (1.6)

That is, $\Sigma = (1.1) \wedge (1.2) \wedge (1.3)$, $\Gamma^0 = (1.4) \wedge (1.5)$ and $\Delta = (1.6)$, whereas the remaining components of Φ_{Yale} follow from Definition 4.2.1.

In order to support epistemic reasoning, we introduce two new sorts, in the style of $\mathcal{EFEC}$: a sort $\mathcal{W}$ for representing possible worlds (variables $w, w', w_1, ...$) and a sort $\mathcal{I}$ for instants (variables $i, i', i_1, ...$). The idea is to represent time as a system of parallel lines, where each world is understood as an identifier for a possible time line. We assume that the constant W_a of sort $\mathcal{W}$ signifies the actual world (see Subsection 2.3). In contrast to $\mathcal{EFEC}$ though, we do not consider the accessibility relation between possible worlds K to be an equivalence relation. Instead, in order to model belief rather than knowledge, we only assume that K is serial, which is equivalent to stating that the agent cannot believe contradictions (also known as the Consistency Axiom):

$\forall i.Bel(f, < W_a, i >) \rightarrow \neg BelNot(f, < W_a, i >)$ $(DOX5)$

$\forall i.BelNot(f, < W_a, i >) \rightarrow \neg Bel(f, < W_a, i >)$ $(DOX6)$

Notice that from the above axioms we do not assume that the actual world is accessible too (K is not reflexive). As a result, erroneous beliefs can still be inferred, requiring a revision mechanism whenever observations (that reflect W_a) do not comply with the agent's beliefs. Additionally, we need to define a domain-independent axiom to ensure the existence of possible worlds in the initial state.

$$\begin{aligned}&\forall f.\neg BelWh(f, < W_a, 0 >) \rightarrow \\ &\exists w_1, w_2.K(w_1, W_a) \wedge K(w_2, W_a) \wedge \\ &HoldsAt(f, < w_1, 0 >) \wedge \neg HoldsAt(f, < w_2, 0 >)\end{aligned} \quad \text{(DOX7)}$$

Notice that according to our assumption of never losing knowledge (there is no non-determinism), it is sufficient to preserve the number of possible worlds generated at the initial timepoint while reasoning, since there is no way of generating more worlds. We do not need to eliminate worlds either, in order to allow for reasoning about the past.

Based on the aforementioned formalization of belief, we extend the definition of a KB to accommodate lack of knowledge at the initial and at future timepoints:

Definition 4.2.2 *An epistemic KB is defined as* $e\Phi = \mathcal{DEC} \wedge \mathcal{DOX} \wedge \Sigma \wedge e\Gamma^0 \wedge \Delta \wedge \Omega$*, where*

- $\mathcal{DEC}$ *is the conjunction of the Discrete Event Calculus domain-independent axioms,*
- $\mathcal{DOX}$ *is the conjunction of the epistemic axioms to support beliefs,*
- Σ *is the conjunction of the domain-dependent axioms,*
- $e\Gamma^0$ *is the initial beliefs, i.e., a conjunction of ground* $Bel(F_i, < W_a, 0 >)$, $BelNot(F_j, < W_a, 0 >)$ *axioms at timepoint 0,*
- Δ *is the narrative of actions, i.e., a conjunction of ground* $Happens(E_i, < w, I_j >)$ *axioms,*
- Ω *is a conjunction of unique name axioms.*

Example (cont.) Returning to the Yale shooting example described before, assuming that the observer does not know whether the gun is loaded or not, the observer's KB can be expressed as follows: $\Sigma = (1.1) \wedge (1.2) \wedge (1.3)$, $e\Gamma^0 =$ (1.4) and $\Delta = (1.6)$, whereas the remaining components of $e\Phi_{Yale}$ follow from Definition 4.2.2.

The definition for $e\Phi$ is more general than the one given for Φ. Specifically, if we assume complete world knowledge at timepoint 0, a single possible world is generated, making $e\Phi$ equivalent to Φ. As before, Δ axioms can be partially defined and then minimized to address the Frame Problem and related issues. eΓ^0

axioms can be partially defined as well; fluents that are unknown at time instant 0 generate the set of possible worlds according to axiom (DOX7).

4.3 The Revision Operator

Now let's turn our attention to the problem of revising a KB $e\Phi$ with an observation Γ^T. Apart from the Principles mentioned in the Subsection 2.1, the special characteristics of the Event Calculus force us to introduce two new principles. The first is the *Principle of Persistence of Background Knowledge*, which states that the revision process will only affect the initial knowledge ($e\Gamma^0$) and/or the narrative (Δ). Thus, the domain-independent ($\mathcal{DEC}$) and domain-dependent (Σ) axioms, as well as the unique name axioms (Ω), should not be affected. This avoids issues associated to the problem of learning the domain from observations, which is not in the scope of this thesis. The second new requirement is the *Principle of Disallowing Proactive Change*, which, informally, states that we cannot use an observation referring to time T in order to add events in the narrative beyond that timepoint. Essentially, this limits the direct effects of an observation (and the corresponding revision) to past timepoints, even though such effects may also have indirect ramifications related to the truth value of fluents in the future. Finally, we adopt the *Principle of Minimal Change* [18] (also known as the *Principle of Persistence of Prior Knowledge* [2]), which states that the new KB should be as "close" as possible to the original KB; in other words, from all the possible change results (revision candidates) that satisfy the other principles, we should choose the one that retains "the most information".

Following the above principles, we can formally define the set of *revision candidates* as follows:

Definition 4.3.1 *Given a KB* $e\Phi = \mathcal{DEC} \wedge \mathcal{DOX} \wedge \Sigma \wedge e\Gamma^0 \wedge \Delta \wedge \Omega$ *and an observation* Γ^T*, a KB* $e\Phi'$ *is a revision candidate of* $e\Phi$ *with* Γ^T *iff:*

- $e\Phi'$ *is of the form* $e\Phi' = \mathcal{DEC} \wedge \mathcal{DOX} \wedge \Sigma \wedge e\Gamma'^0 \wedge \Delta' \wedge \Omega$ *(Principle of Persistence of Background Knowledge).*
- *No formula in* $(\Delta \setminus \Delta') \cup (\Delta' \setminus \Delta)$ *refers to timepoints* $t > T$ *(Principle of Disallowing Proactive Change).*
- $e\Phi'$ *is a consistent KB (Principle of Consistency).*
- $e\Phi' \models \Gamma^T$ *(Principle of Primacy of New Information).*

The set of all revision candidates of $e\Phi$ *with* Γ^T *will be denoted by* $RC(e\Phi, \Gamma^T)$.

Note that Definition 4.3.1 imposes that the part $\mathcal{DEC} \wedge \mathcal{DOX} \wedge \Sigma \wedge \Omega$ of all revision candidates is identical to the corresponding part of the original KB (following the Principle of Persistence of Background Knowledge), and also formalizes all other principles (except from the Principle of Minimal Change). The latter

is not considered because $RC(e\Phi, \Gamma^T)$ is meant to represent all the *conceivably possible* revision results, not the optimal ones. The notion of minimal change is often subjective, context- and/or domain-dependent, so we chose to include it as a separately configurable component of our framework.

To formalize the Principle of Minimal Change, we will use the standard approach of introducing a *preference relation* $\prec^T_{e\Phi}$, which will be presented in detail. The idea is that if $e\Phi_1 \prec^T_{e\Phi} e\Phi_2$, $e\Phi_1$ is strictly more preferred than $e\Phi_2$ for the result of the revision of $e\Phi$ with an observation at timepoint T. Note that this is different from the relations among interpretations [18] and formulas [10] that have been used elsewhere for the same purpose. Establishing the connection among our preference relation and these works is part of our future work. It suffices to assume that $\prec^T_{e\Phi}$ is well founded (so that we can always find a minimal element in a non-empty set).

We are now ready to define the revision operator. Intuitively, the idea is that we select those elements of $RC(e\Phi, \Gamma^T)$ that are minimal with respect to $\prec^T_{e\Phi}$. In case multiple minimal elements exist, their disjunction is taken. It is also interesting to note that, in the special case when $RC(e\Phi, \Gamma^T) = \emptyset$, we do not revise the KB; this can happen, e.g., when the observation itself is inconsistent or when there is no way to satisfy the observation without changing background knowledge, such as the domain axioms. Formally:

Definition 4.3.2 *The revision operator* $\bullet$ *is a binary operator, defined as follows:*

- $e\Phi \bullet \Gamma^T = e\Phi$ *if* $RC(e\Phi, \Gamma^T) = \emptyset$.
- $e\Phi \bullet \Gamma^T = \bigvee\{e\Phi' \mid e\Phi' \in RC(e\Phi, \Gamma^T)$ *and there is no* $e\Phi'' \in RC(e\Phi, \Gamma^T)$ *such that* $e\Phi'' \prec^T_{e\Phi} e\Phi'\}$ *otherwise.*

4.4 Default Fluents

Commonsense reasoning aims to simulate the human ability to make presumptions about the type and essence of ordinary situations they encounter every day. These assumptions include judgments about the physical properties, purpose, intentions and state of people and objects, as well as possible outcomes of their actions and interactions. Drawing inspiration from this notion, we define some fluents with a specific default predicate in the domain axiomatization of the KB and we fill in missing parts of their initialization with a predefined truth value, if we believe that these fluents normally behave in a specific way. In case of an event happening that affects the truth value of the fluent, it ceases to be considered as default and is treated like the rest of the fluents.

We rely on the technique of reification to give formal semantics to default fluents by using the special predicate $Def(f)$ for each of the regular fluents. This special predicate behaves like the rest of the fluents, and if it holds at a specific timepoint it represents that the fluent f is considered default at this specific

timepoint. We also make use of the default negation (not), as well as, the strict negation($\neg$) to define the following domain-independent axioms. The

$\forall f$ not $HoldsAt(Def(f), t) \rightarrow \neg HoldsAt(Def(f), t)$ (DEF1)
$\forall f$ not $(\neg HoldsAt(Def(f), 0)) \wedge$ not $(HoldsAt(f, 0) \vee \neg HoldsAt(f, 0)) \rightarrow$
$HoldsAt(Def(f), 0)$ (DEF2)
$\forall F\ HoldsAt(Def(F), 0) \wedge$ not $(HoldsAt(F, 0) \vee$ not $HoldsAt(F, 0)) \rightarrow$
$(\neg) HoldsAt(F, 0)$ (DEF3)

Axiom DEF1 expresses that if fluents are not explicitly defined as default fluent, then they are indeed non-default. Axiom DEF2 expresses that if some fluents are not defined as non-default fluent and are not initially defined with a truth value at timepoint 0, then they should be consider as default. Finally, DEF3 expresses that if some fluents are considered default and are not initially defined with a truth value at timepoint 0, then they should be instantiated with the predefined default fluents truth value, either as True ($HoldsAt()$) or False ($\neg HoldsAt()$).

Additionally, for each of the domain-dependent logical formulas that represent the effects of actions on the domain fluents, e.g.,
$\bigwedge_i$ $(\neg)$ $HoldsAt(F_i, t) \rightarrow Terminates(E, F, t)$ or
$\bigwedge_i$ $(\neg)$ $HoldsAt(F_i, t) \rightarrow Initiates(E, F, t)$
it is necessary to add an extra logical formula, ensuring that if the truth value of a default fluent changes, then it ceases to be considered as default, as follows:
$(\bigwedge_i$ $(\neg)$ $HoldsAt(F_i, t)) \wedge HoldsAt(Def(F), t) \rightarrow Terminates(E, Def(F), t)$.

Example (cont.) Returning to the Yale shooting scenario, assume now that the observer considers the loaded gun as predefined default fluent, having the truth value of true, then based on axiom 4.4.1 and DEF3, the fluent loaded would be initiated as expressed in 4.4.2.
$HoldsAt(Def(Loaded), 0)$ (4.4.1)
$HoldsAt(Def(Loaded), 0) \wedge$ not $(HoldsAt(Loaded, 0) \vee$ not $HoldsAt(Loaded, 0))$
$\rightarrow HoldsAt(Loaded, 0)$ (4.4.2)

The Yale shooting domain-dependent axioms contain the 4.4.3 axiom. Thus, we also need to add 4.4.4 as previously described, in order to ensure that the fluent will cease to be considered as default, if its truth value changes.
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Loaded, t)$ (4.4.3)
$HoldsAt(Def(Loaded), t) \rightarrow Terminates(Shoot, Def(Loaded), t)$ (4.4.4)

4.5 Preference Relation

To define the preference relation more precisely, we will leverage a parameterized *cost-based model* which assesses minimality based on the amount of information

lost, modified or gained from the original KB in order to accommodate the observation. In particular, considering two KBs $e\Phi, e\Phi'$ the *cost* to move from $e\Phi$ to $e\Phi'$ will be defined on the basis of the formulas that can be inferred by one of these KBs but not the other.

The preference relation is based on a cost function, and depends on a timepoint and the initial $e\Phi$. Let us define the cost function as $C(e\Phi, e\Phi', T)$, that is:
C: $KB \times KB \times \mathcal{T} \mapsto N$. Thus, for every KB and T, we may define a preference relation $\leq^{C,e\Phi,T}$, or simply $\leq$ such that: $e\Phi' \leq e\Phi''$ iff $C(e\Phi, e\Phi', T) \leq C(e\Phi, e\Phi'', T)$.

A composite preference, is a stratification of multiple sub-preferences, divided in prioritized levels. In a composite preference, we first find those KBs optimal with respect to the most important sub-preference, then those optimal with respect to the second most important sub-preference relation and so on. Formally, for each of the sub-preferences $P_1, ..., P_N$, we can define a prioritized composite preference relation, by making use of the prioritized preference symbol & [14, 20], as follows:
$P = P_1 \& P_2 \& ... \& P_N$.

To formalize the proposed preference relation, we first define the following sets, in order to be able to describe the changes observed between the initial KB and the considered revision candidate KB:

Modified non-Default Knowledge $MK^I(e\Phi, e\Phi') =$
$\{BelWh(F, < W_a, I' >) \mid I' \leq I$ and $\neg HoldsAt(Def(F), I')$, and either $e\Phi \models Bel(F, < W_a, I' >)$ and $e\Phi' \models BelNot(F, < W_a, I' >)$, or $e\Phi \models BelNot(F, < W_a, I' >)$ and $e\Phi' \models Bel(F, < W_a, I' >)\}$. This represents all $Bel()$ or $BelNot()$ statements about non-default fluents whose truth value was changed during the transition from $e\Phi$ to $e\Phi'$, up to I.

Modified Default Knowledge $MDK^I(e\Phi, e\Phi') =$
$\{BelWh(F, < W_a, I' >) \mid I' \leq I$ and $HoldsAt(Def(F), I')$, and either $e\Phi \models Bel(F, < W_a, I' >)$ and $e\Phi' \models BelNot(F, < W_a, I' >)$, or $e\Phi \models BelNot(F, < W_a, I' >)$ and $e\Phi' \models Bel(F, < W_a, I' >)\}$. This represents all $Bel()$ or $BelNot()$ statements about default fluents whose truth value was changed during the transition from $e\Phi$ to $e\Phi'$, up to I.

New Knowledge $NK^I(e\Phi, e\Phi') =$
$\{BelWh(F, < W_a, I' >) \mid I' \leq I$, and $e\Phi \models \neg BelWh(F, < W_a, I' >)$, and $e\Phi' \models BelWh(F, < W_a, I' >)$. This represents all $\neg BelWh()$ statements whose unknown value was changed to known during the transition from $e\Phi$ to $e\Phi'$, up to I.

Lost Knowledge $LK^I(e\Phi, e\Phi') =$
$\{\neg BelWh(F, < W_a, I' >) \mid I' \leq I$, and $e\Phi \models BelWh(F, < W_a, I' >)$ and $e\Phi' \models \neg BelWh(F, < W_a, I' >)$. This represents all $Bel()$ or $BelNot()$ statements whose truth value was changed to unknown during the transition from $e\Phi$ to $e\Phi'$, up to I.

New Events $NE^I(e\Phi, e\Phi') = \{Happens(E, < W_a, I' >) \mid I' \leq I,$
$e\Phi \models \neg Happens(E, < W_a, I' >)$ and $e\Phi' \models Happens(E, < W_a, I' >)\}$.
This represents all events that we had to add in the narrative of $e\Phi$ to accommodate the observation, up to I.

Lost Events $LE^I(e\Phi, e\Phi') = \{Happens(E, < W_a, I' >) \mid I' \leq I,$
$e\Phi \models Happens(E, < W_a, I' >)$ and $e\Phi' \models \neg Happens(E, < W_a, I' >)\}$. This represents all events that we had to retract from the narrative of $e\Phi$ to accommodate the observation, up to I.

Note that the above definitions do not consider the consequences of changes for future timepoints, i.e., beyond a certain instant I, or timepoint T. This will be used to ignore any future repercussions of our changes, considering only the changes up to the timepoint of the observation.

Any preference relation can be adapted to any domain requirements by adding, removing or changing any of the sub-preference relations. The belief revision mechanism can be also adjusted according to the domain, accommodating cases where specific changes in the knowledge base matter more than others, by altering the priority among the sub-preference relations.

We propose a composite preference relation based on five sub-preference relations partitioning the above mentioned sets in different "importance categories". We define each of the sub-preference relation stratum along with their corresponding cost functions, each of which make use of the same weight function that degrades the weight over time, expressing the intuition that it should be more expensive to change knowledge about past timepoints than knowledge about more recent timepoints. This weight function was also chosen for simplicity reasons, but it can always be parameterized to specific weights for each of the sub-preference relations. The weight function is defined as follows:

Definition 4.5.1 (Weight function)

$W : \mathcal{T} \mapsto \mathbb{Z}$, *such that:* $w^{T'} = T - T' + 1 \mid T' \leq T$.
This represents that the weight at timepoint T' increases linearly as we move towards the beginning of time.

Example (cont.) Returning to the Yale shooting scenario once again, assume now that we consider a revision candidate $e\Phi'$ that has retracted an event from the initial $e\Phi$ event narrative, namely that the event of shooting did not happen at timepoint 1, then the weight function would return: $w^1 = 2 - 1 + 1 = 2$.

Now we are able to define the prioritized sub-preference relations along with their corresponding cost functions. On the first level, we consider the most important notion of having the minimal changes related to events, namely lost or new events. On the second level, we express the notion of having the least lost knowledge possible in general. On the third level, we consider having the least

modified knowledge about non-default fluents, as it is more likely to prefer fewer changes on this set, rather than the modified default knowledge set, which is based on predefined, "arbitrary" in a way, truth values, as a result this is the next considered level. Last but now least, the fifth level expresses the notion that although it is a positive aspect or revision to gain new knowledge, yet we still want to have the fewer changes possible in respect to the initial KB.

Definition 4.5.2 (Sub-Preference Relations Cost Functions)

$cost_1^T(e\Phi, e\Phi') = w^T \cdot |NE^T(e\Phi, e\Phi')| + w^T \cdot |LE^T(e\Phi, e\Phi')|$

$cost_2^T(e\Phi, e\Phi') = w^T \cdot |LK^T(e\Phi, e\Phi')|$

$cost_3^T(e\Phi, e\Phi') = w^T \cdot |MK^T(e\Phi, e\Phi')|$

$cost_4^T(e\Phi, e\Phi') = w^T \cdot |MDK^T(e\Phi, e\Phi')|$

$cost_5^T(e\Phi, e\Phi') = w^T \cdot |NK^T(e\Phi, e\Phi')|$

Definition 4.5.3 (Sub-Preference Relations Definition)

For $i = 1, \ldots, 5$, we define the relation $\leq_i^{C,e\Phi,T}$ as follows:
$e\Phi_1 \leq_i e\Phi_2$ *iff* $cost_i^T(e\Phi, e\Phi_1) \leq cost_i^T(e\Phi, e\Phi_2)$.

Now, the $\prec_{e\Phi}^T$ relation can be easily defined as the stratification of the aforementioned sub-preference relations, as follows:

Definition 4.5.4 (Composite Preference Relation)

$e\Phi_1 \prec_{e\Phi}^T e\Phi_2$ *iff there is* $j \in \{1, .., 5\}$ *such that:*
$cost_i^T(e\Phi, e\Phi_1) = cost_i^T(e\Phi, e\Phi_2)$ *for* $i < j$ *and* $cost_j^T(e\Phi, e\Phi_1) < cost_j^T(e\Phi, e\Phi_2)$

Informally, the revision candidate $e\Phi_1$ is preferred over $e\Phi_2$ if and only if there is a sub-preference relation, in which the cost is greater for $e\Phi_2$ than $e\Phi_1$. If the revision candidates has equal costs on an upper level, we proceed to check the next sub-preference level downwards. Exactly one sub-preference relation is enough to declare a revision candidate preferred over the other, as the sub-preference relations are stratified.

Finally, we can define the non-strict $\leq$ as follows:

Definition 4.5.5 (Non-Strict Preference Relation)

$\Phi_1 \prec \Phi_2$ *iff* $\Phi_1 \prec_{e\Phi}^T \Phi_2$ *or* $cost_i^T(\Phi, \Phi_1) = cost_i^T(\Phi, \Phi_2)\ \forall\ i$

The proposed $\prec_{e\Phi}^T$ relation has several intuitively desirable formal properties. First, we show that "fewer" changes (with respect to the standard set-theoretic subset relation) are better than "more":

Proposition 1 *Consider three KBs $e\Phi, e\Phi_1, e\Phi_2$. Set $C^T(e\Phi_i) = \{\phi \mid e\Phi \models \phi, e\Phi_i \not\models \phi$ and ϕ refers to a timepoint $t' \leq T\}$, for $i = 1, 2$. If $C^T(e\Phi_1) \subset C^T(e\Phi_2)$, then $e\Phi_1 \prec_{e\Phi}^T e\Phi_2$.*

As a corollary, we get that not changing a KB is always cheaper than changing it, and this will happen whenever the observation does not contradict our expectations:

Proposition 2 *$e\Phi \prec^T_{e\Phi} e\Phi'$ for all $e\Phi, e\Phi', T$.*

Proposition 3 *If $e\Phi \in RC(e\Phi, \Gamma^T)$, then $e\Phi \bullet \Gamma^T = e\Phi$.*

Proposition 4 *If $e\Phi \models \Gamma^T$, then $e\Phi \bullet \Gamma^T = e\Phi$.*

4.6 Epistemic Case reduced to Non-Epistemic

The proposed methodology can be used for both the epistemic and the simplistic non-epistemic case, without alternations. Namely, having initially complete world knowledge, a single world is generated, which contains only $Bel()$ and $BelNot()$, but not $BelWh()$ statements as defined in (DOX7) Subsection 4.2, making Γ^0 equivalent to $e\Gamma^0$, and so $e\Phi$ equivalent to Φ, too. Consequently, minimality of changes is assessed based on the amount of information modified from the original KB in order to accommodate the observation. In particular, considering two KBs Φ, Φ' the *cost* to move from Φ to Φ' will be defined on the basis of the modified knowledge sets (both non-default or default), as well as, new and lost event sets, as defined in Subsection 4.5, but not on the basis of the new knowledge set, as there is no unknown fluent to be changed to known during the transition from Φ to Φ'.

Chapter 5

Implementation & Use Cases

In this chapter we present an implementation of the proposed framework, containing the architecture of the developed system along with the ASP encoding that implements the proposed belief revision formalism, accommodating both the non-epistemic and the epistemic case. We also present the optimization mechanism that we developed for better efficiency. Last but not least, we present use-case scenarios appropriate for the demonstration of the proposed system, as well as how its parameterized priorities among the preference relations affect the belief revision mechanism results.

5.1 Architecture

The developed system implements the steps that an agent performs whenever a new observation arrives, in order to accommodate the new piece of information, using the architecture shown in Figure 5.1. The figure shows the two main reasoning steps performed, along with their corresponding input/output modules (rulesets), and two Java parsers, responsible for the interconnection of the aforementioned reasoning steps. Reasoning is performed by implementing the Event Calculus axiomatizations as ASP rules and by utilizing the Clingo reasoner, version 4.5.4 [12]. The Java parsers are essentially parts of a pure Java program, responsible for automating the process and for connecting the reasoning to the outside world.

At the starting point, when a new observation or event arrives, the Java parser makes use of the domain-dependent axioms and calculates a relation graph among the domain fluents and the new observation/event. Thus, the revision mechanism considers only the relevant fluents and events to the observation. All other irrelevant fluents are excluded, but their initial information is stored for future use (more at Subsection 5.2).

Then, the first reasoning step is executed. It consists of the Discrete Event Calculus (DEC) axioms, the domain-dependent axioms, the initial beliefs, and the main program, all implemented in ASP. The initial beliefs module contains the initial information about the world state, specifically at timepoint 0, and the

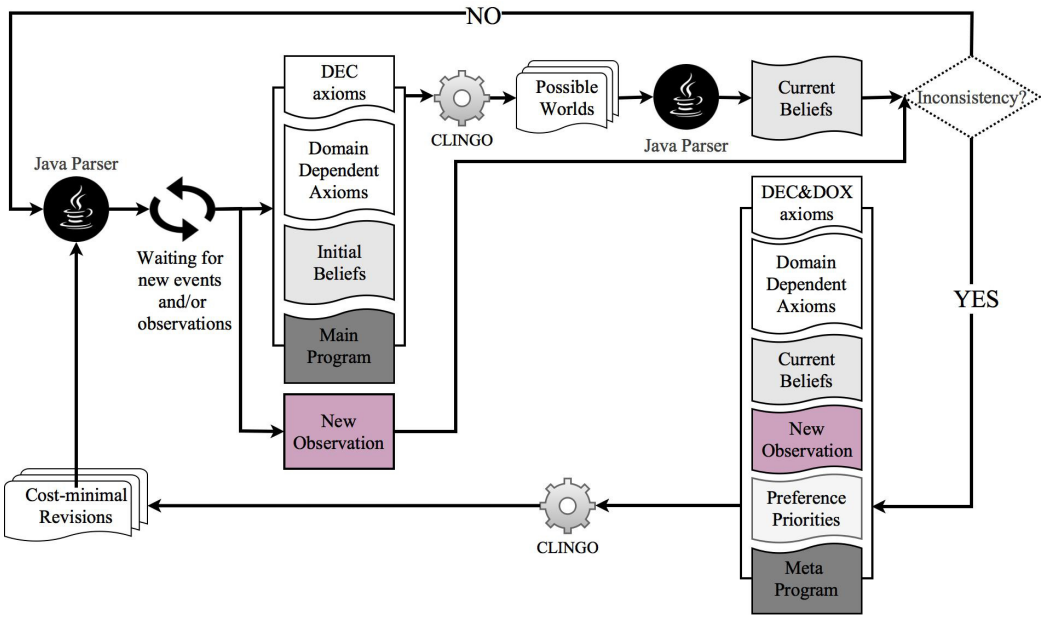

Figure 5.1: The reasoning loop for revising the KB of an agent.

narrative of actions, as well as which of the fluents are considered default, e.g., expressed in ASP:

```
(not) holdsAt(F1,0).
happens(E1,1).
holdsAt(def(F1),0).
```

For each of the predefined default fluents and based on the relation graph, there is a rule in the domain-dependent axiomatization ensuring that these fluents cease to be considered defaults as soon as their truth value changes. This can be caused by an event occurrence, if the preconditions hold.

```
terminates(e1,def(F),T) :- time(T), holdsAt(F1,T),
holdsAt(def(F1),T).
```

The main program module contains a constraint that ensures inertia as our considerations are restricted to deterministic domains, meaning that no fluent is released from the law of inertia. It also contains a rule responsible for the generation of possible worlds, when there are fluents believed to be unknown, and rules necessary for the representation of the default fluents and how they take the predefined truth value whenever unknown. All rules are expressed in ASP as follows:

```
:- {not releasedAt(F,0)}0, fluent(F).
```

```
{holdsAt(F,0)} :- fluent(F).
(not) holdsAt(F,0) :- holdsAt(def(F),0).
```

The first rule is a constraint prohibiting any fluent to be released at any timepoint. The second rule is a choice rule that initiates every fluent as true or false at timepoint 0, thus creating possible worlds. Last but not least, the last rule initiates each of the fluents defined as default with the predefined truth value of true (or false).

Essentially, the first reasoning step generates the running information about the world state up to the timepoint that the new observation was received (timepoint Tmax), taking into consideration the narrative of actions and how those affected the world/fluent truth values. It produces one answer set in the non-epistemic case and multiple answer sets in the epistemic case, each denoting a possible world (see Section 4.2).

Then, a Java parser intervenes to transform the information contained in the generated answer sets, i.e., the possible worlds, into the agent's belief predicates, as explained in Section 4.2. Three special ASP predicates, named believesOriginal(F,T), believesNotOriginal(F,T), unknownOriginal(F,T), are used to represent whether a fluent is believed to be true, false or unknown respectively in each timepoint, and are stored in the module named current beliefs. This module and the new observations module, which is also implemented as an ASP constraint, are used to examine whether an inconsistency arises.

```
:- not holdsAt(f1,(W,t1)), world(W).
OR
:- holdsAt(f1,(W,t1)), world(W).
```

The above rule is a constraint, which makes sure that the observation, here is about fluent f1, must always be true or false, among all possible worlds.

In case of no conflict, the program goes back to the initial state of the loop, accepting the new information and waiting for new events or observations. In case of inconsistency, the second reasoning step is performed, taking as input the new observations module, the current beliefs module and the number of worlds that were produced in the first reasoning step. It also consists of the Discrete Event Calculus (DEC) axioms along with the DOX axioms (see Section 4.2), the domain-dependent axioms, the preference priorities and the meta program, all implemented in ASP.

5.1.1 Belief Revision Algorithm

The meta-program is the most important module as it implements the revision algorithm. It computes the revision using the cost-optimal revisions generator, via a logic program. This program generates all possible combinations of fluents in the initial state (timepoint 0), as well as combinations of event occurrences at each timepoint, for every possible world (the number of which is taken as input), aiming

at keeping only the combinations that lead to a consistent KB and are consistent with the new observation (revision candidates), expressed in ASP as follows:

```
{happens(E,T)} :- time(T), event(E).
{holdsAt(F,(W,0))} :- fluent(F), world(W).
```

The cost associated with each revision candidate is calculated, based on the cost functions described in Section 4.5; this is implemented with ASP rules that penalize each fluent truth value or event occurrence that is different from the output of the current beliefs module. We first present the new and lost event sets, then the lost knowledge set, then the modified non-default knowledge set, then the modified default knowledge sets, and at last we present the new knowledge set, all expressed in ASP. Each of the rules, is activated when a corresponding change is detected between the current KB and the revision candidate KB. Note that new and lost events activate different rules with the same rule head, as in our methodology we consider those in the same specific sub-preference relation.

```
penaltyE(X,E,T):- happens(E,T) , not happensOriginal(E,T),
time(T), event(E), weight(T,X).
penaltyE(X,E,T):- not happens(E,T) , happensOriginal(E,T),
time(T), event(E), weight(T,X).

penaltyLK(X,F,T):- unknown(F,T) , believesOriginal(F,T),
time(T), fluent(F), weight(T,X).
penaltyLK(X,F,T):- unknown(F,T) , believesNotOriginal(F,T),
time(T), fluent(F), weight(T,X).

penaltyMK(X,F,T):- believes(F,T) , believesNotOriginal(F,T),
not believesOriginal(def(F),T), time(T), fluent(F), weight(T,X).
penaltyMK(X,F,T):- believesNot(F,T) , believesOriginal(F,T),
not believesOriginal(def(F),T), time(T), fluent(F), weight(T,X).

penaltyMDK(X,F,T):- believes(F,T) , believesNotOriginal(F,T),
believesOriginal(def(F),T), time(T), fluent(F), weight(T,X).
penaltyMDK(X,F,T):- believesNot(F,T) , believesOriginal(F,T),
believesOriginal(def(F),T), time(T), fluent(F), weight(T,X).

penaltyNK(X,F,T):- believes(F,T) , unknownOriginal(F,T),
time(T), fluent(F), weight(T,X).
penaltyNK(X,F,T):- believesNot(F,T) , unknownOriginal(F,T),
time(T), fluent(F), weight(T,X).
```

Here, we present an ASP encoding of the proposed weight function that we defined in Definition 4.5.1. By maxstep we denote the timepoint that we received the new observation, while with T we denote any timepoint less or equal to maxstep:

```
weight(T,X):- time(T), X = maxstep - T+1.
```

Here, we present all sub-preference relations associated cost functions as defined in Subsection 4.5, all expressed in ASP:

```
costE(N) :- N = #sum{ X,F,T : penaltyE(X,F,T)}.
costLK(N) :- N = #sum{ X,F,T : penaltyLK(X,F,T)}.
costMK(N) :- N = #sum{ X,F,T : penaltyMK(X,F,T)}.
costMDK(N) :- N = #sum{ X,F,T : penaltyMDK(X,F,T)}.
costNK(N) :- N = #sum{ X,F,T : penaltyNK(X,F,T)}.
```

Finally, optimization statements filter out all the revision candidates (answer sets) gradually, according to the predefined preference priorities. The Java parser responsible for the execution of the second reasoning program takes as input a CSV file, containing the priorities among the sub-preference relations (the greater the priority number the more 'important' the preference) and transforms them into ASP, as follows:

```
#minimize{S@5 : costE(S)}.
#minimize{S@4 : costLK(S)}.
#minimize{S@3 : costMK(S)}.
#minimize{S@2 : costMDK(S)}.
#minimize{S@1 : costNK(S)}.
```

More specifically, the revision candidates are filtered through five phases. Firstly, the Clingo reasoner "keeps" those revision candidates/answer sets that have the least event occurrences changes among all. Among the remaining equal answer sets, the reasoner "keeps" those that have the least lost information possible, next those that have the least modified non-default knowledge, and so on. At the end, the revision program returns the answer sets that are now considered optimal revisions.

We have set the priority among the preference based on the intuition that it is highly irregular for us to lose an event occurrence or to add an event to the narrative arbitrarily. The next most important notion is to have the least possible lost information when revising the KB. Finally, it is more important to have minimum modifications to non-default knowledge, than to default, as the latter is something that is predefined (i.e., assumed) but not actually observed. The last preference concerns new knowledge. It stands to reason that acquiring knowledge is a positive aspect of the revision, but we have to minimize the changes, in order to be closer to the original KB state. Note, of course, that the priorities can be adjusted according to the demands and special characteristics of each individual domain.

5.2 Optimization

Although complete and functional, the proposed implementation has a specific exponential complexity, which can be optimized for better efficiency. Thus, we implemented an optimization mechanism that is embedded in our program. Prior to the description of the optimization algorithm, we provide an analysis of the computational complexity of our program for the process of belief revision, as well as a consideration of the worst and the more realistic case scenarios.

5.2.1 Complexity Analysis

The complexity of an answer set program stems from the number of terms that the reasoner has to ground for the computation of answer sets. At first, a problem is expressed as a logic program, and a grounder systematically replaces all variables in the program by (variable-free) terms. Then, the solver takes the resulting propositional program and computes its answer sets (or aggregations of them). Let α be an atom and X be a set of atoms, for a normal logic program P, deciding whether X is a stable model of P is P-complete, and deciding whether α is in a stable model of P is NP-complete. For a normal logic program P with optimization statements, deciding whether X is a stable model of P is co-NP-complete, and deciding whether α is in a stable model of P is Δ_2^P-complete. The aforementioned complexity results apply to propositional programs only. For capturing the complexity of he first-order case, we note that the ground instantiations grd(P) of a first-order program P is exponential in the size of P.

We make use of the Clingo reasoner which includes the combination of the grounder Gringo and the solver Clasp [12]. The complexity of our program, the second reasoning step, stems from the computation of $RC(e\Phi, \Gamma^T)$, as the meta-program generates all possible combinations of fluents in the initial state, that is for timepoint 0, for every possible world, as well as all combinations of event occurrences at each timepoint.

According to the possible-worlds specifications, the number of possible worlds depends on the number of unknown fluents, i.e., in a domain of F fluents, U of which are unknown, we then have 2^U possible worlds, where U$\leq$F. Supposing the worst case scenario, where all F fluents are unknown, the complexity of initializing all fluents at timepoint 0, either as holding or not, for every possible world is $O(2^F * 2^F) = O(2^{2F})$. Additionally, the complexity of generating all possible event occurrences combinations at each timepoint, supposing that we have E events and T timepoints, is $O(2^E * (T-1))$ (we make no use of event occurrences at timepoint T as their effects are shown in the next timepoint that we do not consider). But, when computing all the $RC(e\Phi, \Gamma^T)$, fluent initializations and event narratives are combined, leading to a complexity of $O(2^{2F} * 2^E * (T-1)) = O(2^{2F+E} * (T-1))$.

However, it is highly unusual in having all fluents unknown in the domain. Regardless, we make use of another notion to downscale the number of fluents and events combination generated in our belief revision algorithm. The notion is:

"Do we need to examine any combinations of fluents and events that are irrelevant with the new conflicting observation?" Probably not, as their truth value or their existence in the narrative of events does not affect the fluent concerning the observation. Thus, let us introduce the term of *Irrelevant Fluents and/or Events* and give a formal definition for them.

5.2.2 Irrelevant Fluents and Events

The optimization idea is to restrict the revision mechanism on fluents and events that their truth value does not affect the truth value of Γ^T, so they are irrelevant to it. All other fluents are considered relevant. The following definition express formally those notions:

Definition 5.2.1 *Consider a KB Φ, a fluent F, an event E and a timepoint T. We denote by:*

- Φ^{+F} = $\Phi \setminus$ { *HoldsAt(F,0)*, $\neg$ *HoldsAt(F,0)* } $\cup$ { *HoldsAt(F,0)* }
- Φ^{-F} = $\Phi \setminus$ { *HoldsAt(F,0)*, $\neg$ *HoldsAt(F,0)* } $\cup$ { $\neg$ *HoldsAt(F,0)* }
- $\Phi^{+E,T}$ = $\Phi \setminus$ { *Happens(E,T)*, $\neg$ *Happens(E,T)* } $\cup$ { *Happens(E,T)* }
- $\Phi^{-E,T}$ = $\Phi \setminus$ { *Happens(E,T)*, $\neg$ *Happens(E,T)* } $\cup$ { $\neg$ *Happens(E,T)* }

Definition 5.2.2 *Given a KB $\Phi = \mathcal{DEC} \wedge \Sigma \wedge \Gamma^0 \wedge \Delta \wedge \Omega$, we denote by $\bar{\Phi} = \mathcal{DEC} \wedge \Sigma \wedge \Omega$ the part of the KB consisting of the Discrete Event Calculus domain-independent axioms, the domain-dependent axioms and the unique name axioms respectively. Two KBs Φ_1, Φ_2 will be called axiomatization-equivalent denoted by $\Phi_1 \equiv_a \Phi_2$ if and only if $\bar{\Phi}_1 \equiv \bar{\Phi}_2$, where $\equiv$ is the standard logical equivalence.*

Definition 5.2.3 *Consider a KB Φ and an observation Γ.*

- *We say that a fluent F is* irrelevant *to the KB Φ and the observation Γ, if and only if the following holds for all $\Phi' \equiv_a \Phi$: $\Phi'^{+F} \vdash \Gamma$ if and only if $\Phi'^{-F} \vdash \Gamma$.*

- *We say that an event E is* irrelevant *to the KB Φ and the observation Γ, if and only if the following holds for all $\Phi' \equiv_a \Phi$: $\Phi'^{+E,T} \vdash \Gamma$ if and only if $\Phi'^{-E,T} \vdash \Gamma$.*

Intuitively, Φ^{+F} and Φ^{-F} represent KBs that the initial truth value of a fluent F has been retracted and the truth values of true and false have been added, respectively. Likewise, $\Phi^{+E,T}$ and $\Phi^{-E,T}$ represent KBs that an event information at a specific timepoint has been retracted and the event occurrence or the absence of it have been added in the the two respective KBs. Those KB definitions are needed to express that if a fluent or event does not affect Γ through Initiates or Terminates axioms, then the alternation of their truth values does not affect Γ, and as a result those fluents/events are irrelevant.

5.2.3 Optimization Algorithm

Now we are ready to describe the optimization algorithm. The first Java program parses the new observation module/file to define which fluent is mentioned in it. Then it parses the domain-dependent axioms module and finds which of the fluents and events are related to this specific fluent and which are not, as described above. Then, it generates two new files of domain-dependent axioms and initial beliefs, containing only the relevant rules, choice rules, constraints, initial information at timepoint 0, etc. Those modules/files are used in place of the original whenever necessary. Thus, the proposed revision program deals with relevant fluents only. The program complexity may still be of the same order $O(2^{2F+E} * (T-1))$, but in fact the terms that have to be grounded are fewer.

5.3 Use Case Scenarios

Although simplistic, the use case scenarios presented next can be generalized to account for more complex domains with larger knowledge bases. Firstly, we will present the introductory example of the classical Yale Shooting scenario comprehensively for completeness reasons, and then we will describe some variations of another more advanced scenario of a medical assistant agent.

5.3.1 Yale Shooting Scenario

Consider the classical Yale Shooting scenario, where a loaded gun is fired against a living, walking turkey. An observer may believe that, after the shot, the turkey is dead. If future observations contradict her beliefs, e.g., by noticing that the turkey is still walking, the observer will need to assess different potential revisions of her belief state: can it be that she was so mistaken and the shooter did not fire the gun in the first place? Or is it just that the initial, default belief about the gun being loaded was not accurate? Moreover, how would the revisions be affected if the initial state of the gun is unknown?

5.3.1.1 Non-Epistemic Case

To begin with, let us assume complete world knowledge initially. The observer's KB Φ_{Yale} can be described by the following axiomatization, stating that the gun is loaded at timepoint 0 and fired at timepoint 1.

$Initiates(Load, Loaded, t)$ (1.1)
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Loaded, t)$ (1.2)
$HoldsAt(Loaded, t) \rightarrow Terminates(Shoot, Alive, t)$ (1.3)
$HoldsAt(Alive, 0)$ (1.4)
$HoldsAt(Loaded, 0)$ (1.5)
$Happens(Shoot, 1)$ (1.6)

That is, $\Sigma = (1.1) \wedge (1.2) \wedge (1.3)$, $\Gamma^0 = (1.4) \wedge (1.5)$ and $\Delta = (1.6)$, whereas the

```
----------Relevant Fluents/Events----------
Fluent: alive
    Events:shoot
    Precondition fluent:loaded
Fluent: loaded
    Events:load
    Events:shoot
    Precondition fluent:loaded
----------Relevant Fluents/Events----------
```

Figure 5.2: Yale Shooting scenario: Relevant Fluents/Events Algorithm Output

```
------Possible worlds------
---------World0------------
alive 0
loaded 0
shoot 1
alive 1
loaded 1
---------------------------
```

Figure 5.3: Yale Shooting scenario non-Epistemic Case: The output of the first reasoning step; namely, the possible worlds.

remaining components of Φ_{Yale} follow from Definition 4.2.1. It can be shown that $\Phi_{Yale} \models \neg HoldsAt(Alive, 2)$.

Assume now that the observer receives information that contradicts her current inferences, e.g., $\Gamma^3 = HoldsAt(Alive, 3)$ (note that $\Phi_{Yale} \not\models HoldsAt(Alive, 3)$). A possible reaction to this observation would be that the observer was mistaken and the shooter did not fire the gun. That is, $\Delta' = \emptyset$ and $\Gamma'^0 = \Gamma^0$. So, a revision candidate of Φ_{Yale}, would be $\Phi'_{Yale} = \mathcal{DEC} \wedge \Sigma \wedge \Gamma'^0 \wedge \Delta' \wedge \Omega$.
Another possible revision would be that the observer was mistaken and the gun was not loaded in the first place.
That is, $\Delta'' = \Delta$ and $\Gamma''^0 = (1.4) \wedge \neg HoldsAt(Loaded, 0)$. The revision candidate of Φ_{Yale} is now $\Phi''_{Yale} = \mathcal{DEC} \wedge \Sigma \wedge \Gamma''^0 \wedge \Delta'' \wedge \Omega$.

Consequently, $\Phi'_{Yale}, \Phi''_{Yale} \in RC(\Phi_{Yale}, \Gamma^3)$. Note that many other KBs are included in $RC(\Phi_{Yale}, \Gamma^3)$, but all of them would introduce more changes (with regards to the subset relation) than these two (see also Proposition 1), so they are not considered for the sake of simplicity.

To find the $\prec^3_{e\Phi}$-minimal element(s) of $RC(\Phi_{Yale}, \Gamma^3)$, we only need to compare Φ' with Φ''. The weight associated with the sub-preference cost functions is set to $w(T') = T - T' + 1$, so the corresponding costs are: $cost^3_1(\Phi_{Yale}, \Phi'_{Yale}) = 3$, $cost^3_1(\Phi_{Yale}, \Phi''_{Yale}) = 0$, so $\Phi''_{Yale} \prec^3_{e\Phi} \Phi'_{Yale}$ and Φ''_{Yale} is the revision result, as it keeps the narrative of events intact. Notice that we do not need to check for the second sub-preference relation as there is inequality from the first level of preferences.

We present the output of the relevant fluents/events search algorithm, as well as the output of the first reasoning step; namely, the possible worlds, in Figures 5.2 and 5.3. As we have assumed complete initial knowledge, we have a non-epistemic case and as a result there is one possible world. Finally, we present the optimal revision that accommodates the new observation into our knowledge base in Figure 5.4.

```
------------------------------------------------------------
--------------------Current Beliefs---------------------
-----------------------FLUENTS--------------------------
Fluent\Timepoint        0       1       2       3
loaded                  Bel     Bel     BelNot  BelNot
alive                   Bel     Bel     BelNot  BelNot
-----------------------EVENTS---------------------------
Event\Timepoint         0       1       2       3
load                    -       -       -       -
shoot                   -       Happ    -       -
------------------------------------------------------------
------------------------------------------------------------
--------------------Revised Beliefs---------------------
-----------------Optimal Revision No1-------------------
-----------------------FLUENTS--------------------------
Fluent\Timepoint        0       1       2       3
loaded                  BelNot  BelNot  BelNot  BelNot
alive                   Bel     Bel     Bel     Bel
-----------------------EVENTS---------------------------
Event\Timepoint         0       1       2       3
load                    -       -       -       -
shoot                   -       Happ    -       -
------------------------------------------------------------
------------------------------------------------------------
```

Figure 5.4: Yale Shooting scenario non-Epistemic Case: The output of the Java agent, divided in two sections. The upper section in blue represents the original belief state, and the lower section in green represents the revised belief state of the agent.

```
------Possible worlds------
---------World1------------
alive 0
shoot 1
alive 1
alive 2
alive 3
---------World2------------
alive 0
shoot 1
alive 1
loaded 1
loaded 0
----------------------------
```

Figure 5.5: Yale Shooting scenario Epistemic Case: The output of the first reasoning step; namely, the possible worlds.

5.3.1.2 Epistemic Case

Consider now the epistemic case of the program; namely, the initial beliefs are that the turkey is alive at timepoint 0 and a shot happens at timepoint 1, but we do not know whether the gun is loaded or not at timepoint 0 (resulting in possible worlds shown in Figure 5.5). Thus, we do not know whether the turkey is alive or not at timepoint 3. Assume now that we receive a new contradicting information that the turkey is alive at timepoint 3. We present the optimal revision that accommodates this observation into observer's knowledge base in Figure 5.6.

More specifically, the optimal revision is to assume that we were mistaken and the gun was not loaded in the first place. Thus, we have the least possible cost, as we gain knowledge on the state of the turkey at timepoints 2 and 3, as well as, on the state of the gun at timepoints 0 and 1, while retaining the event narrative intact. Had we accepted the revision that the shooter did not fire the gun, we would lose knowledge and event occurrences at various timepoints, and as a result, the cost would be greater at the first level of sub-preference relations.

5.3.2 Medical Assistant Agent

Consider a medical assistant agent who observes an ill patient and tries to cure him. Based on the symptoms, the agent believes that the patient has either disease 1 or 2 (assume that the two diseases cannot co-exist). The agent also believes that the patient is normally reliable and takes the proposed disease cure. Initially, the agent does not know which of the diseases the patient has, but it believes that the patient took the proposed, more mild, "cure 1" at timepoint 0. Thus, the agent cannot decide what to believe about the patient, if he is cured or not. If future observations alter its beliefs, e.g., by noticing that the patient is still ill, the agent

```
------------------------------------------------------
--------------------Current Beliefs-------------------
----------------------FLUENTS-------------------------
Fluent\Timepoint        0       1       2       3
loaded                  ~BelWh  ~BelWh  BelNot  BelNot
alive                   Bel     Bel     ~BelWh  ~BelWh
----------------------EVENTS--------------------------
Event\Timepoint         0       1       2       3
load                    -       -       -       -
shoot                   -       Happ    -       -
------------------------------------------------------
------------------------------------------------------
--------------------Revised Beliefs-------------------
-----------------Optimal Revision No1-----------------
----------------------FLUENTS-------------------------
Fluent\Timepoint        0       1       2       3
loaded                  BelNot  BelNot  BelNot  BelNot
alive                   Bel     Bel     Bel     Bel
----------------------EVENTS--------------------------
Event\Timepoint         0       1       2       3
load                    -       -       -       -
shoot                   -       Happ    -       -
------------------------------------------------------
------------------------------------------------------
```

Figure 5.6: Yale Shooting scenario Epistemic Case: The output of the Java agent, divided in two sections. The upper section in blue represents the original belief state, and the lower section in green represents the revised belief state of the agent.

will need to assess different potential revisions of its belief state: can it be that it was mistaken and did not propose the correct medication in the first place? Or is it just that the initial, default belief about the patient being reliable was not accurate?

5.3.2.1 Round 1

To begin with, the agent's KB $e\Phi_{disease}$ can be described by the following axiomatization, stating that the patient is sick and took cure 1 at timepoint 0.

$$HoldsAt(ReliablePatient, t) \wedge \neg HoldsAt(Disease2, t) \wedge HoldsAt(Disease1, t) \rightarrow Initiates(ProposeCure1, CuredPatient, t) \quad (2.1)$$

$$HoldsAt(ReliablePatient, t) \wedge \neg HoldsAt(CuredPatient, t) \wedge HoldsAt(Disease1, t) \rightarrow Terminates(ProposeCure1, Disease1, t) \quad (2.2)$$

$$HoldsAt(ReliablePatient, t) \wedge \neg HoldsAt(Disease1, t) \wedge HoldsAt(Disease2, t) \rightarrow Initiates(ProposeCure2, CuredPatient, t) \quad (2.3)$$

$$HoldsAt(ReliablePatient, t) \wedge \neg HoldsAt(CuredPatient, t) \wedge$$

$$HoldsAt(Disease2, t) \rightarrow Terminates(ProposeCure2, Disease2, t) \quad (2.4)$$

$$\neg HoldsAt(ReliablePatient, t) \rightarrow \\ Initiates(ChangeAttitude, ReliablePatient, t) \quad (2.5)$$

$$HoldsAt(ReliablePatient, t) \rightarrow \\ Terminates(ChangeAttitude, ReliablePatient, t) \quad (2.6)$$

$$HoldsAt(Def(ReliablePatient), 0) \quad (2.7)$$

$$\neg HoldsAt(CuredPatient, 0) \quad (2.8)$$

$$Happens(ProposeCure1, 0) \quad (2.9)$$

That is, $\Sigma = (2.1) \wedge (2.2) \wedge (2.3) \wedge (2.4) \wedge (2.5) \wedge (2.6)$, $e\Gamma^0 = (2.7) \wedge (2.8)$ and $\Delta = (2.9)$, whereas the remaining components of $e\Phi_{disease}$ follow from Definition 4.2.2. It can be shown that $e\Phi_{disease} \models \neg BelWh(CuredPatient, 2)$.

Assume now that the agent receives information that contradicts its current inferences, e.g., $\Gamma^2 = BelNot(CuredPatient, 2)$ (note that $e\Phi_{disease} \not\models BelNot(CuredPatient, 2)$). A possible reaction to this observation would be that the agent proposed the wrong cure because patient has disease 2. That is, $\Gamma'^0 = \Gamma^0 \wedge HoldsAt(Disease2, 0)$. So, a revision candidate of $e\Phi_{disease}$, would be $e\Phi'_{disease} = \mathcal{DEC} \wedge \Sigma \wedge \Gamma'^0 \wedge \Delta \wedge \Omega$.

Another possible revision would be that the agent was mistaken and the patient did not take the proposed medication in the first place.
That is, $\Delta'' = \emptyset$ and $\Gamma''^0 = (2.8) \wedge \neg HoldsAt(ReliablePatient, 0)$. The revision candidate of $e\Phi_{disease}$ is now $e\Phi''_{disease} = \mathcal{DEC} \wedge \Sigma \wedge \Gamma''^0 \wedge \Delta'' \wedge \Omega$.

Consequently, $e\Phi'_{disease}, e\Phi''_{disease} \in RC(e\Phi_{disease}, \Gamma^2)$. There are included many other KBs in $RC(e\Phi_{disease}, \Gamma^2)$, but all of them would introduce more changes (with regards to the subset relation) than these two (see also Proposition 1), so they are not considered for the sake of simplicity.

To find the $\prec^2_{e\Phi}$-minimal element(s) of $RC(e\Phi_{disease}, \Gamma^2)$, we only need to compare $e\Phi'$ with $e\Phi''$. The weight associated with the sub-preference cost functions is set to $w(T') = T - T' + 1$, so the corresponding costs are: $cost^2_1(\Phi_{disease}, \Phi'_{disease}) = 0$, $cost^2_1(\Phi_{disease}, \Phi''_{disease}) = 3$, so $\Phi'_{disease} \prec^2_{e\Phi} \Phi''_{disease}$ and $\Phi'_{disease}$ is the revision result, as it keeps the narrative of events intact. Notice that we do not need to check for the second sub-preference relation as there is inequality from the first level of preferences.

We present the output of the relevant fluents/events search algorithm, as well as the output of the first reasoning step; namely, the possible worlds, in Figures 5.7 and 5.8. As we have assumed partial initial knowledge, we have an epistemic case and as a result there are two possible worlds. Finally, we present the optimal

```
----------Relevant Fluents/Events----------
Fluent: curedPatient
    Events:proposeCure1
    Events:proposeCure2
    Precondition fluent:liablePatient
    Precondition fluent:curedPatient
    Precondition fluent:disease1
    Precondition fluent:disease2
Fluent: disease1
    Events:proposeCure1
    Precondition fluent:liablePatient
    Precondition fluent:curedPatient
    Precondition fluent:disease1
Fluent: disease2
    Events:proposeCure2
    Precondition fluent:liablePatient
    Precondition fluent:curedPatient
    Precondition fluent:disease2
Fluent: liablePatient
    Events:changeAttitude
    Precondition fluent:liablePatient
----------Relevant Fluents/Events----------
```

Figure 5.7: Medical Assistant Agent: Relevant Fluents/Events Algorithm Output

```
------Possible worlds------
---------World1------------
def(liablePatient) 0
proposeCure1 0
liablePatient 0
liablePatient 1
liablePatient 2
disease2 0
def(liablePatient) 1
disease2 1
disease2 2
def(liablePatient) 2
---------World2------------
def(liablePatient) 0
proposeCure1 0
liablePatient 0
liablePatient 1
liablePatient 2
disease1 0
def(liablePatient) 1
curedPatient 1
curedPatient 2
def(liablePatient) 2
----------------------------
```

Figure 5.8: Medical Assistant Agent Round 1: The output of the first reasoning step; namely, the possible worlds.

revision that accommodates the new observation into our knowledge base in Figure 5.9.

5.3.2.2 Round 2

Consider now that the medical assistant agent has accepted the result of the revision algorithm; namely, the patient is ill, has disease 2 and not disease 1, and is reliable at timepoint 0, as well as he takes the proposed disease cures. Thus, the agent proposes "cure 2" at timepoint 1 and as the initial knowledge is complete there is one possible world as shown in Figure 5.10. Now, the agent may believe that after the suggestion of cure 2, the patient is eventually cured at timepoint 2, as shown in the upper section of Figure 5.11. However, the agent observes that the patient is still ill at timepoint 2. The agent needs to assess different potential revisions of its belief state, but the only optimal and commonsense explanation would be that the patient is probably not reliable and does not take the proposed medication, as neither of the proposed cures worked, as shown in the lower section of Figure 5.11.

Other revision candidates would propose retracting actions from the narrative, that is an expensive notion based on the sub-preference P1 priority among the others, which in this domain expresses the agent's certainty that it actually does proposes cures to the patient, except for extreme cases of failure.

5.3.2.3 An Alternative Round 2

Consider now that the medical assistant agent is a prototype, thus prone to system failures. As a result, we have changed the priorities among the sub-reference relations to represent the aforementioned notion, e.g.,: $\Phi_1 \prec^2_{e\Phi} \Phi_2$ iff there is j=5,2,3,4,1 such that: $cost^2_i(\Phi, \Phi_1) = cost^2_i(\Phi, \Phi_2)$ for $i < j$ and $cost^2_j(\Phi, \Phi_1) < cost^2_j(\Phi, \Phi_2)$. The agent would now need to assess different potential revisions of its belief state: can it be that it was mistaken and did not propose cure 2? We present the optimal revision that accommodates the observation, that the patient is still ill, into the agent's knowledge base in Figure 5.12 (notice the red dash denoting the retracted event occurrence in the lower section of the Figure).

```
------------------------------------------------------
--------------------Current Beliefs--------------------
-----------------------FLUENTS------------------------
Fluent\Timepoint        0       1       2
liablePatient           Bel     Bel     Bel
curedPatient            BelNot  ~BelWh  ~BelWh
disease2                ~BelWh  ~BelWh  ~BelWh
disease1                ~BelWh  BelNot  BelNot
-----------------------EVENTS-------------------------
Event\Timepoint         0       1       2
changeAttitude          -       -       -
proposeCure1            Happ    -       -
proposeCure2            -       -       -
------------------------------------------------------
------------------------------------------------------
--------------------Revised Beliefs--------------------
-----------------Optimal Revision No1-----------------
-----------------------FLUENTS------------------------
Fluent\Timepoint        0       1       2
liablePatient           Bel     Bel     Bel
curedPatient            BelNot  BelNot  BelNot
disease2                Bel     Bel     Bel
disease1                BelNot  BelNot  BelNot
-----------------------EVENTS-------------------------
Event\Timepoint         0       1       2
changeAttitude          -       -       -
proposeCure1            Happ    -       -
proposeCure2            -       -       -
------------------------------------------------------
------------------------------------------------------
```

Figure 5.9: Medical Assistant Agent Round 1: The output of the Java agent, divided in two sections. The upper section in blue represents the original belief state, and the lower section in green represents the revised belief state of the agent.

```
------Possible worlds------
---------World1------------
disease2 0
def(liablePatient) 0
proposeCure1 0
proposeCure2 1
liablePatient 0
liablePatient 1
liablePatient 2
disease2 1
def(liablePatient) 1
curedPatient 2
def(liablePatient) 2
----------------------------
```

Figure 5.10: Medical Assistant Agent Round 2: The output of the first reasoning step; namely, the possible worlds.

```
-----------------------------------------------------
-------------------Current Beliefs-------------------
----------------------FLUENTS------------------------
Fluent\Timepoint        0       1       2
liablePatient           Bel     Bel     Bel
curedPatient            BelNot  BelNot  Bel
disease2                Bel     Bel     BelNot
disease1                BelNot  BelNot  BelNot
----------------------EVENTS-------------------------
Event\Timepoint         0       1       2
proposeCure1            Happ    -       -
proposeCure2            -       Happ    -
-----------------------------------------------------
-----------------------------------------------------
-------------------Revised Beliefs-------------------
----------------Optimal Revision No1-----------------
----------------------FLUENTS------------------------
Fluent\Timepoint        0       1       2
liablePatient           BelNot  BelNot  BelNot
curedPatient            BelNot  BelNot  BelNot
disease2                Bel     Bel     Bel
disease1                BelNot  BelNot  BelNot
----------------------EVENTS-------------------------
Event\Timepoint         0       1       2
proposeCure1            Happ    -       -
proposeCure2            -       Happ    -
-----------------------------------------------------
-----------------------------------------------------
```

Figure 5.11: Medical Assistant Agent Round 2: The output of the Java agent, divided in two sections. The upper section in blue represents the original belief state, and the lower section in green represents the revised belief state of the agent.

```
------------------------------------------------------
-------------------Current Beliefs--------------------
----------------------FLUENTS-------------------------
Fluent\Timepoint        0       1       2
liablePatient           Bel     Bel     Bel
curedPatient            BelNot  BelNot  Bel
disease2                Bel     Bel     BelNot
disease1                BelNot  BelNot  BelNot
----------------------EVENTS--------------------------
Event\Timepoint         0       1       2
proposeCure1            Happ    -       -
proposeCure2            -       Happ    -
------------------------------------------------------
------------------------------------------------------
--------------------Revised Beliefs-------------------
----------------Optimal Revision No1------------------
----------------------FLUENTS-------------------------
Fluent\Timepoint        0       1       2
liablePatient           Bel     Bel     Bel
curedPatient            BelNot  BelNot  BelNot
disease2                Bel     Bel     Bel
disease1                BelNot  BelNot  BelNot
----------------------EVENTS--------------------------
Event\Timepoint         0       1       2
proposeCure1            Happ    -       -
proposeCure2            -       -       -
------------------------------------------------------
------------------------------------------------------
```

Figure 5.12: Medical Assistant Agent Alternative Round 2: The output of the Java agent, divided in two sections. The upper section in blue represents the original belief state, and the lower section in green represents the revised belief state of the agent.

Chapter 6

Conclusion and Future Work

After presenting our work and the corresponding implementation, we now revisit the main outcome and technical contributions from a more abstract standpoint and also remark prominent future research directions.

This thesis reported on a formal framework for changing Event Calculus theories in the face of new (and potentially unexpected) observations. The proposed framework is necessary in the cases where an intelligent agent observes, or otherwise becomes aware of, information that contradicts what was expected by the underlying theory. Even though the rich technical results from the belief change literature are not generally applicable to our setting, we leveraged on some key ideas and adapted them for our purposes. Our approach was based on a set of principles and a preference relation that models the well-known Principle of Minimal Change. We used the standard approach of introducing a *preference relation* $\prec^T_{e\Phi}$. The idea is that if $e\Phi_1 \prec^T_{e\Phi} e\Phi_2$, $e\Phi_1$ is strictly more preferred than $e\Phi_2$ for the result of the revision of $e\Phi$ with an observation at timepoint T. This is different from the relations among interpretations [18] and formulas [10] that have been used elsewhere for the same purpose. Establishing the connection among our preference relation and these works is part of our future work.

Moreover, we are planning on establishing stronger connections with existing results from belief change (e.g., satisfaction of certain postulates, or connections between our preference relation and various selection functions or orderings that have been used in other contexts), thereby more thoroughly understanding the properties of the proposed framework. We are also planning to study the behavior of AGM postulates for EC and non-monotonic logics. Further, even though our theoretical framework is generic enough to support more complex flavours of action theories and $\mathcal{DEC}$, our implementation will need to be significantly extended to support different Event Calculus dialects and a richer set of commonsense features such as non-determinism, state constraints, and introspective belief changes.

The proposed implementation will also be extended with more optimizations for an improved performance (e.g., finding all relations among a KB's fluents and events a priori, decreasing the execution overhead), as well as, for an efficient

navigation in the search space providing quick filter-out of non-optimal solutions. We consider also conducting an experimental evaluation on our framework performance.

Last but not least, in order to support a complete software agent for on-line reasoning that responses to a sequence of observations, we will also need to extend our framework with iterated belief revision formalisms, as discussed in [3].

Bibliography

[1] C. Alchourron, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] M. Dalal. Investigations into a theory of knowledge base revision: Preliminary report. In *AAAI-88*, pages 475–479, 1988.

[3] Adnan Darwiche and Judea Pearl. On the logic of iterated belief revision. In *Proceedings of the 5th conference on Theoretical aspects of reasoning about knowledge*, pages 5–23. Morgan Kaufmann Publishers Inc., 1994.

[4] F. A. D'Asaro, A. Bikakis, L. Dickens, and R. Miller. Foundations for a probabilistic event calculus. In *LPNMR-17*, pages 57–63, 2017.

[5] R. Demolombe and M. P. Pozos-Parra. A simple and tractable extension of situation calculus to epistemic logic. In *ISMIS-00*, pages 515–524, 2000.

[6] Manfred Eigen and Peter Schuster. *The hypercycle: a principle of natural self-organization*. Springer Science & Business Media, 2012.

[7] Ronald Fagin, Joseph Y Halpern, Yoram Moses, and Moshe Vardi. *Reasoning about knowledge*. MIT press, 2004.

[8] P. Ferraris, J. Lee, and V. Lifschitz. Stable models and circumscription. *Artificial Intelligence*, 175(1):236–263, 2011.

[9] G. Flouris, D. Plexousakis, and G. Antoniou. On generalizing the AGM postulates. In *STAIRS-06*, pages 132–143, 2006.

[10] P. Gardenfors and D. Makinson. Revisions of knowledge systems using epistemic entrenchment. In *TARK-88*, pages 83–95, 1988.

[11] Peter Gärdenfors. *Belief revision*, volume 29. Cambridge University Press, 2003.

[12] Martin Gebser, Benjamin Kaufmann, Roland Kaminski, Max Ostrowski, Torsten Schaub, and Marius Schneider. Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):107–124, 2011.

[13] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *ICLP/SLP*, volume 88, pages 1070–1080, 1988.

[14] Periklis Georgiadis, Ioannis Kapantaidakis, Vassilis Christophides, Elhadji Mamadou Nguer, and Nicolas Spyratos. Efficient rewriting algorithms for preference queries. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 1101–1110. IEEE, 2008.

[15] Marc Hanheide, Moritz Göbelbecker, Graham S Horn, Andrzej Pronobis, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Charles Gretton, Richard Dearden, Miroslav Janicek, et al. Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence*, 247:119–150, 2017.

[16] Jaakko Hintikka. Knowledge and belief. 1962.

[17] Andreas G Hofmann and Brian C Williams. Temporally and spatially flexible plan execution for dynamic hybrid systems. *Artificial Intelligence*, 247:266–294, 2017.

[18] H. Katsuno and A.O. Mendelzon. On the difference between updating a knowledge base and revising it. In *KR-91*, 1991.

[19] R.F. Kelly and A.R. Pearce. Complex epistemic modalities in the situation calculus. In *KR-08*, pages 611–620, 2008.

[20] Werner Kießling. Foundations of preferences in database systems. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 311–322. VLDB Endowment, 2002.

[21] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95, 1986.

[22] J. Lee and R. Palla. Reformulating the situation calculus and the event calculus in the general theory of stable models and in answer set programming. *JAIR*, 43(1):571–620, 2012.

[23] Séverin Lemaignan, Raquel Ros, Lorenz Mösenlechner, Rachid Alami, and Michael Beetz. Oro, a knowledge management platform for cognitive architectures in robotics. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3548–3553. IEEE, 2010.

[24] Vladimir Lifschitz. What is answer set programming? In *AAAI*, volume 8, pages 1594–1597, 2008.

[25] Y. Liu and G. Lakemeyer. On first-order definability and computability of progression for local-effect actions and beyond. In *IJCAI-09*, 2009.

[26] J. Ma, R. Miller, L. Morgenstern, and T. Patkos. An Epistemic Event Calculus for ASP-based reasoning about knowledge of the past, present and future. In *LPAR-13*, pages 75–87, 2013.

[27] R. Miller, L. Morgenstern, and T. Patkos. Reasoning about knowledge and action in an epistemic event calculus. In *Commonsense-13*, 2013.

[28] R. Miller and M. Shanahan. Some alternative formulations of the event calculus. *Computational Logic: Logic Programming and Beyond, Essays in Honour of R. Kowalski Part 2*, 2408(1):452–490, 2002.

[29] R. C. Moore. A formal theory of knowledge and action. In *Formal Theories of the Commonsense World*, pages 319–358. J. Hobbs, R. Moore (Eds.), 1985.

[30] L. Morgenstern. Knowledge preconditions for actions and plans. In *IJCAI-87*, 1987.

[31] Boris Motik, Peter F Patel-Schneider, Bijan Parsia, Conrad Bock, Achille Fokoue, Peter Haase, Rinke Hoekstra, Ian Horrocks, Alan Ruttenberg, Uli Sattler, et al. Owl 2 web ontology language: Structural specification and functional-style syntax. *W3C recommendation*, 27(65):159, 2009.

[32] E.T. Mueller. *Commonsense Reasoning: An Event Calculus Based Approach.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2^{nd} edition, 2015.

[33] Madhura Nirkhe and Sarit Kraus. Formal real-time imagination. *Fundamenta Informaticae*, 23(2, 3, 4):371–394, 1995.

[34] T. Patkos and D. Plexousakis. Reasoning with knowledge, action and time in dynamic and uncertain domains. In *IJCAI-09*, 2009.

[35] Pavlos Peppas. Belief revision. *Foundations of Artificial Intelligence*, 3:317–359, 2008.

[36] R. Petrick and H. Levesque. Knowledge equivalence in combined action theories. In *KR-02*, pages 303–314, 2002.

[37] G. Qi and J. Du. Model-based revision operators for terminologies in Description Logics. In *IJCAI-09*, pages 891–897, 2009.

[38] M.M. Ribeiro, R. Wassermann, G. Flouris, and G. Antoniou. Minimal change: Relevance and recovery revisited. 201:59–80, 2013.

[39] Raquel Ros, Séverin Lemaignan, E Akin Sisbot, Rachid Alami, Jasmin Steinwender, Katharina Hamann, and Felix Warneken. Which one? grounding the referent based on efficient human-robot interaction. In *RO-MAN, 2010 IEEE*, pages 570–575. IEEE, 2010.

[40] R. Scherl and H. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144(1-2):1–39, 2003.

[41] R.B. Scherl. Reasoning about the interaction of knowledge, time and concurrent actions in the situation calculus. In *IJCAI-03*, pages 1091–1096, 2003.

[42] C. Schwering, G. Lakemeyer, and M. Pagnucco. Belief revision and progression of knowledge bases in the epistemic situation calculus. In *IJCAI-15*, 2015.

[43] S. Shapiro, M. Pagnucco, Y. Lespérance, and H.J. Levesque. Iterated belief change in the situation calculus. *Artificial Intelligence*, 175(1):165–192, 2011.

[44] T Siméon, J-P Laumond, and F Lamiraux. Move3d: A generic platform for path planning. In *Assembly and Task Planning, 2001, Proceedings of the IEEE International Symposium on*, pages 25–30. IEEE, 2001.

[45] A. Skarlatidis, A. Artikis, J. Filippou, and G. Paliouras. A probabilistic logic programming event calculus. *TPLP*, 15:213–245, 2015.

[46] T. C. Son and C. Baral. Formalizing sensing actions – a transition function based approach. *Artificial Intelligence*, 125(1-2):19–91, 2001.

[47] Moritz Tenorth and Michael Beetz. Knowrobb•"knowledge processing for autonomous personal robots. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 4261–4266. IEEE, 2009.

[48] Moritz Tenorth and Michael Beetz. Knowrob: A knowledge processing infrastructure for cognition-enabled robots. *The International Journal of Robotics Research*, 32(5):566–590, 2013.

[49] Moritz Tenorth and Michael Beetz. Representations for robot knowledge in the knowrob framework. *Artificial Intelligence*, 247:151–169, 2017.

[50] M. Thielscher. Representing the knowledge of a robot. In *KR-00*, pages 109–120, 2000.

[51] N. Tsampanaki, G. Flouris, and T. Patkos. Steps towards commonsense-driven belief revision in the event calculus. In *Proceedings of the Thirteenth International Symposium on Commonsense Reasoning (Commonsense-2017)*. CEUR-WS, 2017.

[52] F. Van Harmelen, V. Lifschitz, and B. Porter. *Handbook of Knowledge Representation*. Elsevier Science, San Diego, USA, 2007.

[53] M. Van Zee, D. Doder, M. Dastani, and L. Van Der Torre. AGM revision of beliefs about action and time. In *IJCAI-15*, pages 3250–3256, 2015.

[54] S. Vassos and H. Levesque. Progression of situation calculus action theories with incomplete information. In *IJCAI-07*, 2007.

[55] Jan Wielemaker, Guus Schreiber, and Bob Wielinga. Prolog-based infrastructure for rdf: Scalability and performance. In *International Semantic Web Conference*, volume 2870, pages 644–658. Springer, 2003.

[56] Jan Wielemaker, Tom Schrijvers, Markus Triska, and Torbjörn Lager. Swi-prolog. *Theory and Practice of Logic Programming*, 12(1-2):67–96, 2012.