Effective Intentions: Reasoning Effectively with the Effects of Behaviours in International Agents

Effective Intentions

Reasoning Effectively with the Effects of Behaviours in Intentional Agents

Timothy William Cleaver - *B. Inf Tech with Hons*,
School of Information and Communication Technology
Science, Environment, Engineering and Technology
Griffith University

Submitted in fulfilment of the requirements of the degree of
Doctor of Philosophy

October 2009

Acknowledgements

First, and foremost, I would like to acknowledge my fiancée Marianne (Manny) Radovan. She has been the foundation upon which I was able to rely. I thank her for the sacrifices she has made to facilitate this work. I owe her a debt of gratitude for the support and encouragement offered throughout this long journey. She was there in my times of despair and in times of success. I am eternally grateful to her and would not have wanted to travel this path with anybody else.

Secondly, I owe my family a great deal. They have been a constant source of encouragement, advice and support. Without their understanding and patience I would not have been able to tackle this great challenge.

Thirdly, I must extend my thanks to my family-in-law, the Radovans. Your support of Manny and I has been unquestioning and unrestricted. For everything you have done for us I thank you.

Fourthly, I note the financial assistance of the Smart Internet Cooperative Research Centre (SIT-CRC). I would also like to recognize the facilities provided by Griffith University in general, and the School of Information and Communication Technology in particular.

Lastly, I extend my gratitude to my supervisors: Professor Abdul Sattar and Professor John Lloyd. Without your encouragement, advice and guidance this work would never have born fruit. I must also express my thanks for the freedom to define my own agenda, the courage to allow me to follow it and the patience to permit me to complete it.

I hope in time to be able to repay this debt to all of you. Until then, however, I have nothing more to offer than these paltry few words and my undying appreciation.

Synopsis

Practical reasoning encapsulates all thought towards behaviour and action. Central to this lies the relationship between decisions towards behaviour and the establishment of the optimal commitment towards the outcomes of these decisions. A core element in these decisions are the effects actions and behaviours cause. Traditionally, study of causation has not been undertaken from the wider perspective of practical reasoning and bounded rationality. It is from this broader view that this thesis progresses. Given agents with commitments towards ongoing behaviours in complex and dynamic environments, roles that causal knowledge pertaining to these commitments play in the wider practical reasoning of such agents are investigated. Three such roles are identified.

The utilization of behavioural knowledge in the maintenance of the validity and the recognition of the success or failure of behaviour is studied. In particular, mechanisms for the resolution of conflicting evidence for the success or failure of behaviours are evaluated. The trade-off between optimal reasoning and optimal behaviour towards monitoring for such evidence is then examined.

The role of such behavioural knowledge in the revision of an agent's beliefs is then investigated. It is argued that the effect of such belief revisions on the commitments held is a more crucial metric in determining optimal revisions than the informational content retained by the options under consideration. The rationality of such revisions from the traditional belief dynamics perspective is then established.

Causal knowledge is then utilized in the consistency maintenance of the commitments an agent makes. This consistency maintenance leads to dependencies between the commitments held. These dependencies are then used to construct techniques for both the quantification of the commitment an agent places in a given intention and mechanisms for the rational propagation of intention revision.

This thesis makes numerous contributions subsequent to the investigations detailed above. A model of actions, effects and maintenance conditions is formalized. The syntax of the AgentSpeak agent-oriented programming language is then extended to integrate an equivalent representation. The meaning of this additional syntax is then made clear by a corresponding extension to the semantics of the AgentSpeak language. This extension is concerned with the maintenance of the validity of the behaviours under execution and the recognition of their success or failure. It is done in such a way as to facilitate customizable conflict resolution mechanisms and monitoring strategies.

The aforementioned formal model of action, validity and effects lies at the heart of a number of new belief revision operators. These operators are defined in a similar style to those of traditional belief revision and their rationality is proved within the AGM framework. It is shown that the operators proposed both maximize the beneficial effect on the agent's commitments while also being rational revisions from the belief dynamics perspective.

By maintaining the consistency of the behaviours adopted by way of their outcome conditions, dependencies between intentions arise. A formal theory of commitment is developed based on these dependencies. Central to this formal theory is the intuition that the commitment an agent should hold in a given intention is proportional to the cost of dropping or revising it. This theory is then utilized to develop a number of intention revision propagation strategies. These strategies balance the need for stability against optimality. It is then demonstrated how this theory can be extended to capture the additional relationships between intentions inherent in the AgentSpeak model.

Statement of Originality

This work has not previously been submitted for a degree or diploma in any university. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Signed: _____ Date: May 26, 2010

Contents

List of Figures

List of Tables

List of Algorithms

List of Publications

- Timothy William Cleaver and Abdul Sattar. "Reasoning about Success and Failure in Intentional Agents". In: *Proceedings of the 2005 Pacific Rim International Workshop on Multi-Agents (PRIMA 2005)*. 2005;
- Timothy William Cleaver, Abdul Sattar, and Kewen Wang. "Reasoning with the Outcomes of Plan Execution in Intentional Agents". In: *Australian Conference on Artificial Intelligence (Aust-AI 2005)*. 2005, pp. 60–69;
- Timothy William Cleaver, Abdul Sattar, and Raihana Ferdous. "User defined monitoring strategies for BDI agent programs". In: *Proceedings of the fifth International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2006)*. ACM Press, 2006, pp. 1055–1057;
- Timothy William Cleaver and Abdul Sattar. "Intention Guided Belief Revision". In: *Proceedings of the twenty second Conference on Artificial Intelligence (AAAI 2007)*. 2007, pp. 36–41; and
- Timothy William Cleaver and Abdul Sattar. "Quantifying Commitment". In: *PRICAI 2008: Trends in Artificial Intelligence: 10th Pacific Rim International Conference on Artificial Intelligence*. Vol. 5351. Lecture Notes in Computer Science. Springer, 2008, pp. 57–69

An apple a day makes 365 apples a year.
Unknown.

1 Introduction

Practical reasoning encompasses the processes that situated intelligent entities undertake in order to behave. It is a difficult problem that is not as well addressed as the other mainstays of artificial intelligence research such as planning, theorem proving or machine learning. Practical reasoning differs from these other disciplines in that it is a not a problem of optimality but sufficing. This is due to the constraints under which practical reasoning must be undertaken. Some of these constraints are imposed by the environment, others are a consequence of the limitations of the intelligence itself and its embodiment, others, still, are imposed by the embedding of the intelligence within its environment.

Environments can range in complexity along a number of axis. From those in which nothing changes aside from the consequences of an intelligent agent's behaviours, to those in which change occurs independent of the behaviours conducted by the aforementioned embedded agent, be this a consequence of multiple other autonomous entities within the environment or non-intelligent processes. Environments may be such that the effects of behaviours are fixed, knowable and instantaneous or where the consequences of actions and behaviours can be uncertain or chaotic and may take indeterminate amounts of time to be realized. The entirety of the environment may be observable and the observations accurate or the observations noisy and partial.

The reasoning and physical capabilities of the embodiment of intelligent agents also places constraints on the outcomes and processes of practical reasoning. The number of actuators and their type influence the actions that an agent can schedule simultaneously or at all. The amount of memory available determines the size, complexity and richness of any representation an agent may construct about the environment and itself. The sensitivity and variety of sensory mechanisms limit the accuracy and scope of any model an agent may wish to assemble. The algorithms with which an agent may construct models of the environment and with which to

draw inferences from them limit the problems that can be posed to the practical reasoning processes and the answers they may provide.

The embedding of an intelligent agent in an environment also constrains its reasoning so as to come to conclusions in a manner timely within the environment. This potentially precludes many traditional planning and theorem proving approaches due to the inherent computational complexity therein. The majority of scenarios concern cases in which a sufficing but suboptimal solution is better, provided it can be arrived at within the time constraints of the environment, than an optimal solution arrived at too late.

As the demands of these constraints increase, so does the sophistication of the practical reasoning required to behave in a satisfactory way. For an agent with a large memory situated in a static, fully and accurately observable environment with deterministic actions and fixed expected behaviours a look-up table of sensory inputs to actions may result in satisfactory behaviour. Once there is uncertainty with respect to the accuracy of inputs, the effects of behaviours or the appropriateness of a given behaviour for a given input, learning techniques such as reinforcement learning may be more effective in mapping sensory data to actions. If the environment is sufficiently complex that stimulus cannot be directly mapped to behaviour, but instead, to desirable states of the world, planning mechanisms to reason from the current state to the desirable future state may be necessary. Similarly if it is necessary not just to react to the environment, but be proactive in forging the future, planning may be a necessary ingredient in practical reasoning.

Once simultaneous desirable futures become an element of an agent's reasoning, it becomes necessary for it to determine the compatibility of the future states it desires. It must do so to prevent devoting its limited resources endeavouring to bring about futures that can never be realized collectively. Once such a subset of futures has been decided upon and resources devoted to their realization, a measure of commitment to these desires is necessary. Without such commitment an intelligent agent will consider itself free to reconsider the futures it will endeavour to bring about. In doing so it may discard significant current and ongoing work towards one set of futures in favour of another. Such reconsideration of the set of futures towards which it will endeavour can be wasteful. The repeated application of costly reasoning procedures to largely similar inputs risks wasting time so as to reproduce prior conclusions. Excessive reconsideration also inhibits cooperation with other intelligent agents. The unpredictability of the behaviour subsequent to constant reconsideration makes coordination difficult. Without predictability, coordinating agents will be unlikely to place much confidence in the success of any planned cooperative behaviour. This necessary commitment and the associated

reasoning processes is collectively called *intention*.

As argued [14], intention is a central component of successful practical reasoning in scenarios involving complex environments. Complex environments are only partially observable, both in accuracy and scope. They are dynamic beyond the outcomes of the behaviours which a reasoner is undertaking and the results and duration of behaviours are uncertain. The management of multiple potentially inconsistent desirable future states is a common necessity when faced with such environments. These properties are common in the types of environments human beings find themselves in on an ongoing basis. They are also the environments in which artificial intelligences have been deployed successfully [59, 70]. It is to this end and within this context that this work is intended to make its contribution.

Underlying much discussion of the concept of intention is the assumption that the known potential outcomes of behaviour are a critical component of the decision process to, both adopt and, maintain a given intention. Interestingly, this idea has not been heavily investigated in the artificial intelligence context and few, if any, *intentional systems* include it. This thesis aims to investigate the consequences of instilling an intelligent agent with knowledge of expected and potential effects of the behaviours it may adopt and the intentions it forms on the basis of these. In particular, three hypotheses are tested:

Hypothesis:
The successful monitoring of the effects and validity of intentional behaviour requires mechanisms to: (1.) resolve the conflict for the success or failure of a behaviour given evidence for both; and (2.) balance the time spent monitoring against other mental tasks.

Hypothesis:
It is rational for an agent to revise its beliefs in accordance with the intended outcomes and constraints of its behaviours.

Hypothesis:
The explicit modelling of the dependencies between intentions that arise as a consequence of consistency maintenance during decision making facilitates both the quantification of commitment and mechanisms for intention reconsideration propagation.

Consequent to the study into the above hypotheses, a number of contributions to the state of the art in a number of topics were produced. Firstly, the relationship between behaviours and both the effects they produce and the constraints required to ensure their validity were formalized. The resultant formalization was motivated

by a number of brief examples. The developed formalisms were then applied to the motivating examples. This demonstrated both the ability of the formality to capture a number of crucial properties of these examples and the reasoning problems arising from its use. Two primary reasoning challenges were identified. Given the representation of both success and failure conditions, a reasoner must be able to progress given situations in which there is simultaneous evidence supporting both the success and failure of a given behaviour. Further, since the process of monitoring incurs a cost, in terms of both time and lost opportunity, it is necessary to balance this against the benefits gained subsequent to monitoring itself. In order to ensure the practicality of the formalisms discussed they were then introduced to the syntax of the AgentSpeak family of programming languages to facilitate the representation and monitoring of the effects of plans. Multiple strategies to overcome both simultaneous evidence for success and failure and to balance monitoring cost against benefit were developed and the semantics underlying the traditional AgentSpeak architecture was then generalized and extended to allow the introduction of these strategies. These results were captured in [21, 23, 22].

Secondly, novel techniques for the revision of an agent's beliefs were proposed. These techniques built upon the representation of behaviours, maintenance conditions and effects developed for the purpose of monitoring. It was observed that the intentions held by an agent are indirectly dependent on the beliefs that support or invalidate them. As such, it was argued that the effect of a given belief revision on the intentions an agent holds is a more important metric for the choice of belief revision than the informational content preserved in making the revision. It was then demonstrated that revisions based upon the effects on intention were rational with respect to the traditional interpretation of rational belief change. These results were made public in [19].

Thirdly, and finally, it was noted that the constraint of intention consistency implied dependencies between intentions as adopted over time. The choice of means to satisfy a current end is partially dependent on the intentions currently held. These dependencies then motivated and were central to the development of both means to quantify the commitment held in a given intention and the rationality of propagating intention change to dependent intentions. Techniques to establish the strength of the dependencies were proposed. These were based on the roles prior intentions play in the decision process to adopt a new intention. The effect of intention-changing operations on such dependencies and the resulting commitments are then discussed. This discussion is then extended to the rationality of revision propagation subsequent to the application of such intention-change operations. A number of approaches, of increasing sophistication, were

then proposed to balance the necessity of intention stability against more optimal behaviour. Through the application of a number of motivating examples the plausibility of the approach was discussed. These contributions were made available in [20].

Prior to detailing the investigations into each of these hypotheses, and the contributions made therein, the relevant background material will be presented. Firstly, a brief introduction of the philosophical foundations of intention, in particular, and practical reasoning, more generally, will be presented. Following this, an overview of the logical formalization of these intuitions will be provided. Subsequent to this, the state of the art of implemented practical reasoning systems is introduced. Particular attention will be paid to AgentSpeak as it provides the chosen platform with which the hypotheses will be evaluated. In conjunction with the presentation of practical reasoning systems, the state of the art of intention dynamics systems will be discussed. With sufficient exposition on the state of intention and practical reasoning in artificial intelligence complete, the history and approaches to the representation and reasoning with actions and their effects will be outlined. A detailed presentation of the dynamics of belief will conclude the exposition of the context from which the contributions of this thesis proceed. The focus then will move onto each hypothesis in turn. Each problem will be motivated, relevant works pertaining to the particular problem discussed and the proposed solution outlined. Subsequent to the elucidation of the hypotheses, an overview of the novel aspects of the implementation with which these ideas are integrated will be presented. To conclude, the primary contributions of this work will be summarized and opportunities for future work suggested.

Man has made his bedlam; let him lie in it.
Fred Allen.

2

Background

Practical reasoning is a difficult problem, one that has seen much work in the fields of philosophy, mathematics and artificial intelligence. So too is the representation and reasoning with actions, behaviours and their effects. Practical reasoning is concerned primarily with the rational management of both action and behaviour from the perspective of an executing, limited and situated agent. Reasoning with action as discussed, on the other hand, is most concerned with hypothetical reasoning within logical formalisations that capture actions and their effects. Both topics contain many subtle issues which must be sufficiently understood as the context which motivates much progress in these fields. In the case of practical reasoning, the philosophy that outlines many of the large ideas will be presented first. This leads to the more detailed work in which these intuitions have been formalized into logical systems. Given the sophistication of these logical models, the approximations required for implementation on limited physical machines will then be outlined. In contrast, the majority of the presentation of actions and their effects will be from the logical and computational perspectives as much of the philosophy is shared with that of practical reasoning.

In conjunction with issues of practical reasoning, it is necessary to present some preliminaries of theoretical reasoning in order to understand its role in behaviour forming thought. In particular, considerations of the dynamics of the informational state will be discussed. This discussion will initially concentrate on sentential belief in which no further structure is available. Discussion will then progress into more expressive domains in which further structure facilitates more detailed analysis.

2.1 Intentions

When discussing intentions, it is useful to stage the discussion in three phases. Initially the philosophical intuitions and theories outline the bounds and motivations for the following discussion. These concepts are then made formal by their translation into rigorous mathematical notation. This facilitates the systematic investigation of the properties and consequences both of the formalization and the intuitions contained therein. Finally, the concessions required to implement these concepts on physical machines are discussed and their adherence to the properties identified as crucial through both mathematical formalization and philosophical debate are outlined.

2.1.1 Theories of Intention

Intention, as used in common language, is a concept with two distinct aspects [14, 13, 15]. The first is in reference to action or behaviour: i.e. to act intentionally, or with an intention. The other is the abstract concept of having an intention as a mental attitude: to have made a decision and adopted the associated mental processes to monitor and base other decisions upon it. These two facets, although related, play different roles in the practical reasoning of an intelligent agent situated in complex environments. The first is concerned with the post facto attribution of responsibility to an agent for its behaviour. It is primarily concerned with the loose connection between an agent's mental state at the time of acting and the behaviour influencing role of intention within that mental state. Such a perspective on intention is of import in a comprehensive theory of intention, but, is of less interest to those in the development of intelligent systems. Intentional action is characterized by actions such that when queried about their driving motivations, the agent can trace the source of its motivation to an end for which the motivation is self-evident[5]. The second aspect, the abstract concept and associated mental processes of intention, on the other hand, does provide guidance on the use of intention within practical reasoning contexts. In particular, rules prescribing the rationality of the formation of a particular intention have been developed, rules defining the rationality of maintaining an intention by way of non-reconsideration have been presented and rules pertaining to the rationality of particular reconsideration processes proposed.

One of the primary assumptions that underlies this philosophical treatment of intention is the assertion that planning is a prerequisite capability for intentional reasoning. This is due to the tight connection between the notion of plan gen-

eration and the specific benefits intention provides. This is manifest through the need of a planning agent to extend the validity of a generated plan from the time of planning through to the completion of its execution or recognition of failure. Without such validation, much of the effort expended in the planning process is wasted due to the need to rebuild the plans as time passes to ensure their applicability. Intention achieves this temporal extension of validity through mechanisms of non-reconsideration.

A second, equally important, and related, assumption pertains to the nature of the plans an intentional reasoner is capable of constructing. For any fixed reasoning system, it is inevitable that it will face problems and environments that are beyond its reasoning capabilities. This is particularly so in the case of practical reasoning due to the temporal constraints under which planning must proceed. A perfect and complete plan to address an opportunity for behaviour, even if possible to construct, is of no use if the opportunity passes before the plan can commence execution. Consequently, it is assumed, that intentional reasoners develop abstract plans that are iteratively refined in parallel to their execution. In order to facilitate this refinement process, commitment to the current partial solution is necessary to ensure its stability as a context for deciding upon a more detailed approach. In this way, the concept of intention is dependent on an agent iteratively constructing and executing plans. An agent that does so is inevitably lead to form intentions and adopt the mental processes that accompany intention for their management.

Intentions are not adopted in isolation. They are one component type of an overall perspective on intelligence based on mentalistic reductionism. The other primary mental attitudes that are relevant to the practical reasoning of an agent include belief, acceptance and desire.

Belief constitutes the facts an intelligent agent holds true. These facts may be concerned with the state of the environment, including the past, present and future. They may reflect on the reasoners own mental state or those of the other intelligent agent(s) with which it cohabits. They may be partial, or permitting of degrees. They aim at truth and consequently are not under the deliberative control of the reasoner.

Acceptance is a context relative adaptation of a reasoner's beliefs to the practical problem currently under deliberation. To accept something as given in a practical reasoning problem is to forgo any doubt about its truth regardless or in spite of the current beliefs towards it. This can be necessary to expedite or facilitate the deliberation within a practical reasoning problem. However, what is accepted in one practical reasoning context need not be accepted in another. So, although, what a reasoner accepts in a particular context is voluntary, it is unconstrained by

the need for consistency across contexts.

Desires, on the other hand, represent the motivational elements that drive the reasoners behaviour at a distance. A desire need not directly map to behaviour, in fact it need be nothing more than a predilection towards or against a particular behaviour or state of the world. Typically, however, desires are hierarchically organized based on higher-order attitudes (desires pertaining to the desirability of other desires for instance). This hierarchy does not imply an ordering of preference or applicability of the desires contained therein. An agent's endorsement or rejection of a desire for use in practical reasoning is dependent on the intentions that govern their desires. For an agent to endorse a desire in its practical reasoning the agent must treat it as a justifying end such that means may be adopted towards it and subsequent reasoning proceeds compatibly with it. Typically, desires are consistent with the values an agent holds and the things it considers good. In many cases, the valuations of an agent are insufficient to uniquely identify the best, may be conflicting, or the agent may have no means by which to compare differing evaluations. Thus, in such cases, the desirability pertaining to a given behaviour must go beyond the agent's evaluation of the good. Regardless of the constraints regarding, or the evaluation of, its desires, when deliberating over a practical reasoning problem, the conclusion a reasoner draws in terms of the intentions it adopts must be compatible with its beliefs, desires and existing intentions.

There are a pair of principles pertaining to the rationality of holding a given intention at a particular time: the *historical principle of deliberative rationality* and the *historical principle of non-deliberative rationality*.

(2.1.1) Principle (Historical Principle of Deliberative Rationality):
If [an agent] S at time t_1 forms the intention to [do action] A at time t_2 on the basis of deliberation at time t_1, then it is rational of S at t_1 to intend to A at t_2 if and only if:

1' for those intentions of S's that play a direct role as a background of S's deliberation, it is rational of S at t_1 so to intend*; and*

2 S reasonably supposes that A is at least as well supported by his reasons for actions as its relevant, admissible alternatives.

(Bratman [14, page 85])

(2.1.2) Principle (Historical Principle of Non-Deliberative Rationality):
In the basic case it is rational of [an agent] S at time t_1 to intend [to do action] A at time t_2 just in case:

a it was rational of S at t_0 to form this intention; and

b it was rational of S from t_0 to t_1 not to reconsider this intention.

(Bratman [14, page 80])

It is through adherence the *historical principle of deliberative rationality* that the rationality of newly adopted and reconsidered intentions are assured. Adhesion to the *historical principle of non-deliberative rationality* then extends this established rationality forward through time.

It is rational for a reasoner to refrain from reconsidering its intentions by default. Unless otherwise externally pressured through changes to the beliefs, desires or acceptances pertaining to that intention, the rationality attributable to a given intention is persistent. This temporal projection of rationality sans deliberation is referred to as *non-reflective reason-preserving non-reconsideration*. It is a direct consequence of the deliberation in adopting an intention that it will be stable unless otherwise perturbed. Non-reflective non-reconsideration differs from the reflective case in that in the latter the reasoner deliberates over the problem of whether to reconsider its intention and decides not to, whereas, in non-reflective non-reconsideration there is an absence of deliberation entirely. Reason preserving non-reconsideration refers to instances of reflective deliberation that ultimately result in the maintenance of the intention in question, provided pertinent mental attitudes offer the same basis for its maintenance as for its adoption. Conversely, non-reason-preserving non-reconsideration is a deliberation over whether to reconsider a given intention such that reconsideration is discarded but the motivations for not reconsidering are different from the motivations behind the original adoption and subsequent maintenance of the given intention.

A secondary, and equally important, advantage of the adoption of intentions are their role, not only in posing problems for a practical reasoning system to overcome, but in constraining the problem space in which it must search for their solution. This is a direct consequence of the need for the intentions a reasoner concurrently adopts to be internally consistent. An intelligent agent with inconsistent intentions opens itself to be guilty of undermining its own behaviour. Thus, when faced with a practical reasoning problem, the reasoner must exclude from its deliberation any option that will conflict with its current intentions. Of course, an intention in conflict with the options available to a given problem itself may prove a sufficient obstacle to identifying a solution that the reasoner has to reconsider this intention to facilitate a solution to its current issue. In general, however, it is

assumed that this is a rarity and consistent options will be available thus facilitating the rational non-reconsideration of the reasoners existing intentions. A consequence of the consistency condition on intentions is that they are agglomerative: *"If at one and the same time I rationally intend to A and rationally intend to B then should be both possible and rational for me, at the same time, to intend A* and *B"* Bratman [13, page 220].

Provided a potential solution is compatible with the reasoners ongoing intentions, it need also be consistent with the beliefs a reasoner accepts as true within the context of the current problem before accepting the option as a complete solution and thus intention. For a particular option to be compatible with a reasoners beliefs, the reasoner must accept that the proposed solution is possible to both execute and refine successfully throughout its execution. It must also believe that it is at least as optimal solution as any other solution it has available to it given what it accepts as true. Note that this need not mean that the solution is globally optimal, as it may be beyond the reasoners capabilities or the time constraints under which it must reason to construct such a solution. Additionally, the reasoners pre-existing intentions may preclude such a globally optimal solution. The reasoner must also not believe that when the time comes for this solution to be brought to bear that it will not do so. It may be ambivalent about whether it may so act or not but it may not believe explicitly that it will not so act[1]. It need not, additionally, believe conclusively that the solution will solve the problem. Only that the solution may do so.

As intentions must be compatible with the reasoners beliefs, so must they be with its desires. The imposition of such a constraint is significantly less however. For this to be satisfied it is sufficient that the option under consideration be in the furtherance of one or more of the reasoners desires. Because desires have less constraints on their internal compatibility, the relationships between desires and other mental attitudes must be as forgiving. Thus an intention can be rationally held in spite of being incompatible with one or many of the reasoners desires. The reasoner may, in fact, hold a desire that is in direct conflict with one or more of the desires that the current option is under consideration for.

Given the established rationality of holding a particular intention, a mechanism for the transfer of this rationality to behaviour at the time of action is necessary. The currently accepted mechanism for this is the *intention-action principle*.

(2.1.3) Principle (Intention-Action Principle):
If it is rational of [an agent] S to have a present-directed intention to [do action]

[1] This is referred to as the *Asymmetry Thesis* by Bratman [14].

A, and S successfully executes this intention and thereby intentionally A's, then it is rational of S to A. *(Bratman [14, page 55])*

It is rational of a reasoner to maintain a present-directed intention towards action provided it satisfies the *no-regret condition*. This condition makes explicit the reasoners autonomy at the time of action and the defeasibility of the intentions that motivate it.

(2.1.1) Condition (No-Regret Condition[2]):
If at time t_2 an agent S is poised to execute an intentional action A based on an intention to A adopted at time t_1 prior to t_2, then it is rational for S to retain its intention towards Aing at t_2 provided that S projects that at future time t_3 that she will not prefer to have intended any alternatives to A that she considered applicable at time t_2 Bratman [13, pages 59–90].

By considering the projected mental state of the reasoner at the close of the intention in which the present-directed intention is embedded, the practical reasoner can ensure the rationality of its present-directed intention prior to accepting its consequences.

A related principle regarding the projection into the future of the rationality of forming a given intention to the point at which this intention bears itself upon the agent's behaviours is known as the *linking principle*.

(2.1.4) Principle (Linking Principle):
If, on the basis of deliberation, an agent rationally settles at time t_1 on an intention to [do action] A at time t_2 if (given that) [accepted context] C, and if she expects that under C at t_2 she will have rational control of whether or not she A's, then she will not *suppose [that] at t_1 that if C at t_2 she should, rationally, abandon her intention in favour of an intention to perform an alternative to A.* *(Bratman [13, page 64])*

One must distinguish the case between deliberating over a particular intention and the intentional act of reconsidering it. For one to truly reconsider an intention, a reasoner must retract the intention in question from their mental state and be prepared to accept options in conflict with it as admissible. Deliberations that reinforce the intention, involve changes to the belief or desire reasons for the intentions retention, or do not consider currently inadmissible options cannot be said to be true instances of reconsideration.

[2] t_3 is the completion time of the plan in which her intention to *A* is embedded.

Behaviours can, and often will, result in a multitude of effects once executed. It is an important aspect of practical rationality that one be able to intend the execution of an action or behaviour without the need to also intend all the expected consequences considered throughout the deliberation to adopt the intention towards such execution. Typically one or more of these effects will be the intended outcomes of such execution, but all effects need not be. This remains true even in scenarios in which it is advantageous to simply ignore some of the known outcomes of an action throughout the deliberation process. Of those considered, an agent need not be irrational for not intending all the consequences of its actions. This is important in light of the resource constraints of an agent, as the requirement to intend all the consequences of a given action would require the functional roles of intention to apply to these consequences also. Thus an agent would need to endeavour to ensure the satisfaction of these consequences and reason with this in mind. This problem has been termed, both, the problem of the *package deal* and the *side-effect problem*.

Intentions, in their most general form, can range over the validity of other mental attitudes in given practical reasoning problems. This has been discussed, primarily, in the context of autonomy, self governance and desires. In these scenarios, intentions are formed that police the use of desires as justified ends towards which the agent is prepared to work. Though intention extends the will of a reasoner into the future, the defeasibility of such intentions allows the reasoner to identify with the desires such policies endorse without being beholden to them. In order to avoid infinite regresses in assigning authority to these intentions, they are structured such that they self referentially endorse their own authority as a controller of the agent's reasoning and behaviour.

Intentions offer more than just problems for practical reasoners to solve. Intentions can, in many scenarios, provide solutions to the practical reasoning problems that intentions can pose. One such instance in which this is the case arises when a reasoner is presented with a new opportunity for action and the intentions it is currently committed to can be modified efficiently to cater for this new opportunity. By turning to its existing commitments, the reasoner avoids generating alternatives, deliberating over these and the overhead of managing an additional independent intention. In such cases the reasoner can be said to have *overloaded* its intention [75]. Although this offers an optimal use of existing mental structures and thus a rational reasoning strategy, it does not offer optimality of the resulting behaviour. Because of this, complete means-ends reasoning and deliberation is more appropriate for *important* practical reasoning problems. In particular, practical reasoning problems where a the reasoner must aim for optimality of outcome and not opti-

mality of reasoning. Unfortunately, it remains unclear as to which decisions can be considered *important* in any given situation.

2.1.2 Logics of Intention

The logics that claim to formalize the concept of intention are overwhelmingly based on modal or multi-modal intuitions. A modal logic is a logic such that one or more modes of truth are distinguished[18, 56]. For instance, a proposition or any well formed formula may be believed true, always true, true at a point in time, or necessarily true. The modes and modalities of modal logics facilitate such distinctions. The mathematical machinery behind the semantics of such logics allow for the study of formal properties that constitute such modes of truth in a rigorous way and facilitate the capture of the intuitions behind such perspectives unambiguously. The semantics of modal logics are typically interpreted via Kripke structures, named after their discoverer, or as also known, possible world semantics. The idea being that there are a number of worlds (consistent descriptions of the truth of propositions). Each world can value propositions or other well formed formula differently. Worlds are related such that the truth of a formula under a given modality at a given world is defined by the worlds to which it is related via that modality. Each world related to a given world is considered accessible from that world. Traditionally, if a property (ϕ) holds of all worlds accessible from the "current" world, provided there are any, then the property necessarily ($\Box\ \phi$) [3] holds at the "current" world. If a property holds at one or more accessible worlds from the "current" world, then the property is considered possible ($\Diamond\ \phi$)[4] at the "current" world. If a property is necessary at a given world, then by definition it is also possible at that same world[5]. Thus, the truth values of propositions qualified by a modality are dependent on the worlds accessible from the world in question and the truth value of the proposition at the accessible worlds.

The attribution of meaning using possible worlds semantics gains its popularity from a number of mathematically elegant properties. Many of these properties relate to *correspondence theory*[105]. Correspondence theory studies the relationship between the accessibility relation between worlds and the axioms that hold for modalities whose semantics are defined by those relations. Table 2.1 provides the equivalences between the properties of the accessibility relations and the ax-

[3]The actual interpretation is defined by the modal operator itself. Reference is made to necessarily and possibly as these were the original modes of truth studied in early modal logic.

[4]Note, also that $(\Box\ \phi) \equiv \neg\,(\Diamond\ \neg\phi)$.

[5]Provided there exists at least one world which is related to the given world.

iom they necessitate[6]. In this table R represents the accessibility relation between worlds. Note, the axiom **K** holds for all *normal*[7] modal logics.

Name	Axiom	Property
K	$(\Box\,\phi \rightarrow \psi) \rightarrow ((\Box\,\phi) \rightarrow (\Box\,\psi))$	—
T	$(\Box\,\phi) \rightarrow \phi$	R is reflexive
B	$\phi \rightarrow (\Box\ (\Diamond\ \phi))$	R is symmetric
4	$(\Box\,\phi) \rightarrow (\Box\ (\Box\,\phi))$	R is transitive
5	$(\Diamond\,\phi) \rightarrow (\Box\ (\Diamond\,\phi))$	R is Euclidean
D	$(\Box\,\phi) \rightarrow (\Diamond\,\phi)$	R is serial
trivial	$\phi \rightarrow (\Box\,\phi)$	Reflexive dead-end

Table 2.1: Correspondences between Properties of Relations and Validity of Axioms (adapted from [18]).

The prototypical example of the use of multi-modal logics in the specification of agent state and reasoning is that of BDI logic[80, 84, 79, 78, 85, 83, 81, 82], although alternatives exist [67, 24, 25, 32, 97]. BDI logic recognizes the primacy of belief, desire and intention in successful practical reasoning. Each mental attitude is formalized as a normal modal logic operator such that $(\mathsf{Bel}\ \phi)$, $(\mathsf{Des}\ \phi)$ and $(\mathsf{Int}\ \phi)$[8] intuitively represents the fact the agent believes, desires or intends ϕ respectively. Each modality is associated with its own accessibility relation over the set of possible worlds. The worlds accessible from a given world via the belief accessibility relation define the worlds the agent believes possible at this world. Similarly, worlds accessible via the desire and intention accessibility relations are those that the agent desires possible and intends possible, respectively, given this world. Thus, if a proposition holds at all the worlds accessible via the agent's belief, desire or intention accessibility relation from the "current" world then the agent believes, desires or intends the proposition in the "current" world.

[6] This table is by no means comprehensive as there is extensive literature advocating many additional properties of the accessibility relation for a wide variety of interpretations. Those presented here were chosen as they will be pertinent in later discussion.

[7] Non-normal modal logics are those that do not permit **K** as an axiom. This is achieved semantically through worlds at which everything is "possible" including false ($\perp$) and nothing is "necessary".

[8] This notation comes from [110]. It emphasizes the semantic difference between predicates and modalities by utilizing distinctly different syntactic styles for each.

The accessibility relation between worlds that defines the belief modality is serial, transitive and euclidean. This relation will be denoted as B. This corresponds to the well studied modal logic **KD45**, so called because it satisfies the axioms **K**, **D**, **4**, **5** in table 2.1. The intuition behind these axioms as appropriate for belief is that, in the case of **K**, if an agent believes a well formed formula ϕ implies another ψ then when it believes ϕ it also believes ψ. In the case of **D**, if the agent believes a well formed formula then it also does not believe that the formula does not hold. This forces the agent's beliefs to be consistent. In the case of **4**, when an agent believes in the truth of a given well formed formula, it also believes that it believes this. This means that an agent is reflective on what it believes true. Similarly, in the case of **5**, if an agent does not believe the falsity of a well formed formula, then it believes that it does not believes it. This provides the agent with reasoning powers about what it does not believe as well as what it does. Because the rules of inference of normal modal logic include the rule of necessitation (if ϕ is a theorem, so is $(\Box\ \phi)$ for all normal modalities) in addition to uniform substitution and modus ponens, an agent believes all theorems of its belief logic. This is traditionally known as the problem of logical omniscience and will be discussed more fully in section 2.1.2.

In contrast to belief, the accessibility relations for desire and intention are less restrictive, requiring only that every world be related to at least one other world. These relations will be denoted D and I respectively. This leads to the axioms **K** and **D** holding for these modalities. Consequently, the only restriction on the desires[9] and intentions of the agent is that they are internally consistent. Thus, an agent cannot desire or intend a well formed formula while simultaneously desiring or intending the negation of this formula. Like belief, because these modalities satisfy **K**, the rule of inference necessitation applies and consequently an agent desires and intends all theorems of its desire and intention logics.

In essence, BDI is not a single logic, but a framework for a family of logics depending on the interaction axioms chosen between modalities. These interaction axioms define the truth of a given formula under a particular modality when its truth under another has been established. Such axioms constitute the syntactic representation of a semantic relationship between the accessibility relations of the modalities in question. In particular, one accessibility relation can relate a given world to a subset ($\subseteq$), the same set or super set of the worlds that another accessibility relation does. Alternatively, it can be required that the intersection of the

[9]Desires, as traditionally interpreted, need not be internally consistent. Thus, in this context, the notion of desires actually more accurately reflects the concept of goals which do require internal consistency.

accessibility relations be non empty. All of the pair-wise interaction axioms are presented in table 2.2 are generated by the application of the subset relationship to each combination of accessibility relations. Some of these interaction axioms are absurd given the interpretation of beliefs, desires and intentions we wish to adopt for such logics. For instance, the axiom $(\mathsf{Des}\ \phi) \rightarrow (\mathsf{Int}\ \phi)$ can be interpreted to mean that the agent is to intend all of its desires. In the form of practical reasoning and resource limitations envisaged for the agents to which BDI is applicable, this axiom is in direct contrast to these restrictions and subsequently seems an unfit characterization of their reasoning. Of the axioms presented in table 2.2, only $(\mathsf{Int}\ \phi) \rightarrow (\mathsf{Des}\ \phi)$ seems to capture a valid property of the mental state of agents engaged in practical reasoning. Particularly, it captures the fact that whatever an agent intends must also be desired.

Relationship	**Axiom**
$\mathtt{D} \subseteq \mathtt{I}$	$(\mathsf{Int}\ \phi) \rightarrow (\mathsf{Des}\ \phi)$
$\mathtt{I} \subseteq \mathtt{D}$	$(\mathsf{Des}\ \phi) \rightarrow (\mathsf{Int}\ \phi)$
$\mathtt{D} \subseteq \mathtt{B}$	$(\mathsf{Bel}\ \phi) \rightarrow (\mathsf{Des}\ \phi)$
$\mathtt{B} \subseteq \mathtt{D}$	$(\mathsf{Des}\ \phi) \rightarrow (\mathsf{Bel}\ \phi)$
$\mathtt{B} \subseteq \mathtt{I}$	$(\mathsf{Int}\ \phi) \rightarrow (\mathsf{Bel}\ \phi)$
$\mathtt{I} \subseteq \mathtt{B}$	$(\mathsf{Bel}\ \phi) \rightarrow (\mathsf{Int}\ \phi)$
$\mathtt{D} \cap \mathtt{I} \neq \varnothing$	$(\mathsf{Int}\ \phi) \rightarrow \neg(\mathsf{Des}\ \neg\phi)$
$\mathtt{I} \cap \mathtt{D} \neq \varnothing$	$(\mathsf{Des}\ \phi) \rightarrow \neg(\mathsf{Int}\ \neg\phi)$
$\mathtt{D} \cap \mathtt{B} \neq \varnothing$	$(\mathsf{Bel}\ \phi) \rightarrow \neg(\mathsf{Des}\ \neg\phi)$
$\mathtt{B} \cap \mathtt{D} \neq \varnothing$	$(\mathsf{Des}\ \phi) \rightarrow \neg(\mathsf{Bel}\ \neg\phi)$
$\mathtt{I} \cap \mathtt{B} \neq \varnothing$	$(\mathsf{Bel}\ \phi) \rightarrow \neg(\mathsf{Int}\ \neg\phi)$
$\mathtt{B} \cap \mathtt{I} \neq \varnothing$	$(\mathsf{Int}\ \phi) \rightarrow \neg(\mathsf{Bel}\ \neg\phi)$

Table 2.2: Relationship between accessibility relations and the axioms they entail (adapted from [82, 110]).

BDI logic is not, however, restricted to representing worlds as sets of propositions. As originally formulated, BDI logic represents worlds as branching time structures. The past is represented as a single linear sequence of states and the associated actions that transitioned one state to the next. The future, in contrast, is represented as a tree of possible futures. Each path represents a possible future evolution of the world. Each transition is labelled with an action that the agent may execute. Multiple edges emanating from a single node may be labelled with

the same action but lead to different resulting states. This captures the intuition that the environment can and will evolve contrary to the agent's actions (which may fail). Formally, the logic that captures such branching time tree structures is CTL*[37]. In addition to operators for specifying truth at the next state ($\circ\,\phi$), at some future state ($\Diamond\,\phi$), at all future states ($\Box\,\phi$), and until a condition holds another remains true ($\phi\,\mathcal{U}\,\psi$), CTL* facilitates quantification over all paths $\mathsf{A}\phi$ and the existence of paths $\mathsf{E}\phi$.

Relationship	**Axiom**
$\mathtt{D} \subseteq_{sub} \mathtt{I}$	$(\mathsf{Int}\ \mathsf{A}\phi) \rightarrow (\mathsf{Des}\ \mathsf{A}\phi)$
$\mathtt{I} \subseteq_{sub} \mathtt{D}$	$(\mathsf{Des}\ \mathsf{A}\phi) \rightarrow (\mathsf{Int}\ \mathsf{A}\phi)$
$\mathtt{D} \subseteq_{sub} \mathtt{B}$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow (\mathsf{Des}\ \mathsf{A}\phi)$
$\mathtt{B} \subseteq_{sub} \mathtt{D}$	$(\mathsf{Des}\ \mathsf{A}\phi) \rightarrow (\mathsf{Bel}\ \mathsf{A}\phi)$
$\mathtt{B} \subseteq_{sub} \mathtt{I}$	$(\mathsf{Int}\ \mathsf{A}\phi) \rightarrow (\mathsf{Bel}\ \mathsf{A}\phi)$
$\mathtt{I} \subseteq_{sub} \mathtt{B}$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow (\mathsf{Int}\ \mathsf{A}\phi)$
$\mathtt{D} \cap_{sub} \mathtt{I} \neq \varnothing$	$(\mathsf{Des}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{E}\phi)$
$\mathtt{I} \cap_{sub} \mathtt{D} \neq \varnothing$	$(\mathsf{Int}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\mathsf{E}\phi)$
$\mathtt{D} \cap_{sub} \mathtt{B} \neq \varnothing$	$(\mathsf{Des}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Bel}\ \neg\mathsf{E}\phi)$
$\mathtt{B} \cap_{sub} \mathtt{D} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\mathsf{E}\phi)$
$\mathtt{I} \cap_{sub} \mathtt{B} \neq \varnothing$	$(\mathsf{Int}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Bel}\ \neg\mathsf{E}\phi)$
$\mathtt{B} \cap_{sub} \mathtt{I} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{E}\phi)$

Table 2.3: Relationship between sub-world constrained accessibility relations and the axioms they entail (adapted from [82, 110]).

Given this commitment towards the use of CTL* as the basis of the worlds over which the belief, desire and intention accessibility relations range, a more fine grained presentation and analysis of the interaction axioms of table 2.2 can be conducted. As a preliminary to doing so, it is necessary to refine the relationships that can hold between accessibility relations. This can be achieved by imposing additional conditions on the worlds that the accessibility relations range over. In particular, it can be demanded that one world can be a sub-world (*sub*) of another. One world is a sub-world of another provided all the paths in the former are contained in the later, although, the later may contain paths additional to those in the former. Thus, it can be required that, in addition to the requirements outlined in table 2.2, all worlds related by one accessibility relation be sub-worlds of those related by the other. Table 2.3 outlines the axioms following from requiring a sub-world

Relationship	Axiom
$\mathtt{D} \subseteq_{sup} \mathtt{I}$	$(\mathsf{Int}\ \mathsf{E}\phi) \rightarrow (\mathsf{Des}\ \mathsf{E}\phi)$
$\mathtt{I} \subseteq_{sup} \mathtt{D}$	$(\mathsf{Des}\ \mathsf{E}\phi) \rightarrow (\mathsf{Int}\ \mathsf{E}\phi)$
$\mathtt{D} \subseteq_{sup} \mathtt{B}$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow (\mathsf{Des}\ \mathsf{E}\phi)$
$\mathtt{B} \subseteq_{sup} \mathtt{D}$	$(\mathsf{Des}\ \mathsf{E}\phi) \rightarrow (\mathsf{Bel}\ \mathsf{E}\phi)$
$\mathtt{B} \subseteq_{sup} \mathtt{I}$	$(\mathsf{Int}\ \mathsf{E}\phi) \rightarrow (\mathsf{Bel}\ \mathsf{E}\phi)$
$\mathtt{I} \subseteq_{sup} \mathtt{B}$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow (\mathsf{Int}\ \mathsf{E}\phi)$
$\mathtt{D} \cap_{sup} \mathtt{I} \neq \varnothing$	$(\mathsf{Des}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{A}\phi)$
$\mathtt{I} \cap_{sup} \mathtt{D} \neq \varnothing$	$(\mathsf{Int}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\mathsf{A}\phi)$
$\mathtt{D} \cap_{sup} \mathtt{B} \neq \varnothing$	$(\mathsf{Des}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Bel}\ \neg\mathsf{A}\phi)$
$\mathtt{B} \cap_{sup} \mathtt{D} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\mathsf{A}\phi)$
$\mathtt{I} \cap_{sup} \mathtt{B} \neq \varnothing$	$(\mathsf{Int}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Bel}\ \neg\mathsf{A}\phi)$
$\mathtt{B} \cap_{sup} \mathtt{I} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{A}\phi)$

Table 2.4: Relationship between super-world constrained accessibility relations and the axioms they entail (adapted from [82, 110]).

relation in addition to those of table 2.2. Table 2.4 outlines the axioms following from requiring a super-world (sup) relation in addition to those of table 2.2. This allows for the discussion of the rationality of interactions between beliefs, desires and intentions that refer both to future options and inevitability.

The analysis of the interactions between modalities can extend beyond the discussion of the pair-wise relationships above. Ternary relations between all three modalities can be enforced. Although many systems of logic are resultant from different selections of ternary relations, only a small number are consistent with the essential properties of practical reasoning presented thus far. These conforming systems are presented in table 2.5. Of these systems, the most appropriate system is typically dependent on the environment in which the agent is to be situated and the task for which it is envisaged.

As discussed in section 2.1.1, an agent need not intend all the necessary consequences of its actions that it utilizes in deciding on the means for an end. This can be captured in a number of the systems described in table 2.5 but not all. To do so, the following formulæ must be satisfiable:

$$(\mathsf{Int}\ \phi) \wedge (\mathsf{Bel}\ \phi \rightarrow \psi) \wedge \neg\,(\mathsf{Int}\ \psi) \tag{2.1}$$

Name	Condition(s)	Formula Schema
BDI – $S1$	$\mathtt{B} \subseteq_{sup} \mathtt{D} \subseteq_{sup} \mathtt{I}$	$(\mathsf{Int}\ \mathsf{E}\phi) \rightarrow (\mathsf{Des}\ \mathsf{E}\phi) \rightarrow (\mathsf{Bel}\ \mathsf{E}\phi)$
BDI – $S2$	$\mathtt{B} \subseteq_{sub} \mathtt{D} \subseteq_{sub} \mathtt{I}$	$(\mathsf{Int}\ \mathsf{A}\phi) \rightarrow (\mathsf{Des}\ \mathsf{A}\phi) \rightarrow (\mathsf{Bel}\ \mathsf{A}\phi)$
BDI – $S3$	$\mathtt{B} \subseteq \mathtt{D} \subseteq \mathtt{I}$	$(\mathsf{Int}\ \phi) \rightarrow (\mathsf{Des}\ \phi) \rightarrow (\mathsf{Bel}\ \phi)$
BDI – $R1$	$\mathtt{I} \subseteq_{sup} \mathtt{D} \subseteq_{sup} \mathtt{B}$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow (\mathsf{Des}\ \mathsf{E}\phi) \rightarrow (\mathsf{Int}\ \mathsf{E}\phi)$
BDI – $R2$	$\mathtt{I} \subseteq_{sub} \mathtt{D} \subseteq_{sub} \mathtt{B}$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow (\mathsf{Des}\ \mathsf{A}\phi) \rightarrow (\mathsf{Int}\ \mathsf{A}\phi)$
BDI – $R3$	$\mathtt{I} \subseteq \mathtt{D} \subseteq \mathtt{B}$	$(\mathsf{Bel}\ \phi) \rightarrow (\mathsf{Des}\ \phi) \rightarrow (\mathsf{Int}\ \phi)$
BDI – $W1$	$\mathtt{B} \cap_{sup} \mathtt{D} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\mathsf{A}\phi)$
	$\mathtt{D} \cap_{sup} \mathtt{I} \neq \varnothing$	$(\mathsf{Des}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{A}\phi)$
	$\mathtt{B} \cap_{sup} \mathtt{I} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{A}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{A}\phi)$
BDI – $W2$	$\mathtt{B} \cap_{sub} \mathtt{D} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\mathsf{E}\phi)$
	$\mathtt{D} \cap_{sub} \mathtt{I} \neq \varnothing$	$(\mathsf{Des}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{E}\phi)$
	$\mathtt{B} \cap_{sub} \mathtt{I} \neq \varnothing$	$(\mathsf{Bel}\ \mathsf{E}\phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\mathsf{E}\phi)$
BDI – $W3$	$\mathtt{B} \cap \mathtt{D} \neq \varnothing$	$(\mathsf{Bel}\ \phi) \rightarrow \neg\,(\mathsf{Des}\ \neg\phi)$
	$\mathtt{D} \cap \mathtt{I} \neq \varnothing$	$(\mathsf{Des}\ \phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\phi)$
	$\mathtt{B} \cap \mathtt{I} \neq \varnothing$	$(\mathsf{Bel}\ \phi) \rightarrow \neg\,(\mathsf{Int}\ \neg\phi)$

Table 2.5: Systems of BDI logic (adapted from [82, 110]).

$$(\mathsf{Int}\ \phi) \wedge (\mathsf{Des}\ \phi \rightarrow \psi) \wedge \neg\,(\mathsf{Int}\ \psi) \tag{2.2}$$

$$(\mathsf{Des}\ \phi) \wedge (\mathsf{Bel}\ \phi \rightarrow \psi) \wedge \neg\,(\mathsf{Des}\ \psi) \tag{2.3}$$

However, the above formulæ are not satisfiable in the system BDI-R3.
In order to capture the *asymmetry thesis* it is necessary that:

$$(\mathsf{Int}\ \phi) \wedge (\mathsf{Bel}\ \neg\phi) \tag{2.4}$$

be unsatisfiable and:

$$(\mathsf{Int}\ \phi) \wedge \neg\,(\mathsf{Bel}\ \phi) \tag{2.5}$$

be satisfiable in the logic. This ensures that it is irrational for an agent to intend ϕ while believing that it will not achieve ϕ. Conversely, it guarantees that intending ϕ while not believing that one will achieve ϕ is rational. Combined, these requirements prevent an agent from pursuing impossibilities while still allowing progress towards goals without needing a guarantee of success. endequation

$$(\mathsf{Bel}\ \phi) \wedge \neg\,(\mathsf{Int}\ \phi) \tag{2.6}$$

$$(\mathsf{Des}\ \phi) \wedge \neg\,(\mathsf{Int}\ \phi) \tag{2.7}$$

capture another important requirement of rationality called the *non-transference principle*. Provided the above formulæ are satisfiable in the logic, this ensures that the agent's desires are not determined by its beliefs and its intentions are not determined by its beliefs nor its desires. This allows an agent to avoid situations in which every belief must be adopted as a desire or intention and all desires adopted as intentions.

In addition to reflective reasoning on beliefs, which follows from the axioms **4** and **5**, it can be required that an agent believes that it intends or desires a given formula whenever it in fact does so. This form of reflectivity is captured by the following axioms [87]:

$$(\mathsf{Int}\ \phi) \rightarrow (\mathsf{Bel}\ \ (\mathsf{Int}\ \phi)) \tag{2.8}$$

$$(\mathsf{Des}\ \phi) \rightarrow (\mathsf{Bel}\ \ (\mathsf{Des}\ \phi)) \tag{2.9}$$

It is more typically captured by equivalences instead of implication. The intuition that an agent only has intentions for which it has desires can be captured by the following axiom:

$$(\mathsf{Int}\ \phi) \rightarrow (\mathsf{Des}\ \ (\mathsf{Int}\ \phi)) \tag{2.10}$$

Another property that can be captured in BDI logic is that of *no infinite deferral*. Provided the following:

$$(\mathsf{Int}\ \phi) \rightarrow \mathsf{A}\,(\Box\,\neg(\mathsf{Int}\ \phi)) \tag{2.11}$$

is satisfiable, an agent cannot maintain an intention indefinitely. If the environment does not evolve in such a way as to force the agent to drop an intention, via the intended formula becoming believed impossible or already true, this forces the agent to reason towards the eventual achievement of the intention.

One facet of rational thought towards action and behaviour that is lacking from the above presentation is that of probabilities and pay-offs, both core components of decision theoretic approaches to rationality. This limitation was overcome in [86] in which both were introduced into the above logic. Probability was introduced, semantically, through the addition of functions that assigned a probability (real value) to each time point in each world and, syntactically, via a probability operator. Similarly, pay-offs were introduced, semantically, by the introduction of pay-off function that for each world and time point provides a partial mapping

from paths to real values and, syntactically, through the introduction of a pay-off operator. Because uncertainty is represented via the multitude of possible worlds, each belief accessible world was required to have identical probability distributions, and each of the probability distributions sum to one. The probabilities and path pay-offs in desire accessible worlds are generated from the decision tree representation by a simple transformation function. Traditional decision theoretic decision procedures such as *maximin* and *maxexpvalue* were then applicable to deliberate over the desires in the generation of intention. In the case of *maximin* the intentions were unconditional. Conversely, the intentions generated through the application of *maxexpvalue* were conditional on the evolving state of the world.

It was shown in [47] that the above formalization is unsatisfactory when the dynamics of the modalities are considered. Problems arise due to the closure under both disjunction and conjunction for each modality. These problems are highlighted by the following examples:

(2.1.1) Example (Closure under Disjunction):

$$(\mathsf{Int}\ (\circ\ \phi)) \wedge (\circ\ (\mathsf{Bel}\ \phi)) \rightarrow (\circ\ (\mathsf{Int}\ \phi))$$

Consider a situation in which John intends to go to the beach. From the axiom above John will maintain this intention as long as he believes it to be achievable. If, or when, John discovers that it is not possible to go to the beach, this intention can be dropped (and, indeed, the static constraints would force it to be dropped). This is just what we want.

However, lets assume that John also believes it possible to fly to London, but has no intention of doing so. Because we have:

$$(\mathsf{Int}\ (\circ\ \text{`}\textit{go to the beach}\text{'}))$$

we also have (under a possible worlds semantics) the disjunctive intention:

$$(\mathsf{Int}\ (\circ\ \text{`}\textit{go to the beach}\text{'} \vee \text{`}\textit{go to London}\text{'}))$$

Now, when it turns out that visiting the beach is impossible, the intention towards visiting the beach will be duly dropped. But, unfortunately, the intention towards the disjunction ('*go to the beach*'$\vee$'*go to London*'*) will be maintained (as the disjunct remains a possibility). From application of the static constraints, it can then be deduced that John, at the next time point, will intend to fly to London. In other words, John will be forced to adopt as new intentions any beliefs about the future he still holds!*

(Georgeff and Rao [47, Section 2])

(2.1.2) Example (Closure under Conjunction):
[Suppose] John intends to obtain milk from the milk bar and cereal from the supermarket. He goes to the milk bar, sees that it is closed, and thus abandons the intention of obtaining milk. As a result John also gives up his intention to have milk and cereal. However, if intentions are closed under conjunction–as they are under a possible worlds model–intending to have milk and cereal implies an intention to have milk and an intention to have cereal. While the former two can no longer hold, using the above axiom of intention maintenance, the intention to have cereal would be (incorrectly) maintained. *(Georgeff and Rao [47, Section 2])*

The proposed solution was the introduction of only-belief, only-desire and only-intend modalities. These differ semantically from the normal belief, desire and intention modalities in that an agent only-believes (only-desires, only-intends) ϕ if ϕ is true in all belief (desire, intention) accessible worlds and the set of belief (desire, intention) accessible worlds include all the worlds where ϕ is true. Formally:

$$M, \omega_t \vDash (\lceil \mathsf{Bel} \rfloor\, \phi) \text{ iff } \forall \omega' \in \mathcal{W}, \omega \mathtt{B} \omega' \text{ iff } M, \omega'_t \vDash \phi \tag{2.12}$$

$$M, \omega_t \vDash (\lceil \mathsf{Des} \rfloor\, \phi) \text{ iff } \forall \omega' \in \mathcal{W}, \omega \mathtt{D} \omega' \text{ iff } M, \omega'_t \vDash \phi \tag{2.13}$$

$$M, \omega_t \vDash (\lceil \mathsf{Int} \rfloor; \phi) \text{ iff } \forall \omega' \in \mathcal{W}, \omega \mathtt{I} \omega' \text{ iff } M, \omega'_t \vDash \phi \tag{2.14}$$

Three functions:

$$\overset{\omega}{\underset{t}{*_{\mathtt{B}}}} : \{\omega' \mid \omega \mathtt{B}_t \omega'\} \mapsto \{\omega'' \mid \omega \mathtt{B}_{t'} \omega''\} \tag{2.15}$$

$$\overset{\omega}{\underset{t}{*_{\mathtt{D}}}} : \{\omega' \mid \omega \mathtt{D}_t \omega'\} \mapsto \{\omega'' \mid \omega \mathtt{D}_{t'} \omega''\} \tag{2.16}$$

$$\overset{\omega}{\underset{t}{*_{\mathtt{I}}}} : \{\omega' \mid \omega \mathtt{I}_t \omega'\} \mapsto \{\omega'' \mid \omega \mathtt{I}_{t'} \omega''\} \tag{2.17}$$

map for each world ω the set of belief (desire, intention) accessible worlds from ω at time t to the set of belief (desire, intention) accessible worlds from ω at time t'. In defining the revisions in this way it is assumed that the uncertainty the agent holds of its environment remains constant as a result of revision. This follows from the fact that these functions map a world at one time point to the next, no fewer worlds are related, nor any additional. Using these functions the following constraint captures the semantic requirement of intention maintenance in light of changing beliefs and desires (adapted from [47, BDFC1]):

2.1.1 Definition (Intention Maintenance Constraint):

$$\forall \omega' \in \omega \mathsf{B}_t \omega', \wp(\overset{\omega}{\underset{t}{*}}_{\mathtt{I}}(\overset{\mathtt{B}}{\mapsto}_{\mathtt{I}}(\omega'))) = \wp(\overset{\omega}{\underset{t}{*}}_{\mathtt{B}}(\omega')) \cap \wp(\overset{\omega}{\underset{t}{*}}_{\mathtt{D}}(\overset{\mathtt{B}}{\mapsto}_{\mathtt{D}}(\omega'))) \cap \wp(\overset{\mathtt{B}}{\mapsto}_{\mathtt{I}}(\omega')) \neq \varnothing \quad (2.18)$$

where:

$\wp$ *: captures the set of paths through the branching time structure that constitute each possible world, beginning at the initial time point* t_0 *and extending into the infinite future.*

$\overset{\mathtt{B}}{\mapsto}_{\mathtt{D}}$ *: is a function that maps belief accessible worlds to desire accessible worlds, for all worlds and all time points.*

$\overset{\mathtt{B}}{\mapsto}_{\mathtt{I}}$ *: is a function that maps belief accessible worlds to intention accessible worlds, for all worlds and all time points.*

All models that satisfy the *intention maintenance constraint* validate the following formulæ(adapted from [47, Theorem 5]):

$$(\mathsf{Int}\ \mathsf{A}\ (\circ\ \mathsf{E}\phi)) \wedge \mathsf{A}\ (\circ\ (\lceil \mathsf{Bel} \rfloor\ \mathsf{A}\phi)) \wedge \mathsf{A}\ (\circ\ (\lceil \mathsf{Des} \rfloor\ \mathsf{A}\phi)) \rightarrow \mathsf{A}\ (\circ\ (\mathsf{Int}\ \mathsf{E}\phi)) \quad (2.19)$$

$$(\lceil \mathsf{Int} \rfloor; \mathsf{A}\ (\circ\ \mathsf{A}\phi)) \wedge \mathsf{A}\ (\circ\ (\mathsf{Bel}\ \mathsf{E}\phi)) \wedge \mathsf{A}\ (\circ\ (\mathsf{Des}\ \mathsf{E}\phi)) \rightarrow \mathsf{A}\ (\circ\ (\mathsf{Int}\ \mathsf{E}\phi)) \quad (2.20)$$

which demonstrates that the constraint as stated avoids the problems of disjunctive and conjunctive closure.

Although BDI logics are able to capture a large number of intuitions about rational thought, there are a number of properties of such logics that make them less than ideal. Of these, the problem of *logical omniscience* is the most devastating. It requires an agent to believe all valid formula and be a perfect reasoner equivalent to closure under logical consequence. The first requirement forbids an agent from

inconsistent beliefs. This means that an agent cannot believe ϕ and ψ when $\neg\psi$ is entailed by ϕ without degenerating into believing anything (all formulæ). The second requirement necessitates an agent treating logically equivalent propositions as equivalent regardless of their content. These requirements fly in the face of resource boundedness given the complexity of the satisfiability and validity problems for these logics are PSPACE complete. These problems are not specific to BDI however, they are a consequence of the possible world semantics utilized to define their meaning. Thus, any other specification language backed by the same semantics will suffer the same problems. A number of alternatives devoid of these issues have been developed in Fagin, Halpern, Moses, and Vardi [38, Chapter 9].

One such alternative solution to the issue of logical omniscience, in addition to conjunctive and disjunctive closure, was proposed in [17]. In particular, the use of non-normal (or impossible) worlds was advocated as a mechanism for the fine grained control of properties of the belief-desire-intention modalities. A non-normal world is a world at which the truth of formulæ need not be recursively specified. The semantics are extended to include a set of such worlds ($\mathcal{W}^\star$) and the accessibility relations for each modality extended to optionally range over these. Constraints on these relations and valuations at the impossible worlds can then be defined in order to extend the axioms supported.

The properties identified [17] as appropriate for intention include:

$\models \neg (\mathsf{Int}\ \phi \wedge \neg\phi)$ — Intention Consistency I

$\models (\mathsf{Int}\ \phi) \rightarrow \neg (\mathsf{Int}\ \neg\phi)$ — Intention Consistency II

$\models (\mathsf{Int}\ \phi) \wedge (\mathsf{Int}\ \psi) \rightarrow (\mathsf{Int}\ \phi \wedge \psi)$ — Conjunctive Composition

$\not\models (\mathsf{Int}\ \phi \wedge \psi) \rightarrow (\mathsf{Int}\ \phi) \wedge (\mathsf{Int}\ \psi)$ — Conjunctive Decomposition

$\not\models (\mathsf{Int}\ \phi \vee \psi) \rightarrow (\mathsf{Int}\ \phi) \vee (\mathsf{Int}\ \psi)$ — Disjunctive Decomposition

$\not\models (\mathsf{Int}\ \phi) \rightarrow (\mathsf{Int}\ \phi \vee \psi)$ — Disjunctive Composition

$\models (\mathsf{Int}\ \phi \vee \psi) \wedge (\mathsf{Int}\ \neg\phi) \rightarrow (\mathsf{Int}\ \psi)$ — Disjunctive Closure

$\not\models (\mathsf{Int}\ \phi) \wedge (\mathsf{Int}\ \phi \rightarrow \psi) \rightarrow (\mathsf{Int}\ \psi)$ — General Intentional Closure

From $\phi \equiv \psi$ do not infer $(\mathsf{Int}\ \phi) \equiv (\mathsf{Int}\ \psi)$ — Intentional Equivalence

and the constraints on the valuations and relations required to satisfy these[10]:

$M, \omega \models \phi \wedge \psi$ iff $M, \omega \models \phi$ and $M, \omega \models \psi$ — RM I

$M, \omega \models \neg\phi$ iff $M, \omega \not\models \phi$ — RM II

$M, \omega \models (\mathsf{Bel}\ \phi)$ iff $\forall \omega' \in \mathcal{W} \cup \mathcal{W}^{\star}$ if $\omega \mathtt{B} \omega'$ then $M, \omega' \models \phi$ — RM $\mathcal{B}$

$M, \omega \models (\mathsf{Int}\ \phi)$ iff $\forall \omega' \in \mathcal{W} \cup \mathcal{W}^{\star}$ if $\omega \mathtt{I} \omega'$ then $M, \omega' \models \phi$ — RM $\mathcal{I}$

$M, \omega \models \neg (\mathsf{Int}\ \phi \wedge \neg\phi)$ iff $\forall \omega \in \mathcal{W}, \mathcal{W} \cap \omega\mathtt{I} \neq \varnothing$ — Intention Consistency I

$M, \omega \models (\mathsf{Int}\ \phi) \rightarrow \neg (\mathsf{Int}\ \neg\phi)$ iff $\forall \omega' \in \omega\mathtt{I}\ M, \omega' \models \phi$ if $\exists \omega'' \in \omega\mathtt{I}\ M, \omega'' \not\models \neg\phi$ — Intention Consistency II

$M, \omega \models (\mathsf{Int}\ \phi) \wedge (\mathsf{Int}\ \psi) \rightarrow (\mathsf{Int}\ \phi \wedge \psi)$ iff $(\forall \omega' \in \omega\mathtt{I}\ M, \omega' \models \phi \wedge M, \omega' \models \psi)$ if $(\forall \omega'' \in \omega\mathtt{I}\ M, \omega'' \models \phi \wedge \psi$ — Conjunctive Composition

$M, \omega \not\models (\mathsf{Int}\ \phi \wedge \psi) \rightarrow (\mathsf{Int}\ \phi) \wedge (\mathsf{Int}\ \psi)$ iff $\exists \omega^{\star} \in \omega\mathtt{I}\ M, \omega^{\star} \not\models \phi$ or $M, \omega^{\star} \not\models \psi$ — Conjunctive Decomposition

$M, \omega \models (\mathsf{Int}\ \phi \vee \psi) \wedge (\mathsf{Int}\ \neg\phi) \rightarrow (\mathsf{Int}\ \psi)$ iff $\forall \omega' \in \omega\mathtt{I}\ M, \omega' \models \phi \vee \psi \wedge M, \omega' \models \neg\phi$ if $\forall \omega'' \in \omega\mathtt{I}\ M, \omega'' \models \psi$ — Disjunctive Closure

[10] Where $\omega\mathtt{I} = \{\omega' \in \mathcal{W} \cup \mathcal{W}^{\star} \mid \omega \mathtt{I} \omega'\}$.

The primary drawback of an approach that utilizes the impossible worlds semantics lie in the philosophical justification for the assumption of the existence of impossible worlds.

2.1.3 Implementations of Intention

Implementations of intentions typically take two forms: as implemented in rational agent architectures, and, as the target of rational reconsideration functions. There are a number of rational agent architectures that claim the concept of intention as central to their operation. A majority of these systems[33, 55, 54, 16] belong to a family of architectures that can trace their history to the Procedural Reasoning System (PRS[44, 45, 46, 57, 109]). From this family tree, AgentSpeak[77] will be treated as prototypical and presented in detail. In particular, we build upon the presentation and extensions of Jason[10]. This choice is justified by the clear separation between architecture, language and semantics emphasized in the AgentSpeak literature in addition to the work relating the semantics of AgentSpeak to those of the BDI logic presented in section 2.1.2. Work towards rational intention reconsideration functions is less advanced, however, the state-of-the-art will be presented following the detailing of the AgentSpeak approach.

AgentSpeak

AgentSpeak[77, 58] is a language and architecture for the specification and implementation of intentional agents. The language provides a declarative means of specifying the initial belief state and behavioural knowledge of such an agent. The architecture provides an abstract specification of the data flow and computation of the processing cycle that defines the agent's reasoning. The semantics formalizes the architecture, and in conjunction with the initial specification of the agent's state, provides the meaning of any AgentSpeak program.

(2.1.2) Definition (AgentSpeak Agent):
An AgentSpeak agent is defined as a tuple:

$$\langle \mathcal{B}, \mathcal{L}_{\mathcal{B}}, Cn, \mathcal{P}, \mathcal{I}, \mathcal{L}_{\mathcal{I}}, \mathcal{E}, \mathcal{A}, \gamma_{\mathcal{E}}, \gamma_{\mathcal{I}}, \gamma_{\mathcal{O}} \rangle$$

where:

(B) *: represents the beliefs of the agent. Traditionally, each belief is represented as a ground atom, however, it need not be so restricted.*

$(\mathcal{L}_{\mathcal{B}})$ *: is the language in which the agent's beliefs are expressed.*

(Cn) *: is the consequence operator by which the agent derives new beliefs from old.*

$(\mathcal{P})$ *: is the set of plans available to the agent. Each plan is of the form of definition 2.1.3*

$(\mathcal{I})$ *: is a set of intentions. Each intention is represented as a stack of partially instantiated plans in which there are both events yet to be dispatched and variables yet to be instantiated.* $\iota[\tau : @ \leftarrow \pi]$ *denotes that plan* $\tau : @ \leftarrow \pi$ *is the topmost plan of intention stack* ι*.*

$(\mathcal{L}_{\mathcal{I}})$ *: is the language in which the agent's intentions are expressed.*

$(\mathcal{E})$ *: denotes the current set of events awaiting processing by the agent. There are two types of events, internal and external. Internal events are those generated through the execution of plans. External events are generated by changes of belief due to new perceptions of the environment. Internal events result in a new plan being pushed onto the stack from which the event originated. External events on the other hand cause the creation of new intention stacks. Each event is a pair* $\langle \tau, \iota \rangle$ *where* τ *is the content of the event and* ι *is the intention from which it originated. When events are externally generated, the intention will be* $\perp$*.*

$(\mathcal{A})$ *: describes the set of actions the agent has scheduled to do.*

$(\gamma_{\mathcal{E}})$ *: is a function that selects from* $\mathcal{E}$ *the next event to be processed. This is a point of customization for agent architects.*

$(\gamma_{\mathcal{O}})$ *: is a function that selects from amongst all the instantiations of all the applicable plans which to adopt as the means by which the event may be handled. This provides an additional point of customization for tailoring the agent's behaviour to its environment.*

$(\gamma_{\mathcal{I}})$ *: is a function that selects the next intention to process. Again, this function represents an avenue for designers to adapt their agents to the problem domain to which they will be applied.*

(γ_{θ}) *: is a function that selects a unifier at random.*

(2.1.3) Definition (AgentSpeak Plan):
An AgentSpeak plan has the form:

$$\tau : @ \leftarrow \pi$$

where:

(τ) *is the event to which the plan responds;*

($@$) *is a condition that must be satisfied by the agent's beliefs in order for the plan to be applicable; and*

(π) *is the sequence of goals and actions that make up the plans content (see figure 2.1).*

agent	::=	$\mathcal{B}\,\mathcal{P}$	
$\mathcal{B}$	::=	$\beta_1, \ldots, \beta_n$	$(n \geq 0)$
$\mathcal{P}$	::=	$\rho_1, \ldots, \rho_n$	$(n \geq 1)$
ρ	::=	$\tau : @ \leftarrow \pi$	
τ	::=	$+atom \mid -atom \mid +\eth \mid -\eth$	
$@$	::=	$condition \mid \top$	
condition	::=	$atom \mid \neg atom \mid condition \;\&\; condition$	
π	::=	$sequent; \top \mid \top$	
sequent	::=	$\alpha \mid \eth \mid update \mid sequent; sequent$	
atom	::=	$\mathsf{P}(\vec{t})$	
α	::=	$\mathsf{A}(\vec{t})$	
$\eth$	::=	$!atom \mid ?atom$	
update	::=	$+\beta \mid -atom$	
β	::=	$\mathsf{P}(\vec{g})$	
term	::=	$\mathsf{F}(\vec{t}) \mid \mathsf{variable} \mid ground$	
ground	::=	$\mathsf{F}(\vec{g}) \mid \mathsf{string} \mid \mathsf{number}$	
$\vec{t}$	::=	$term_1, \ldots, term_n$	$(n \geq 0)$
$\vec{g}$	::=	$ground_1, \ldots, ground_n$	$(n \geq 0)$

Figure 2.1: Formal Grammar for AgentSpeak Language

Given this definition of an agent, an abstract interpreter (algorithm 2.1 on page 33) describing the dynamics of the mental attitudes was proposed [77] and its operational semantics given. The algorithm as presented is abstract in the sense that

there is no failure handling and the interaction with the environment in which the agent is situated is unspecified. The lack of failure handling in the original AgentSpeak(L) was a consequence of the earlier systems it was intended to abstract. These systems, both PRS [57] and dMARS [33, 34], offered such facilities but in incompatible ways. Consequently, subsequent extensions to AgentSpeak, such as Jason, re-introduced failure handling in novel ways. One such extension will be detailed later in this chapter. The formal grammar of the AgentSpeak language that defines the source over which this interpreter operates is given in figure 2.1.

This interpreter gives rise to an associated architecture (figure 2.2). As can be seen from the architecture and abstract interpreter, the execution cycle of the agent contains two independent processes:

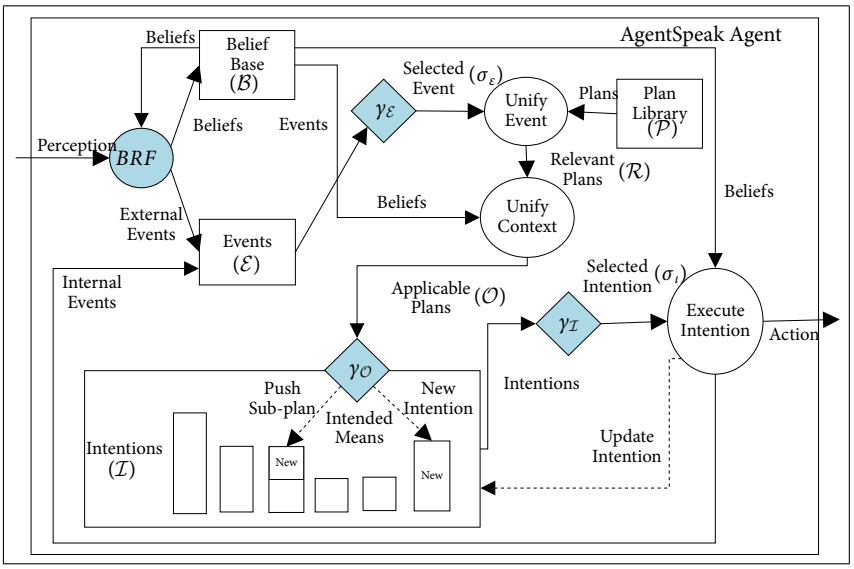


Figure 2.2: AgentSpeak Architecture (adapted from [66])

Belief Revision, Option Selection and Intention Adoption: Firstly, an agent receives a perception from the environment. This causes the agent to reconsider its beliefs (BRF) and revise them accordingly. This revision causes events (external events) indicating that a certain aspect of the environment has changed. These external events and those generated from the execution of plans (internal events) are stored in an event cache ($\mathcal{E}$). Internal events represent sub-goals for the agent to handle at a later point. From the event cache an event is selected ($\gamma_\mathcal{E}$) and all the

plans in the plan base ($\mathcal{P}$) that respond to the event are extracted. If there are no events to process then processing moves to intention processing. Those plans that respond to the selected event are called the relevant plans ($\mathcal{R}$). Each relevant plan is then tested to determine whether it is applicable at the given time ($\mathcal{O}$). A plan is applicable if the context conditions (φ) for its adoption are satisfied by the agent's current belief base. If no plans are valid given the agent's current circumstance, processing moves to intention execution. From these applicable plans one must be selected ($\gamma_{\mathcal{O}}$) as the *intended means*. This involves the placement of the plan onto the appropriate intention stack. If the selected event was generated by a change in beliefs, a new stack is created onto which the plan is placed. Otherwise the event was generated from an already executing plan and the new plan is pushed onto the stack containing this already executing plan.

Plan Execution and Monitoring: From the set of intentions ($\mathcal{I}$), one must be selected ($\gamma_{\mathcal{I}}$) to execute. If there are no intentions pending execution then processing returns to monitoring the environment, revising beliefs and posting events. There are a number of processes invoked depending on the type of the next element in the plan body atop the intention selected for execution. If it is an "achieve goal" then an appropriate event is inserted in the event cache and the intention becomes suspended (only to be reanimated once the event has been handled [58]). If the first element of the body of the top-most plan on the selected intention is a "test goal" then the contents of the test is unified with the elements of the belief base. A random unifier is selected and applied to the containing plan from which test goal is then removed. If no such unification can be made then an "add test" event is posted to the set of events and processing follows analogously to "achieve goals". If the intention is an "action" then the action is executed (added to the action set $\mathcal{A}$) and removed from the plan. Alternatively, if the intention is an assertion then, provided the assertion is ground given the current unifier, it is added to the agent's beliefs and a corresponding event is generated. Finally, if the intention is a retraction, the belief base is searched for all beliefs that unify with the retraction and these are removed, with appropriate events posted as necessary. Once the last intention in a plan is achieved, it is taken from the intention stack and the achieve goal for which it was created is also removed.

The primary motivation for having an event driven language/architecture is that it allows plan selection to be delayed until the last moment before a single plan must be committed to. This allows the agent to make use of the most up-to-date information when deciding on the appropriateness of various alternatives.

This is advantageous because the difference between the belief base at the time of dispatching the event and the time at which it is actually processed may be considerable. Because many plans may respond to the same event, the context provides a means for selecting only those appropriate for the given environmental conditions and through the unification process adapts the generic plan to the specific situation. However, the unification process does not guarantee a unique plan for a given event and in many cases may give rise to more options. The event driven framework also facilitates the use of recursion as the primary repetition construct.

```
1  void Reason() {
2     B = beliefs;
3     P = plans;
4     A = ∅;
5     E = ∅;
6     I = ∅;
7     while(⊤) {
8        p = perceive();
9        ⟨B, E'⟩ = BRF(B, p);
10       E = E ∪ E';
11       if (E ≠ ∅) {
12          Process−Event(E, B, P, I);
13       }
14       if (I ≠ ∅) {
15          Process−Intention(E, B, I, A);
16       }
17    }
18 }
```

Algorithm 2.1: Abstract AgentSpeak Interpreter (as adapted from [77])

The AgentSpeak interpreter (algorithm 2.1) contrasts with the traditional specification of a BDI practical reasoning agent interpreter (algorithm 2.5) in that the traditional approach is purely procedurally specified whereas AgentSpeak is inherently event driven. However, the intuition and dynamics remain largely equivalent.

```
1  void Clean−Intention(ι) {
2    while (first(body(top(ι))) = ⊤) {
3      x = pop(ι);
4      // where θ is a maximally general unifier such that xθ = head(top(ι))θ
5      push(head(top(ι))θ ← rest(body(top(ι)))θ, ι);
6    }
7  }
```

Algorithm 2.2: Remove Complete Intention

```
1  void Process−Event(ℰ, ℬ, 𝒫, ℐ) {
2    ⟨τ, ι⟩ = γ_ℰ(ℰ);
3    ℰ = ℰ ∖ {⟨τ, ι⟩};
4    ℛ = RelPlans(𝒫, ⟨τ, ι⟩); // see definition 2.1.5 on page 46
5    if (ℛ = ∅) {
6      // Error handling specifies whether ε is returned to ℰ
7      return;
8    }
9    𝒪 = ApplPlans(ℬ, ℛ); // see definition 2.1.6 on page 46
10   if (𝒪 = ∅) {
11     // Error handling specifies whether ε is returned to ℰ
12     return;
13   }
14   if (ι = ⊥) {
15     ℐ = ℐ ∪ {γ_𝒪(𝒪)};
16   } else {
17     push(γ_𝒪(𝒪), ι);
18   }
19 }
```

Algorithm 2.3: Process an Event

```
1  void Process–Intention(ℰ, ℬ, ℐ, 𝒜) {
2    switch(first(body(top(γ_ℐ(ℐ))))) {
3      case ⊤: Clean–Intention(γ_ℐ(ℐ)); break;
4      case !∂(t⃗):
5        ℰ = ℰ ∪ ⟨+!∂(t⃗), γ_ℐ(ℐ)⟩;
6        ℐ = ℐ ∖ γ_ℐ(ℐ);
7      break;
8      case ?atom(t⃗):
9        pop(γ_ℐ(ℐ));
10       if (∃β ∈ ℬ | θβ = θatom(t⃗)) {
11         push(head(top(γ_ℐ(ℐ)))θ ← rest(body(top(γ_ℐ(ℐ))))θ, γ_ℐ(ℐ));
12       } else {
13         ℰ = ℰ ∪ ⟨+?atom(t⃗), γ_ℐ(ℐ)⟩;
14         ℐ = ℐ ∖ γ_ℐ(ℐ);
15       }
16     break;
17     case α(t⃗):
18       pop(γ_ℐ(ℐ));
19       push(head(top(γ_ℐ(ℐ))) ← rest(body(top(γ_ℐ(ℐ)))), γ_ℐ(ℐ));
20       𝒜 = 𝒜 ∪ {α(t⃗)};
21     break;
22     case +atom(g⃗):
23       ℬ = ℬ ∪ {atom(g⃗)};
24       ℰ = ℰ ∪ {⟨+atom(g⃗), γ_ℐ(ℐ)⟩};
25       pop(γ_ℐ(ℐ));
26       push(head(top(γ_ℐ(ℐ))) ← rest(body(top(γ_ℐ(ℐ)))), γ_ℐ(ℐ));
27     break;
28     case −atom(t⃗):
29       for (β ∈ ℬ | ∃θ . β = atom(t⃗)θ) {
30         ℬ = ℬ ∖ {β};
31         ℰ = ℰ ∪ {⟨−β, γ_ℐ(ℐ)⟩};
32       }
33       pop(γ_ℐ(ℐ));
34       push(head(top(γ_ℐ(ℐ))) ← rest(body(top(γ_ℐ(ℐ)))), γ_ℐ(ℐ));
35     break;
36   }
37 }
```

Algorithm 2.4: Process an Intention

```
1  void BDI_Control() {
2     B = beliefs;
3     I = intentions;
4     while (⊤) {
5        p = perceive();
6        B = BRF(B, p);
7        D = options(B, I);
8        I = filter(B, D, I);
9        ρ = plan(B, I, A);
10       while (¬(empty(ρ) ∨ succeeded(I, B) ∨ impossible(I, B))) {
11          α = first(ρ);
12          execute(α);
13          ρ = rest(ρ);
14          p = perceive();
15          B = BRF(B, p);
16          if (reconsider(I, B)) {
17             D = options(B, I);
18             I = filter(B, D, I);
19          }
20          if (¬(sound(ρ, I, B))) {
21             ρ = plan(B, I, A);
22          }
23       }
24    }
25 }
```

Algorithm 2.5: BDI Control Loop (adapted from [110])

Rubbish Robot Example

```
1  adjacent(a,b).
2  adjacent(b,c).
3  adjacent(c,d).
4  adjacent(d,a).
5  location(robot,a).
6  location(bin,d).
7
8  +!location(robot,X) : location(robot,X)
9    <-
10       TRUE.
11
12 +!location(robot,X) : location(robot,Y)
13       & adjacent(Y,Z)
14       & not(location(robot,X))
15   <-
16       move(Y,Z);
17       !location(robot,X).
18
19 +location(waste,X) : not(location(robot,X))
20       & location(bin,Y)
21   <-
22       !location(robot, X);
23       pick(waste);
24       !location(robot, Y);
25       drop(waste).
```

Figure 2.3: Example AgentSpeak code for a rubbish robot (adapted from [77])

Figure 2.3 gives a brief example of an AgentSpeak program. It demonstrates how beliefs are specified and outlines the structure of plans. As can be seen from the syntax, AgentSpeak is heavily influenced by logic programming, in particular Prolog, relying on variable substitutions, logical consequence procedures, negation as failure and the requirement that atoms be ground before negation. Aside from these similarities there are some differences between AgentSpeak and traditional logic programming: Plans consist of an event to which they respond, a context which must be be satisfied with respect to the current beliefs and a list of actions and/or goals. Contrastingly, a rule in a traditional logic program consists of a head and body. The sequence of actions/goals within a plan are executed in a stepwise manner whereas the entire body of a rule is executed as a single unit. This allows the actions/goals of multiple plans to be interleaved. The triggering events within this framework, can be both data and goal directed. Plans can be invoked upon the addition or deletion of belief(s) or directly by other plans. Conversely, in traditional logic programming rules are only invoked upon a query or by other rules.

The operational semantics[104, 74] of an AgentSpeak program[11, 9, 10] is a transition system over the tuple $\langle agent, C, T, S\rangle$ where:

(2.1.4) Definition (Operational Semantics (adapted from [10])):
The basis of the states in the transition system that defines the semantics of an AgentSpeak program are:

(*agent*) *: is the agent program defined according to the formal grammar (see figure 2.1) including the selection functions $\gamma_{\mathcal{E}}$, $\gamma_{\mathcal{O}}$, $\gamma_{\mathcal{I}}$ and γ_{θ}.*

(C) *: is the agent's circumstance $\langle \mathcal{B}, \mathcal{P}, \mathcal{I}, \mathcal{E}, \mathcal{A}\rangle$ that encapsulates the elements of the agent's state that are persistent. $\mathcal{B}$, $\mathcal{P}$, $\mathcal{I}$, $\mathcal{E}$ and $\mathcal{A}$ are as per definition 2.1.2.*

(T) *: The elements of this tuple $\langle \mathcal{R}, \mathcal{O}, \sigma_\iota, \sigma_\varepsilon, \sigma_\omega\rangle$ are the assignments an agent makes within its mental state that are only valid within a single reasoning cycle.*

- ($\mathcal{R}$) *: are the (relevant) plans that have triggering events that unify with the selected event (σ_ε)[see definition 2.1.5].*
- ($\mathcal{O}$) *: are the relevant plans whose context is a consequence of the agent's beliefs (applicable plans)[see definition 2.1.6].*
- ($\sigma_\iota, \sigma_\varepsilon, \sigma_\omega$) *: are the selected intention, event and option respectively.*

(S) *: the symbolic representation of the rules in the agent's reasoning cycle:*

$$\{\mathsf{Init}, \mathsf{SelEv}, \mathsf{RelPl}, \mathsf{ApplPl}, \mathsf{SelAppl}, \mathsf{AddIM}, \mathsf{SelInt}, \mathsf{ExecInt}, \mathsf{ClrInt}\}$$

Each rule in the transition system contains an antecedent, consequent and optional qualifications. The antecedent defines the conditions under which the transition is valid. The consequent is the definition of the transition. The current state must be reflected in the tuple on the left of the transition relation ($\longrightarrow$) for the transition to be possible. The transition moves the state from that of the left of the transition relation to that on the right. The elements of the resultant state can be refined using further qualifications defining their resulting value. It should be noted that unless otherwise indicated, the values of unmentioned components remains unchanged.

Rule 2.1.1 is used to initialize the agent's beliefs, plans and actions. The beliefs and plans are initialized from the supplied agent program (*agent*). The actions are initialized to the empty set. It corresponds to lines 2 to 6 of algorithm 2.1 and lines 2 to 3 of algorithm 2.5.

$$\frac{\top}{\langle agent, C, T, \mathsf{Init}\rangle \longrightarrow \langle agent, C', T, \mathsf{SelEv}, \rangle}$$

Rule 2.1.1
Init

where: $C'_{\mathcal{B}} = agent_{\mathcal{B}}$, $C'_{\mathcal{P}} = agent_{\mathcal{P}}$, $C'_{\mathcal{A}} = \varnothing$, $C'_{\mathcal{E}} = \varnothing$ and $C'_{\mathcal{I}} = \varnothing$

Rules 2.1.2 and 2.1.3 formalize the process of selecting an event and removing it from the set of events. When there are no events the agent proceeds to processing its intentions instead. These rules capture the intuition of lines 11 to 13 of algorithm 2.1 and lines 2 to 3 of algorithm 2.3[11].

$$\frac{agent_{\gamma\varepsilon}(C_{\mathcal{E}}) = \langle \tau, \iota\rangle}{\langle agent, C, T, \mathsf{SelEv}\rangle \longrightarrow \langle agent, C', T', \mathsf{RelPl}\rangle}$$

Rule 2.1.2
SelEv$_1$

where: $C'_{\mathcal{E}} = C_{\mathcal{E}} \setminus \{\langle \tau, \iota\rangle\}$ and $T'_{\sigma_\varepsilon} = \langle \tau, \iota\rangle$

[11]The process of inspecting the environment and incorporating this into the agent's beliefs remains beyond the current state of the art in published semantics for AgentSpeak. Thus there is no rule pertaining to this.

$$\frac{C_{\mathcal{E}} = \varnothing}{\langle agent, C, T, \mathsf{SelEv}\rangle \longrightarrow \langle agent, C, T, \mathsf{SelInt}\rangle}$$

Rule 2.1.3
SelEv$_2$

The process of generating the set of relevant plans is formalized in rules 2.1.4 and 2.1.5. Rule 2.1.4 handles the case when the set of relevant plans is not empty. Rule 2.1.5 handles the case where the set of relevant plans is empty and forwards processing onto intention execution. These rules capture the essence of lines 5 to 8 of algorithm 2.3 and line 7 of algorithm 2.5. Central to the operation and meaning of these rules is the function RelPlans$(C_{\mathcal{P}}, \tau)$ (definition 2.1.5).

$$\frac{T_{\sigma_\varepsilon} = \langle \tau, \iota\rangle \wedge \text{RelPlans}(C_{\mathcal{P}}, \tau) \neq \varnothing}{\langle agent, C, T, \mathsf{RelPl}\rangle \longrightarrow \langle agent, C, T', \mathsf{ApplPl}\rangle}$$

where: $T'_{\mathcal{R}} = \text{RelPlans}(C_{\mathcal{P}}, \tau)$

Rule 2.1.4
RelPl$_1$

$$\frac{T_{\sigma_\varepsilon} = \langle \tau, \iota\rangle \wedge \text{RelPlans}(C_{\mathcal{P}}, \tau) = \varnothing}{\langle agent, C, T, \mathsf{RelPl}\rangle \longrightarrow \langle agent, C, T, \mathsf{SelInt}\rangle}$$

Rule 2.1.5
RelPl$_2$

Applicable plans are captured in the semantics by way of rules 2.1.6 and 2.1.7 which apply the function ApplPlans$(C_{\mathcal{B}}, T_{\mathcal{R}})$ (see definition 2.1.6). This function is the generator for the agent's options provided the agent's beliefs and relevant plans. These rules capture the spirit of lines 9 to 13 of algorithm 2.5 and line 8 of algorithm 2.3.

$$\frac{\text{ApplPlans}(C_{\mathcal{B}}, T_{\mathcal{R}}) \neq \varnothing}{\langle agent, C, T, \mathsf{ApplPl}\rangle \longrightarrow \langle agent, C, T', \mathsf{SelAppl}\rangle}$$

Rule 2.1.6
ApplPl$_1$

where: $T'_{\mathcal{O}} = \text{ApplPlans}(C_{\mathcal{B}}, T_{\mathcal{R}})$

$$\frac{\text{ApplPlans}(C_{\mathcal{B}}, T_{\mathcal{R}}) = \varnothing}{\langle agent, C, T, \mathsf{ApplPl}\rangle \longrightarrow \langle agent, C, T', \mathsf{SelInt}\rangle}$$

Rule 2.1.7
ApplPl$_2$

Given there are applicable plans to choose from, this rule (rule 2.1.8) applies the option selection function to the available options. Although different in implementation, this corresponds intuitively to line 9 of algorithm 2.5 in that both result in a single means for the achievement of the agent's intention(s). In conjunction with rules 2.1.10 and 2.1.9 the correlation with lines 14 to 18 of algorithm 2.3 is clear.

$$\frac{agent_{\gamma_{\mathcal{O}}}(T_{\mathcal{O}}) = \langle \rho, \theta\rangle}{\langle agent, C, T, \mathsf{SelAppl}\rangle \longrightarrow \langle agent, C, T', \mathsf{AddIM}\rangle}$$

Rule 2.1.8
SelAppl

where: $T'_{\sigma_\omega} = \langle \rho, \theta\rangle$

$$\frac{T_{\sigma_\varepsilon} = \langle \tau, \bot\rangle \wedge T_{\sigma_\omega} = \langle \rho, \theta\rangle}{\langle agent, C, T, \mathsf{AddIM}\rangle \longrightarrow \langle agent, C, T', \mathsf{SelInt}\rangle}$$

Rule 2.1.9
ExtEv

where: $C'_{\mathcal{I}} = C_{\mathcal{I}} \cup \{[\rho\theta]\}$

$$\frac{T_{\sigma_\varepsilon} = \langle \tau, \iota \rangle \wedge T_{\sigma_\omega} = \langle \rho, \theta \rangle}{\langle agent, C, T, \mathsf{AddIM} \rangle \longrightarrow \langle agent, C, T', \mathsf{SelInt} \rangle}$$

Rule 2.1.10 **IntEv**

where: $C'_{\mathcal{I}} = C_{\mathcal{I}} \cup \{\iota[\rho\theta]\}$

Intention selection is achieved in rules 2.1.11 and 2.1.12 through the application of the customizable intention selection function. Again, there is a close relation between this rule and portions of the AgentSpeak interpreter (line 15 of algorithm 2.1 and line 2 of algorithm 2.4). Here, however, the close relationship with the generic BDI control loop of algorithm 2.5 is broken. The differences in approach become apparent and irreconcilable. The classic BDI loop adopts a single completely ground[12] plan with which it aims to satisfy all of its concurrent intentions. The plan is reconsidered whenever the environment evolves in such a way to invalidate it. AgentSpeak, on the other hand, maintains multiple partially instantiated plan stacks for each focus of intention it is committed to. Planning is incremental and utilizes sub-goals as the mechanism for this. Once intended, validity checking of the agent's plans is relaxed and progresses in concert with the agent's current focus.

$$\frac{C_{\mathcal{I}} \neq \varnothing \wedge agent_{\gamma_{\mathcal{I}}}(C_{\mathcal{I}}) = \iota}{\langle agent, C, T, \mathsf{SelInt} \rangle \longrightarrow \langle agent, C, T', \mathsf{ExecInt} \rangle}$$

Rule 2.1.11 **SelInt$_1$**

where: $T'_{\sigma_\iota} = \iota$

$$\frac{C_{\mathcal{I}} = \varnothing}{\langle agent, C, T, \mathsf{SelInt} \rangle \longrightarrow \langle agent, C, T', \mathsf{SelEv} \rangle}$$

Rule 2.1.12 **SelInt$_2$**

[12] A ground plan is one in which all actions are primitive and contain no outstanding variable bindings.

The following rules 2.1.13 to 2.1.18 embody the mechanisms for handling the different types of goal available in the body of AgentSpeak plans. As such, each rule is the formalization of the corresponding case in algorithm 2.4.

$$\frac{T_{\sigma_\iota} = \iota[\tau : @ \leftarrow \alpha; \pi]}{\langle agent, C, T, \mathsf{ExecInt}\rangle \longrightarrow \langle agent, C', T', \mathsf{ClrInt}\rangle}$$

Rule 2.1.13 **Action**

where: $C'_{\mathcal{A}} = C_{\mathcal{A}} \cup \alpha$
$T'_{\sigma_\iota} = \iota[\tau : @ \leftarrow \pi]$
$C'_{\mathcal{I}} = (C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\}) \cup \{T'_{\sigma_\iota}\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau : @ \leftarrow !atom; \pi]}{\langle agent, C, T, \mathsf{ExecInt}\rangle \longrightarrow \langle agent, C', T, \mathsf{SelEv}\rangle}$$

Rule 2.1.14 **AchvGl**

where: $C'_{\mathcal{E}} = C_{\mathcal{E}} \cup \{\langle +!atom, T_{\sigma_\iota}\rangle\}$
$C'_{\mathcal{I}} = C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau : @ \leftarrow ?atom; \pi] \wedge \{\theta \mid atom\theta \in Cn(C_{\mathcal{B}})\} \neq \varnothing}{\langle agent, C, T, \mathsf{ExecInt}\rangle \longrightarrow \langle agent, C', T, \mathsf{ClrInt}\rangle}$$

Rule 2.1.15 **TestGl$_1$**

where: $T'_{\sigma_\iota} = \iota[(\tau : @ \leftarrow \pi)]\theta'$
$\theta' = agent_{\gamma_\theta}(\{\theta \mid atom\theta \in Cn(C_{\mathcal{B}})\})$
$C'_{\mathcal{I}} = (C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\}) \cup \{T'_{\sigma_\iota}\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau : @ \leftarrow ?atom; \pi] \wedge \{\theta \mid atom\theta \in Cn(C_\mathcal{B})\} = \varnothing}{\langle agent, C, T, \mathsf{ExecInt}\rangle \longrightarrow \langle agent, C', T, \mathsf{ClrInt}\rangle}$$

Rule 2.1.16 **TestGl$_2$**

where: $C'_\mathcal{E} = C_\mathcal{E} \cup \{\langle +?atom, T_{\sigma_\iota}\rangle\}$
$C'_\mathcal{I} = C_\mathcal{I} \setminus \{T_{\sigma_\iota}\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau : @ \leftarrow +\beta; \pi]}{\langle agent, C, T, \mathsf{ExecInt}\rangle \longrightarrow \langle agent, C', T', \mathsf{ClrInt}\rangle}$$

Rule 2.1.17 **AddBel**

where: $C'_\mathcal{B} = C_\mathcal{B} \cup \{\beta\}$
$C'_\mathcal{E} = C_\mathcal{E} \cup \{\langle +\beta, \bot\rangle\}$
$T'_{\sigma_\iota} = \iota[\tau : @ \leftarrow \pi]$
$C'_\mathcal{I} = (C_\mathcal{I} \setminus \{T_{\sigma_\iota}\}) \cup \{T'_{\sigma_\iota}\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau : @ \leftarrow -atom; \pi]}{\langle agent, C, T, \mathsf{ExecInt}\rangle \longrightarrow \langle agent, C', T', \mathsf{ClrInt}\rangle}$$

Rule 2.1.18 **DelBel**

where: $C'_\mathcal{B} = C_\mathcal{B} \setminus \{\beta \in C_\mathcal{B} \mid \exists\theta \,.\, \beta = atom\theta\}$
$C'_\mathcal{E} = C_\mathcal{E} \cup \{\langle -\beta, \bot\rangle \mid \beta \in C_\mathcal{B} \wedge \exists\theta \,.\, \beta = atom\theta\}$
$T'_{\sigma_\iota} = \iota[\tau : @ \leftarrow \pi]$
$C'_\mathcal{I} = (C_\mathcal{I} \setminus \{T_{\sigma_\iota}\}) \cup \{T'_{\sigma_\iota}\}$

These rules (rules 2.1.19 to 2.1.21) handle the clearing of intentions whose ends have been met by their completed means. Cumulatively, they correspond to algorithm 2.2.

$$\frac{T_{\sigma_\iota} = [\tau : \varpi \leftarrow \top]}{\langle agent, C, T, \mathsf{ClrInt}\rangle \longrightarrow \langle agent, C', T, \mathsf{SelEv}\rangle}$$

Rule 2.1.19
ClrInt$_1$

where: $C'_{\mathcal{I}} = C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau' : \varpi' \leftarrow \eth; \pi][\tau : \varpi \leftarrow \top]}{\langle agent, C, T, \mathsf{ClrInt}\rangle \longrightarrow \langle agent, C', T, \mathsf{ClrInt}\rangle}$$

Rule 2.1.20
ClrInt$_2$

where: $C'_{\mathcal{I}} = (C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\}) \cup \{\iota[(\tau' : \varpi' \leftarrow \pi)\theta]\}$
$\theta = agent_{\gamma_\theta}(\{\theta \mid \theta\tau = \theta\eth\})$

$$\frac{T_{\sigma_\iota} \neq [\tau : \varpi \leftarrow \top] \wedge T_{\sigma_\iota} \neq \iota[\tau : \varpi \leftarrow \top]}{\langle agent, C, T, \mathsf{ClrInt}\rangle \longrightarrow \langle agent, C, T, \mathsf{SelEv}\rangle}$$

Rule 2.1.21
ClrInt$_3$

The above transition system can be visualized as follows:

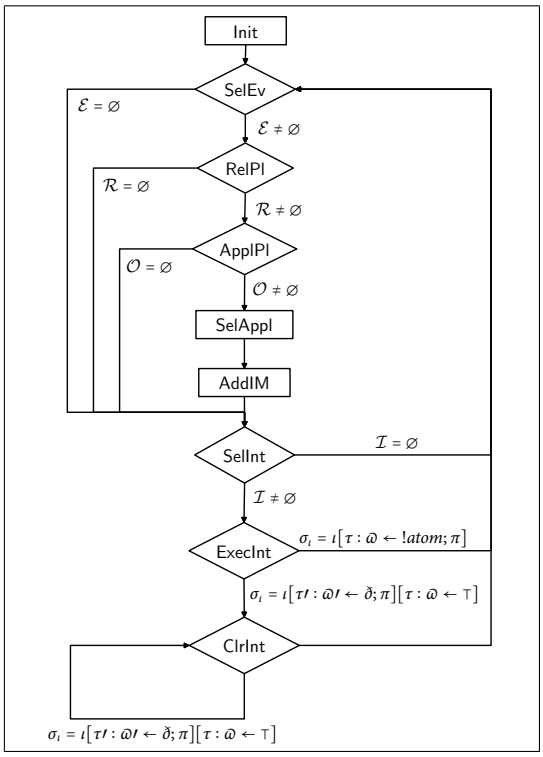


Figure 2.4: AgentSpeak State Transition Diagram

2.1.5 Definition (Relevant Plans):

The relevant plans include all the plans in the input plan library ($\mathcal{P}$) that have a trigger that can be unified with the input triggering event (τ'). Thus the relevant plan set is the set of such plans and their associated unifiers.

$$\text{RelPlans}(\mathcal{P}, \tau') = \left\{ \langle \rho, \theta \rangle \;\middle|\; \begin{array}{l} \rho \in \mathcal{P} \wedge \\ \rho = \tau : \bar{\omega} \leftarrow \pi \wedge \\ \theta\tau = \tau'\theta \end{array} \right\}$$

2.1.6 Definition (Applicable Plans):

The applicable plans are those in the input set of plan (ρ), unifier (θ) pairs ($\mathcal{R}$) such that the input belief base ($\mathcal{B}$) entails the context of the plan given the existing unifier (θ) and any additional bindings necessary (θ').

$$\text{ApplPlans}(\mathcal{B}, \mathcal{R}) = \left\{ \langle \rho, \theta' \circ \theta \rangle \;\middle|\; \begin{array}{l} \langle \rho, \theta \rangle \in \mathcal{R} \wedge \\ \rho = \tau : \bar{\omega} \leftarrow \pi \wedge \\ \bar{\omega}\theta\theta' \in Cn(\mathcal{B}) \end{array} \right\}$$

AgentSpeak has been extended along a number of axis since its introduction: efficient intention selection mechanisms utilizing decision-theoretic task scheduling [7]; speech act base inter-agent communication [68]; plan exchange [4]; constraints and optimization goals [26]; ontological reasoning [69]; and automated belief revision [3].

Much of the motivation for the development of AgentSpeak was to bridge the gap between implemented agent systems and their theoretical underpinnings. To this end, a subset of AgentSpeak (as presented here) has been verified[12, 11] to satisfy the asymmetry thesis and other philosophical requirements of rationality as formalized in BDI logic. Additionally an approach to the verification of the properties of AgentSpeak programs via model checking has been developed [8]. In proving the asymmetry thesis, the following equivalences were proposed for the beliefs, desires and intentions of an AgentSpeak agent.

2.1.7 Definition (Belief):
An agent in circumstance C believes a formula ϕ provided ϕ is a logical consequence of the agent's belief [11].

$$(\mathsf{Bel}_C\ \phi) \equiv \phi \in Cn(C_{\mathcal{B}})$$

2.1.8 Definition (Intention):
An agent in circumstance C intends a formula ϕ provided ϕ is the content of an achieve event that is the triggering event of a plan within the agent's intentional structure or part of an intention currently associated with an event in the agent's pending events [11].

$$(\mathsf{Int}_C\ \phi) \equiv \phi \in \bigcup_{\iota \in C_{\mathcal{I}}} agls(\iota) \vee \bigcup_{\langle \tau, \iota \rangle \in C_{\mathcal{E}}} agls(\iota)$$

where:

$$\begin{aligned} agls(\bot) &= \varnothing \\ agls(\iota[\rho]) &= \begin{cases} \{atom\} \cup agls(\iota) & \text{if } \rho = +!atom : @ \leftarrow \pi \\ agls(\iota) & \text{otherwise} \end{cases} \end{aligned}$$

2.1.9 Definition (Desire):
An agent in circumstance C desires a formula ϕ provided the agent intends it or there is an achieve event pending such that ϕ is its content [11].

$$(\mathsf{Des}_C\ \phi) \equiv \langle +!\phi, \iota \rangle \in C_{\mathcal{E}} \vee (\mathsf{Int}_C\ \phi)$$

As demonstrated in [11, 12] AgentSpeak satisfies the following axioms:

Axiom	Satisfied?
$\models (\mathsf{Int}\ \phi) \rightarrow \neg (\mathsf{Bel}\ \neg\phi)$	$\bot$
$\not\models (\mathsf{Int}\ \phi) \rightarrow (\mathsf{Bel}\ \phi)$	$\top$
$\not\models (\mathsf{Bel}\ \phi) \rightarrow (\mathsf{Int}\ \phi)$	$\top$
$\models (\mathsf{Int}\ \phi) \rightarrow \neg (\mathsf{Des}\ \neg\phi)$	$\top$
$\not\models (\mathsf{Int}\ \phi) \rightarrow (\mathsf{Des}\ \phi)$	$\bot$
$\not\models (\mathsf{Des}\ \phi) \rightarrow (\mathsf{Int}\ \phi)$	$\top$
$\models (\mathsf{Des}\ \phi) \rightarrow \neg (\mathsf{Bel}\ \neg\phi)$	$\bot$
$\not\models (\mathsf{Des}\ \phi) \rightarrow (\mathsf{Bel}\ \phi)$	$\top$
$\not\models (\mathsf{Bel}\ \phi) \rightarrow (\mathsf{Des}\ \phi)$	$\top$

Table 2.6: AgentSpeak and the Asymmetry Thesis

Intention Revision

For an agent to appropriately balance the commitment it places in a given intention against the opportunities such an intention prohibits and the dynamic properties of the environment on which it depends, appropriate intention reconsideration policies are necessary. Such policies must provide computationally efficient approximations of the benefits of reconsideration so that the cost of reconsideration can be avoided when reconsideration cannot be justified. A number of techniques have been applied to this end and experimentally evaluated in the context of the Tileworld[76] benchmark domain. The first such methodologies[61, 62, 63] were static policies that reconsidered intentions at fixed intervals. When the interval between reconsideration was small, an agent was claimed to act *cautiously*. In contrast, as the interval between reconsideration operations increased, the agent's *boldness* was said to increase. An agent's *boldness* was then evaluated against different environmental factors and different costs of planning. The *life-expectancy* of holes, that defined how long a hole persisted in the environment, was varied. The *gestation period*, too, was controlled. The gestation period defined the interval between the appearance of successive holes. The time to process perception information, called

the *time cost of sensing*, was another control variable. The *dynamism* of the environment, which determined the ratio of environment cycles to the number of agent reasoning cycles, was another factor examined during the testing. The *accessibility* the agent had to the information in the environment was also controlled. The accessibility was defined as the proportion of the environment the agent can receive sense information from. Another control was the *determinism* of the environment. This determined the probability with which actions produced their expected effects. Finally, the *noise* inherent in the agent's sensors was varied in a controlled way. This determined the probability with which the information provided to the agent was accurate. Given these varied environmental factors, an agent's *effectiveness* was determined as the proportion of the number of holes visited given the total number of holes it may have potentially visited. The experiments demonstrated that:

1. The effectiveness of an agent decreased as the dynamism of the environment and the boldness of the agent increased. As the boldness of the agent increased it re-planned less frequently. As the dynamism of the environment increased, holes appeared and disappeared more rapidly. This resulted in holes disappearing before the agent was capable of reaching them.

2. The cost of planning was more influential on the success of cautious agent's than those that were bold. This follows directly from the fact that a cautious agent will re-plan more frequently than a bold agent and thus incur the additional cost more frequently.

3. The cost of planning was more influential on effectiveness than the dynamism of the environment.

4. When the agent's accessibility was limited, the amount of deliberation an agent engaged in did not influence its effectiveness. This was due to the lack of information for the agent to deliberate over. This rendered deliberation ineffective. As accessibility increased, so did the agent's effectiveness. A corollary, was that planning cost was shown irrelevant in light of varied accessibility.

5. The effectiveness of an agent was demonstrated to be increase linearly with respect to the accuracy of the information with which it was provided.

6. Dynamism of the environment was more influential on agent effectiveness than accessibility.

7. Accessibility to the environment is of greater importance to agent effectiveness than determinism.

8. Dynamism is more influential than determinism.

The reasoning capabilities of the agents in question were then extended to more sophisticated reconsideration strategies and the experiments revisited[90, 91, 92, 73]. The first such strategy was that of *discrete deliberation scheduling* that treats the choice to act and the choice to deliberate as actions. Decision theory was then utilized to select the "action" with the highest utility, be it reconsideration or action. Thus, the reconsideration function of algorithm 2.5 can be implemented as a function that:

1. generates a utility for the default action (the first action in the agent's currently executing plan) through conditionalizing the probability of the action generating a particular state of the environment by the utility of that state of the environment;

2. generates the utility of deliberating given the agent's beliefs about the state and dynamics of the environment; and,

3. choosing to deliberate only when the utility of doing so is greater than the utility of acting (see algorithm 2.6).

The abstract *discrete deliberation scheduling* algorithm (algorithm 2.6) was then specialized for the Tileworld domain [93]. The utility of a given state of the environment ($U_e(e)$) was defined as the inverse of the distance the agent is in that environment state from the hole the agent intends to move towards. The utility of deliberation $U_\alpha(\textit{deliberate})$ was defined as the average distance between the agent and every location on the Tileworld divided by the estimated number of new holes that have appeared since the agent last deliberated. The resulting agent was then shown to be as effective as the better performing of the cautious and bold agents for each scenario while executing fewer actions to do so.

The second approach applied to intention reconsideration in the context of the Tileworld was the off-line solution of a partially observable Markov decision processes to generate an optimal reconsideration strategy [92]. A partially observable Markov decision process is a system of states, such that at any time the system is in one of these states, the current state cannot be determined with complete certainty and transitions between states are the result of the execution of named actions. Thus, these systems can be formalized as a tuple $\langle S, \mathcal{A}, O, R, T \rangle$ where:

```
1  boolean reconsider(B, I) {
2     ρ = γ_I(I);
3     α_default = first(ρ);
4     U_α(α_default) = Σ_{e∈E} P(e | α_default) U_e(e);
5     U_α(deliberate) = computeUtility(B);
6     if (U_α(deliberate) > U_α(default)) {
7        return ⊤;
8     }
9     return ⊥;
10 }
```

Algorithm 2.6: Discrete Deliberation Scheduling

(S) is the set of states;

(A) is the set of actions;

(O) is an observation function that defines a probability distribution over the set of observations (Ω) given a state and the action executed in the state ($O : S \times \mathcal{A} \mapsto \Pi(\Omega)$);

(R) is the reward function that defines the reward for executing a given action in a particular state ($R : S \times \mathcal{A} \mapsto \mathbb{R}$); and,

(T) is a state transition function that defines a probability distribution over the states resulting from the execution of an action in a given state ($T : S \times \mathcal{A} \mapsto \Pi(S)$).

The generic formalization above was then adapted to the Tileworld domain by instantiating: each state as a probability distribution over the set of states that defines the agent's beliefs and a set of propositions that constitute the agent's intentions; the set of actions as the set containing the action to deliberate and the action to act $\{act, deliberate\}$; and, the reward function as mirroring the evaluation of the *worth* of the state the agent intends whenever the agent decides to act, and, the evaluation of the *worth* of the current state whenever the agent chooses to deliberate. Once solved, the resulting optimal reconsideration policy, to reconsider whenever a hole that is closer than that intended appears or the intended hole disappears, was shown to produce an agent as effective, but no more, than the *discrete decision scheduling* with higher overall commitment to its intentions but suffering a higher cost of action.

2.2 Effects

There are a myriad of purposes to which reasoning with the outcomes of action can be applied. Subsequently, this variety of purposes is reflected in the multitude of reasoning mechanisms developed to satisfy these applications. This diversity of reasoning procedures has led to the development of a substantial number of representations that ease the reasoning required to comply with one or more of these applications. A non-exhaustive selection of these representation and reasoning schemes will be presented below. Following this, a number of strategies for the monitoring of effects and environments will be outlined.

2.2.1 Representation & Reasoning

The representation of, and reasoning with, actions and their effects is of central concern in practical reasoning systems. This follows from the need of such systems to reason from the ends they desire and intend to means by which they may be achieved. However, this is not the only problem that formalizations of action and effects are utilized to solve. Such formalisms can be used deductively to reason from a specification of actions, their effects and their order of execution to what holds as a result. They can be applied to explain the timing of action execution from a specification of the consequences of actions and specifications of the world at particular time points. Alternatively, they can be used to induce the consequences of action in light of information regarding the timing of action execution and the state of the world over time. Consequently, there are a number of, related but distinct, perspectives from which this issue has been investigated: logical specification and logic programming, planning, and action languages.

The logical approach to reasoning with action and effect requires an axiomatization of the problem such that it is reduced to that of theorem-proving. Thus, it builds upon the rigour of logic, leverages the efficiency of pre-existing and optimized theorem-proving systems while inheriting the formal semantics of the underlying logic which in turn facilitates correctness proofs.

The situation calculus is a formalization of action and effect in first-order predicate logic. It defines an ontology that includes situations, fluents and actions. Fluents are predicates and functions whose interpretations are situation dependent. Thus, they are relations or functions that contain a single distinguished situation argument which defines their evaluation. Situations have two widely accepted interpretations: as a snapshot of the universe[95]; or as a history of named actions

rooted at a special initial situation[88]. In both cases, the situation calculus implicitly defines time as a branching structure rooted at an initial situation.

When viewing situations as descriptions of the universe at a given instant of time, the axiomatization typically utilizes a single special function *Result* and a special predicate *Holds*. The result function (Result(a, s)) maps the current situation s to the situation resulting from executing the action a in s. The Holds(f, s) predicate denotes that fluent f is true in situation s. *Effect axioms* are then defined using *Holds* and *Result*. Effect axioms have the form $\forall s[\text{Holds}(\beta, \text{Result}(\alpha, s) \leftarrow \Pi]$ or $\forall s[\neg\text{Holds}(\beta, \text{Result}(\alpha, s) \leftarrow \Pi]$ where:

(Π) is a first-order formula that defines additional conditions under which *Holds* attains its truth value. It may be empty but may not refer to situations outside of *Holds* predicates;

(s) is a variable that ranges over situations;

(β) is a formula that holds in the situation resulting from executing action α in s.

This approach utilizes all the expressiveness of first-order logic and thus must utilize techniques to overcome the problems inherent in doing so. These problems are discussed in detail on page 54. The traditional mechanism for doing so is to adopt *Circumscription* which augments first-order logic with a second-order formula that minimizes of the set of objects of which a predicate holds. By carefully selecting the predicates to minimize, this representation facilitates the expression of domain constraints, ramifications, concurrency, non-determinism, continuous change for reasoning both forwards and backwards through time.

When interpreting situations as histories of named actions, the progression of situations is defined according to a distinguished function symbol *do* which maps situations and actions to the situation resulting from executing the action argument in the context of the situation argument[13]. The preconditions defining the applicability of a particular action are defined using *Poss* predicates of the form $\text{Poss}(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s)$ where A is the action in question, s is a variable that represents situations and Π is a first-order formula [14] that defines the conditions under which the action is applicable. The effects of actions are defined using *successor state axioms* which are of the form:

[13]This plays the same role as that of Result.

[14]It may not, however, reference the special function *do*.

(2.2.1) Definition (Relational Fluent Successor State Axioms):

$$F(\vec{x}, do(a,s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s).$$

This axiom form simply states that relational fluent F holds in the situation resulting from doing a in s iff a is one of the actions that cause F to hold, or F was already held to be true and a is an action that does not cause F to become false.

(2.2.2) Definition (Functional Fluent Successor State Axioms):

$$f(\vec{x}, do(a,s)) = y \equiv \gamma_f(\vec{x}, y, a, s) \vee f(\vec{x}, s) = y \wedge \neg(\exists y')\gamma_F(\vec{x}, y', a, s)$$

Similarly, this axiom form states that functional fluent f takes the value y in the situation resulting from executing a in situation s if and only if a is an action that causes f to evaluate to y, or y already evaluated to y in situation s and a is not an action that causes f to take on an evaluation y' that is not y.

This particular style of axiomatization is advantageous in that it avoids many of the difficulties other styles of formalization suffer[15]. It is also succinct, in that an axiomatization using this approach increases in the number of axioms linearly with the number of fluents and actions. However, unlike approaches that utilize non-monotonic consequence relations, domain constraints must be compiled into the successor state axioms instead of standing independently.

The situation as histories of named actions interpretation has been transformed, modulo a number of representational constraints, into logically equivalent Prolog programs. These Prolog programs provide an efficient implementation, using regression from the query to the initial situation, for answering queries of domains specified in this representation. This subset of the situation calculus, in conjunction with a number of procedurally inspired macros, constitutes the dynamic systems programming language GoloG[64] and its decendants[31, 27, 28, 30].

As a consequence of the openness with respect to the truth of formulæ first-order logic facilitates, a number of problems arise in the process of axiomatizing a given situation calculus theory. The first and most widely studied is known as the *frame problem*. In any axiomatization of a dynamic system, not only must the changes between situations as a consequence of executing a particular action be made explicit, but also what does not change. In general it is assumed that change is the exception and therefore any solution to the frame problem must be able to deduce from a specification of what does change all that does not.

[15] In particular, it provides a solution to the frame problem, which will be discussed later.

A similar issue, called the *qualification problem* requires the axiomatization to conclude that the listed preconditions for the applicability of actions are complete. This is required in order to prevent the assumption of additional conditions that invalidate correct conclusions in particular reasoning tasks. Typically, this problem arises when generating explanatory narratives given incomplete information about the occurrence or lack thereof of particular actions.

A third related problem is that of the *ramifications of actions*. This includes the indirect effects of action and their relationship to unchanging constraints of the environment. This issue is particularly acute when ramifications lead to domino like chains of events. Again, this is a minimization problem stemming from the intuition that the occurrence of events is exceptional, the non-occurrence is the norm. Thus, a reasoning procedure should minimize the occurrence of exogenous events as a consequence of domain constraints to include only those exogenous actions that are necessary.

The situation calculus ontology [88], although simple, has been demonstrated to be sufficiently rich as to model domains containing continuous, concurrent, exogenous and stochastic actions with rich ramifications and interactions without modification to the foundational axioms.

The fluent calculus[101] is an extension of the situation calculus that introduces the concept of a *state*. A state is a collection of fluents that hold at a given instant of time collected together using the special function "$\circ$". This conceptualization of state requires extended uniqueness of names assumptions beyond those employed in the standard situation calculus described above. The equality of states is defined according to the equality of the fluents contained therein. Each state may contain multiple instances of a given fluent such that $a \circ a \circ b \neq a \circ b \circ b$. In a later axiomatization[103] this was changed so that $\circ$ was idempotent and consequently operated much like the union ($\cup$) operator of set theory. Each situation is associated with the state of the fluents that hold in that situation using the State(s) function where s is the situation in question. Thus, each relational fluent becomes reified (a Boolean function term) instead of retaining its status as a predicate. The effects of actions are specified according to state update axioms that define the state that holds as a consequence of executing a particular action. These take the form:

(2.2.3) Definition (Universal state update axioms):

$$\Pi(s) \rightarrow \Gamma[State(Do(a,s)), State(s)]$$

where s is the current situation; State$(\mathrm{Do}(a,s))$ is the state resultant from executing action a in state State(s) associated with s; and Γ is a set of conditions that

will hold in state State(Do(a, s)) provided Π holds in the state State(s) associated with s. The advantage of this form of axiomatization is that the number of times state update axioms are referenced in proofs are typically fewer than the number of times successor state axioms are referenced in equivalent formalizations in the traditional situation calculus discussed above. This efficiency increase comes at the expense of richer equality constraints, the cost of which has been offset by efficient equality solvers. Just as the (simplified) situation calculus has been embodied in practical robotic systems via the GoloG (and successors) implementation, so has the (simplified) fluent calculus in the form of FLUX[102, 100]. FLUX differs in that it sacrifices the completeness of its reasoning engine for efficiency. Its design is such that it avoids another of the primary disadvantages of the situation calculus approach in that the agent's representation of the world is progressed with time. This avoids the need to reason back to the initial situation in order to draw conclusions. The efficiency gains of doing so are significant, particularly in scenarios where the tasks required of the agent have non-trivial durations.

The event calculus[96] is another approach to formalizing dynamic systems. It, unlike the situation calculus and its extension the fluent calculus, is a linear rather than a branching time logic. This follows from the choice of representing time explicitly. Also, like the fluent calculus, all fluents are reified and consequently can be quantified over. The basic predicates that define the event calculus are as follows:

$\boxed{\mathsf{Initiates}(\alpha, \beta, \tau)}$: β starts to hold after α is executed at time τ.

$\boxed{\mathsf{Terminates}(\alpha, \beta, \tau)}$: β ceases to hold after α is executed at time τ.

$\boxed{\mathsf{Initially}_T(\beta)}$: β holds initially (at time 0).

$\boxed{\mathsf{Initially}_F(\beta)}$: β does not hold initially (at time 0).

$\boxed{\tau_1 < \tau_2}$: time τ_1 is earlier than τ_2.

$\boxed{\mathsf{Happens}(\alpha, \tau_1, \tau_2)}$: α begins executed at time τ_1 and completes at time τ_2.

$\boxed{\mathsf{HoldsAt}(\beta, \tau)}$: β holds at time τ.

$\boxed{\mathsf{Releases}(\alpha, \beta, \tau)}$: β no longer obeys the law of inertia after α is executed at time τ.

$\boxed{\mathsf{Clipped}(\tau_1, \beta, \tau_2)}$: fluent β is terminated between τ_1 and τ_2.

$\boxed{\mathsf{Declipped}(\tau_1, \beta, \tau_2)}$: fluent β is initiated between τ_1 and τ_2.

$\boxed{\mathsf{Cancels}(\alpha_1, \alpha_2, \beta)}$: when executed concurrently, α_1 cancels the effect β of α_2.

$\boxed{\mathsf{Canceled}(\alpha, \beta, \tau_1, \tau 2)}$: an event occurs during the interval $\tau_1 - \tau_2$ which cancels the β effect of α.

$\boxed{\mathsf{Trajectory}(\beta_1, \tau, \beta_2, \delta)}$: if β_1 becomes true at τ then β holds at $\tau + \delta$.

Using circumscription, as in the circumscribed situation calculus, to minimize the extension of the *Initiates*, *Terminates*, *Happens*, *Releases*, *Trajectory* and *Cancels* predicates allows the event calculus to avoid the traditional issues of logical formalisms of actions: namely the frame and ramification problems. The minimization of *Initiates* and *Terminates* provides completeness for the effects of actions and the minimization of *Happens* provides completeness for the occurrence of events. Minimizing *Releases* ensures that only the required predicates are freed from the common-sense laws of inertia. Minimizing *Cancels* ensures that the interactions between concurrent actions are limited to those stated and minimizing *Trajectory* prevents extraneous events triggered as a consequence of values reaching their specified thresholds.

Planning systems, in contrast to logical methods, abandon the formality of logical specification in favour of custom planning algorithms, heuristics and incomplete search. This enables these systems to scale to problems beyond the capabilities of current theorem-proving mechanisms. The prototypical example of such a system is that of STRIPS[39]. In this approach dynamic systems are specified as a database of facts and a set of operators that can be applied to modify the database. The database constitutes a formalization of the initial state of the domain in a suitable formalism such as first-order logic. Each operator represents an action routine whose execution results in the embodiment of the system taking certain actions in the environment in which it is embedded. Operators are defined as a tuple $\langle N, P, D, A \rangle$ where N is the name of the operator defined as a well-formed formula of the logic underlying the database; P is the precondition for the applicability of the operator, again represented as a formula of the database theory; A is a list of well-formed formula that must be added to the database as a consequence of applying the operator and D is a list of formula to be removed from the theory as a consequence of operator application.

Typically operators are not defined explicitly but are provided as operator schema. The variables in the schema are not logical variables but place holders for constants from the logical theory. These schema variables are substituted through the process of determining the applicability of a particular operator given the current database. If there is an instance of the precondition that logically follows from the database then the operator is applicable with the schema variables replaced uniformly throughout the operator with the corresponding logical constants from the instance derived from the database.

In order to delete all database facts that match a particular formula schema, special variables that can match anything are available in delete lists. These match any variables take the value of all constants from the database such that the database entails the resulting substitution into the well-formed formula to be deleted. So as to ensure that a deleted formula cannot hold via inference within the database subsequent to being deleted, the database is partitioned into primitive and non-primitive predicates. Delete lists are then restricted to these primitive predicates. Each non-primitive predicate is associated with each primitive predicate on which its validity depends. When any of these primitive dependencies are deleted from the database, so are all non-primitive dependants.

The search procedure is not complete, it only applies relevant operators to the current database instead of all of them. A relevant operator is one such that either the add or delete list removes a difference between the current database and the goal clause. Note that an operator schema may produce multiple relevant operators such that the logical variable/schema variable substitutions are distinct. This distance metric between a formula and theory guides the search and avoids the application of many operators that do not move the database towards the goal.

In the same way the logical specifications of actions and effects have been extended to support probabilistic effects, concurrent actions, sense actions and online execution so has the STRIPS formulation and its derivative approaches.

More modern automated planning approaches[48] tend to maintain the STRIPS representation or representations similar to it. They differ primarily in their use of more advanced heuristics. These may be directly applied to the STRIPS style representation or via translations to other formalisms so as to take advantage of the efficient solvers for the target representation. An example of the former approach is the utilization of AND/OR graphs to dramatically constrain the search space in GraphPlan like systems. Examples of the second include translations into Satisfiability (SAT), Constraint Satisfaction (CSP) or Model Checking problems. Typically extensions to the STRIPS representation to facilitate the use of more of the target solvers capabilities follow in cases where the second strategy is adopted.

Action languages[43] are formalizations of causal reasoning that use a natural English language like syntax. Their semantics derive from transition systems that consist of states, a fluent evaluation function that maps fluents and states into values and a set of triples that define the states resultant from executing an action in a state. These triples will be designed such that the traditional problems of the logic approach to action are avoided. These languages are typically separated into *action descriptions languages* and *action query languages*. *Action description languages* define a transition system and the *action query languages* allow assertions about transition systems to be expressed. Provided a transition system and a query, the reasoning task is to find whether the assertion holds in the provided transition system. There are a number of action description languages: $\mathcal{A}, \mathcal{B}, \mathcal{C}$ and a number of action query languages: $\mathcal{P}, \mathcal{Q}_n, \mathcal{R}_n$.

The action description language $\mathcal{A}$ allows transition systems to be built from statements of the form: A **causes** L **if** F. This expresses a dynamic law that states that the action/transition A causes literal L to hold in the resulting state provided the fluent F held in the initial state.

The action description language $\mathcal{B}$ is an extension of $\mathcal{A}$ in that it facilitates the expression of indirect effects of action through the specification of static laws[16] that define constraints on the truth of related fluents/literals in all states. These laws take the form: L **if** F. Laws of this form state that whenever the fluents F hold in a particular state, literal L must hold also[17].

The action description language $\mathcal{C}$ takes an alternative approach to $\mathcal{B}$. It provides additional expressiveness, without being a strict super-set. It facilitates the description of non-deterministic and concurrent actions and the ability to suppress the laws of inertia as necessary. Transaction systems are specified in $\mathcal{C}$ using "**caused** F **if** G" for static laws and "**caused** F **if** G **after** U" for dynamic laws. The static law: "**caused** F **if** G" states that the fluent(s) F are caused whenever the fluent(s) G hold. The dynamic law "**caused** F **if** G **after** U" states that the fluent(s) F hold after the formula U holds of the transition system (which may include actions as well as fluents) provided G holds.

The action query language $\mathcal{P}$ facilitates the posing of queries about the current state of the world, or the effects of sequences of actions (i.e. projection problems). The syntax for doing so is "**now** F" and "**necessarily** F **after** $A_1; \ldots; A_k$". A query of the form: **now** F is satisfied provided the initial state of the transition system satisfies F. Queries pertaining to the truth of a fluent F after a succession of actions

[16] These are equivalent to the domain constraints of the situation calculus.

[17] Notice, this formula makes no mention of specific states or transitions.

$A_1; \ldots; A_k$ are satisfied provided that the states that terminate all paths that follow the action sequence $A_1; \ldots; A_k$ and satisfy the static laws of the transition system all satisfy F.

The action query language $\mathcal{Q}_n$, in contrast to $\mathcal{P}$, is designed for answering queries about a particular narrative (known executed sequence of actions). Queries in $\mathcal{Q}_n$ have two forms: one pertaining to the occurrence of actions at a particular time (A **occurs at** t_i); and, the truth value of a fluent at a particular time (F **holds at** t_i). The n in the name of the language defines the length of histories (number of transitions in the queried transition system) that the answer will be valid for. For alternative values of n the results may vary due to the need for *all* histories of the required length that satisfy the queried transition system to satisfy the posed query also.

$\mathcal{R}_n$ combines the expressive possibilities of both $\mathcal{P}$ and $\mathcal{Q}_n$ by facilitating queries about both hypothetical and actual action sequences. The syntax extends that of $\mathcal{Q}_n$ by introducing the queries of the form: **necessarily** F **after** $A_1; \ldots; A_k$ **at** t_i. The limitation expressed with respect to $\mathcal{Q}_n$ regarding the narrative length applies equally to $\mathcal{R}_n$.

2.2.2 Monitoring

Execution monitoring is a situated agent's process of observing the world, checking for discrepancies between the actual world and its internal representation of it, and recovering from such discrepancies [29]. Successful mechanisms for doing so are heavily dependent on three major factors: the task(s) that is being monitored; the properties of the environment in which the agent is situated; and the architecture of the agent that is conducting the monitoring [6].

Tasks can be classified according to: the nature of the penalty/reward function, be it constant, dynamic or sequentially dependent; the costs of actions and failures; and whether it demands parallel goals and the (in)dependence between them.

Environments can be classed based on: the presence of noise and unpredictability; whether actions are reversible or permanent; the potential for fatal actions; and whether other agents are similarly situated within it.

Architectures can be categorized according to: the accuracy, speed, costs, error profiles and independence of their sensors; effector influence on sensors and other effectors; the actions available to it; and its reaction times/reasoning abilities.

A *monitoring strategy* must encompass details regarding what to monitor, when to monitor it and how to recover from execution difficulties. For many of the models of action and effects discussed in section 2.2, corresponding monitoring strate-

gies have been developed in response to the first and third problems. However, very few of these formalisms have attempted to address the scheduling of monitoring.

There are a small number of generic strategies for deciding when to monitor: monitor for discrepancies after the execution of every action; monitor according to a periodic schedule, typically every reasoning cycle of the agent, or with a dynamic period proportional to the error-proneness of the monitored behaviour [50]; or schedule sensory actions and associated discrepancy detection as an explicit action during plan generation and maintenance. Another approach that has been advocated [71] is the use of an external monitor process whose role is to interrupt the executor or planner in situations in which a discrepancy has been detected. Unfortunately this does not eliminate the problem, only move it elsewhere [6].

An approach to planning that integrates sensing and recovery is that of *rationale-based monitoring*[107]. The distinguishing factor of this approach is that the relevant elements of the plan undergoing generation are monitored throughout the planning process itself. Thus, should an assumption based on the current world state be sensed to be invalid, the planner will modify the current plan (*"add"* or *"cut"* goals) under construction to reflect this or *"jump"* to one of the alternatives that reflects the new state more effectively. The mapping of monitors and changes to the world model to their suggested plan transformations is summarized in Table 2.7. Monitors are generated for both the current plan and the alternatives under consideration. In both cases, monitors are constructed to detect the change of truth values for: the preconditions of sub-goals; usability conditions (conditions that must hold for the plan to be valid yet lie beyond the agent's control); and, quantified conditions (the extensions under universal quantifiers within preconditions). A precondition that holds during planning can be sensed to become false, and conversely, a precondition assumed to need agent control to hold may be sensed to become true without the agent's intervention. Similarly, the valuation of usability conditions can both become true or false in contrast to the agent's current evaluation of the state of the environment. Alternatively, the extension of the individuals to which a universal quantifier refers can expand or contract as a result of sensory information. Detecting such changes can result in additional sub-goals to the current plan, the removal of newly unnecessary sub-goals from the current behaviour or a switch to another alternative as the current optimum solution.

The frequency with which the individual monitors were evaluated against the world model was a constant number of sense-plan transformation/act cycles. The number of monitors generated was minimised as the overall cost of monitoring increased with each additional monitor utilized.

An alternative to developing generic monitoring strategies that can be applied

Monitor Type	World state change	Transformation(s)
Sub-goal	$\top \rightarrow \bot$	add, jump
	$\bot \rightarrow \top$	cut
Usability-condition	$\top \rightarrow \bot$	jump
Quantified-condition	increased extension	add, jump
	decreased extension	cut

(a) Plan-based Monitors

Monitor Type	World state change	Transformation(s)
Sub-goal	$\bot \rightarrow \top$	jump
Usability-condition	$\bot \rightarrow \top$	jump
Quantified-condition	decreased extension	jump

(b) Alternative-based Monitors

Table 2.7: Rationale-based Monitoring and Suggested Plan Transformations

to many tasks is to identify a particular task or task class and develop optimal monitoring strategies for the selected task. This was the methodology utilized in demonstrating the superiority of the interval-reduction methodology for monitoring for deadlines [6]. The drawback of this approach is that the agent must recognize that the monitoring problem it faces belongs to a particular class and must then have a library of monitoring strategies that contains strategies for the class identified.

2.3 Belief Dynamics

Study of the dynamics of belief can be categorized in terms of the structure present in the belief state. Classical belief revision is concerned primarily with sentential belief in which little structure exists relating individual beliefs. In particular, classical belief revision proposes a number of postulates that define of what rational belief change consists. Modal belief revision then extends the findings of classical belief revision to encapsulate belief states with more structure. Consequently, classical belief revision will be presented initially and more deeply before the ex-

tensions afforded by modal belief change are discussed.

2.3.1 Classical Belief Revision

Classical belief revision[41, 51] is the study of belief bases, belief sets and the operations that can transform them over time. In particular, the operations that can modify a belief base provided an unchanging domain. Consequently, belief revision is an operation of belief correction. When changing belief bases to maintain accuracy towards a dynamic domain the operation of update[60], with different semantics, is appropriate.

Belief bases are sets of sentences in some logical language such that each sentence contained therein is explicitly believed by the agent. The operations of change are executed on this set of sentences directly, with the consequences drawn subsequent to the change operations. Belief sets, on the other hand, are logically closed sets of sentences such that there is no semantic differentiation between the sentences held directly and those that follow logically from those. Thus, the operations of change operate on the entire closed set. Although these behave identically in static contexts, once their dynamics are taken into consideration the application of the standard operators of change lead to different outcomes. A belief base ($\mathcal{B}$) can be transformed into the corresponding unique belief set ($\mathcal{B}'$) through the application of the logical consequence (Cn) operator for the logic of beliefs ($\mathcal{B}' = Cn(\mathcal{B})$). Having done so, the belief set ($\mathcal{B}'$) will contain either β or $\neg\beta$ for all formula of the language ($\mathcal{L}_{\mathcal{B}}$) of beliefs.

The primitive operations of belief change are: expansion ($+$) and contraction ($-$). The decomposition principle ensures that these operations are sufficient to derive any additional operators.

(2.3.1) Principle (Decomposition Principle):
Every legitimate belief change is decomposable into a sequence of expansions and contractions.

Expansion corresponds to the integration of new information into the informational state without regard for consistency. Consequently, expansion is typically modelled as set union on the belief base/set ($\mathcal{B}$) with the formula with which to expand (β)[41]:

$$\mathcal{B} + \beta = \mathcal{B} \cup \{\beta\} \tag{2.21}$$

This will succeed irrespective of whether the negation of the formula by which to expand the beliefs are already contained within the belief base ($\neg\beta \in \mathcal{B}$).

The operation of contraction is used for the removal of information. Contraction is a more sophisticated operation than expansion in that it is necessary to not only remove the formula by which the belief base is to be contracted, but also all formulæ that entail it. The challenge this poses is that many sets of formulæ within the belief base may entail the formula to be contracted and a choice must be made between these. To facilitate discussion of contraction operators it is necessary to introduce the concept of a *remainder set* and some syntax by which to refer to them. The remainder set of a belief base (the *progenitor*) and a set of formulæ (the *rejector*) is the set of sets of formulæ such that each set of formulæ is a maximally consistent subset of the progenitor that does not entail any of the formula in the rejector[41]. More formally:

(2.3.1) Definition (Remainder Set[18]):
For any two sets $\mathcal{B}$ (the progenitor) and $\mathcal{B}'$ (the rejector) of sentences, $\mathcal{B} \perp\!\!\!\perp \mathcal{B}'$, $\mathcal{B}$'s remainder set modulo $\mathcal{B}'$ is the set such that $\mathcal{B}'' \in \mathcal{B} \perp\!\!\!\perp \mathcal{B}'$ if and only if:

(1.) $\mathcal{B}'' \subseteq \mathcal{B}$

(2.) $Cn(\mathcal{B}'') \cap \mathcal{B}' = \varnothing$

(3.) *There is no set $\mathcal{B}'''$ such that $\mathcal{B}'' \subset \mathcal{B}''' \subseteq \mathcal{B}$ and $Cn(\mathcal{B}''') \cap \mathcal{B}' = \varnothing$.*

When the rejector is a single belief β, $\mathcal{B} \perp\!\!\!\perp \beta$ will serve as an abbreviation of $\mathcal{B} \perp\!\!\!\perp \{\beta\}$.

Given contraction operators should be minimal (drop only the minimum number of formulæ required to satisfy the conditions of a contraction), it is clear that remainder sets provide a partial answer. As mentioned above, the remaining issue is the rational choice between the elements of the remainder set. There are a number of choices along a spectrum of commitment that can be made. A single option among those generated by the remainder set operation may be selected using a selection function [41].

(2.3.2) Definition (Selection Function):
Let $\mathcal{B}$ be a set of sentences. A selection function *for $\mathcal{B}$ is a function γ such that for all sentences β:*

(1.) *If $\mathcal{B} \perp\!\!\!\perp \beta$ is non-empty, then $\gamma(\mathcal{B} \perp\!\!\!\perp \beta)$ is a non-empty subset of $\mathcal{B} \perp\!\!\!\perp \beta$, and*

(2.) *If $\mathcal{B} \perp\!\!\!\perp \beta$ is empty, then $\gamma(\mathcal{B} \perp\!\!\!\perp \beta) = \{\mathcal{B}\}$*

[18] The non-traditional symbol ($\perp\!\!\!\perp$) has been chosen for remainder sets in preference to the traditional symbol ($\perp$) in order to avoid confusion between belief remainder sets and falsity.

Such a selection leads to *maxi-choice contraction*[41]:

2.3.3 Definition (Maxi-Choice Contraction):

$$\mathcal{B} - \beta = \gamma(\mathcal{B}\perp\!\!\!\perp\beta)$$

A more conservative approach is to take only the formulæ that appear in all elements of the remainder set (*full-meet contraction*)[41]:

2.3.4 Definition (Full-Meet Contraction):

$$\mathcal{B} - \beta = \bigcap(\mathcal{B}\perp\!\!\!\perp\beta)$$

An approach that occupies the middle ground between full-meet and maxi-choice contraction is to assume the selection function can return multiple elements and then take the common elements of these selections (*partial-meet contraction*)[41]:

2.3.5 Definition (Partial-Meet Contraction):
Let $\mathcal{B}$ be a set of sentences and γ a selection function for $\mathcal{B}$. The partial-meet contraction *on $\mathcal{B}$ that is generated by γ is the operation $\sim_\gamma$ such that for any sentence β:*

$$\mathcal{B} \sim_\gamma \beta = \bigcap \gamma(\mathcal{B}\perp\!\!\!\perp\beta)$$

An operation $-$ on $\mathcal{B}$ is a partial-meet contraction if and only if there is a selection function γ for $\mathcal{B}$ such that for all sentences β, $\mathcal{B} - \beta = \mathcal{B} \sim_\gamma \beta$

Should the selection function select a single element of the remainder set then partial-meet contraction degenerates into maxi-choice contraction. Similarly, if the selection function does not eliminate any elements of the remainder set then partial-meet contraction regresses into full-meet contraction.

Given the complexity surrounding the operation of contraction, it is instructive to identify rationality postulates that define the ideal contraction operator [1, 41, 40]. The basic postulates that do so are:

(Success) : If $\beta \notin Cn(\varnothing)$, then $\beta \notin Cn(\mathcal{B} - \beta)$.

(Inclusion) : $\mathcal{B} - \beta \subseteq \mathcal{B}$.

(Failure) : If $\beta \in Cn(\varnothing)$, then $\mathcal{B} - \beta = \mathcal{B}$.

(Vacuity) : If $\beta \notin Cn(\mathcal{B})$, then $\mathcal{B} - \beta = \mathcal{B}$.

(Relevance) : If $\beta \in \mathcal{B}$ and $\beta \notin \mathcal{B} - \beta$, then there is a set $\mathcal{B}'$ such that $\mathcal{B} - \beta \subseteq \mathcal{B}' \subseteq \mathcal{B}$ and that $\beta \notin Cn(\mathcal{B}')$ but $\beta \in Cn(\mathcal{B}' \cup \{\beta\})$.

(Core-retainment) : If $\beta \in \mathcal{B}$ and $\beta \notin \mathcal{B} - \beta$, then there is a set $\mathcal{B}'$ such that $\mathcal{B}' \subseteq \mathcal{B}$ and that $\beta \notin Cn(\mathcal{B}')$ but $\beta \in Cn(\mathcal{B}' \cup \{\beta\})$.

(Closure) : If $\mathcal{B}$ is logically closed, then so is $\mathcal{B} - \beta$ for all β.

(Relative Closure) : $\mathcal{B} \cap Cn(\mathcal{B} - \beta) \subseteq \mathcal{B} - \beta$.

(Extensionality) : If $\beta \leftrightarrow \beta' \in Cn(\varnothing)$, then $\mathcal{B} - \beta = \mathcal{B} - \beta'$.

(Uniformity) : If it holds for all subsets $\mathcal{B}'$ of $\mathcal{B}$ that $\beta \in Cn(\mathcal{B}')$ if and only if $\beta' \in Cn(\mathcal{B}')$, then $\mathcal{B} - \beta = \mathcal{B} - \beta'$.

(Recovery) : $\mathcal{B} \subseteq Cn((\mathcal{B} - \beta) \cup \{\beta\})$.

(Conjunctive inclusion) : If $\beta \notin Cn(\mathcal{B} - (\beta \wedge \beta'))$ then $\mathcal{B} - (\beta \wedge \beta') \subseteq \mathcal{B} - \beta$.

(Conjunctive overlap) : $(\mathcal{B} - \beta) \cap (\mathcal{B} - \beta') \subseteq \mathcal{B} - (\beta \wedge \beta')$.

(Conjunctive factoring) : Either $\mathcal{B} - (\beta \wedge \beta') = \mathcal{B} - \beta$, $\mathcal{B} - (\beta \wedge \beta') = \mathcal{B} - \beta'$ or $\mathcal{B} - (\beta \wedge \beta') = (\mathcal{B} - \beta) \cap (\mathcal{B} - \beta')$.

(Conjunctive trisection) : If $\beta \in \mathcal{B} - (\beta \wedge \beta')$ then $\beta \in \mathcal{B} - (\beta \wedge \beta' \wedge \beta'')$.

(Partial anti-monotony) : $\mathcal{B} - \beta \cap Cn(\{\beta\}) \subseteq \mathcal{B} - (\beta \wedge \beta')$.

(Core identity) : $\beta' \in \mathcal{B} - \beta$ if and only if $\beta' \in \mathcal{B}$ and there is no $\mathcal{B}' \subseteq \mathcal{B}$ such that $\beta \notin Cn(\mathcal{B}')$ and $\beta \in Cn(\mathcal{B}' \cup \{\beta'\})$.

(Meet identity) : If $\beta, \beta' \in \mathcal{B}$, then $(\mathcal{B} - \beta) \cap (\mathcal{B} - \beta') = \mathcal{B} - (\beta \wedge \beta')$.

(Fullness) : If $\beta' \in \mathcal{B}$ and $\beta' \notin \mathcal{B} - \beta$, then $\beta \notin Cn(\mathcal{B} - \beta)$ and $\beta \in Cn((\mathcal{B} - \beta) \cup \{\beta'\})$.

Of these, *closure*, *inclusion*, *vacuity*, *success*, *extensionality*, and *recovery* are considered basic and the minimum for rationality.

There are a number of logical connections between the postulates such that if the contraction operator "–" satisfies [51]:

1. *inclusion* and *core-retainment*, then it satisfies *failure*.
2. *inclusion* and *core-retainment*, then it satisfies *vacuity*.
3. *relevance*, then it satisfies *core-retainment* and *relative closure*.
4. *inclusion* and *relative closure*, then it satisfies *closure*.
5. *uniformity*, then it satisfies *extensionality*.
6. *partial anti-monotony*, then it satisfies *conjunctive trisection*.
7. *extensionality* and *conjunctive trisection*, then it satisfies *partial anti-monotony*.

Additionally, if a contraction operator "–" for a logically closed set $\mathcal{B}$ satisfies[51]:

1. *extensionality* and *vacuity*, then it satisfies *uniformity*.
2. *core-retainment*, then it satisfies *recovery*.
3. *relevance*, then it satisfies *recovery*.
4. *closure*, *inclusion*, *vacuity*, and *recovery*, then it satisfies *relevance*.
5. *closure*, *inclusion*, *vacuity*, *extensionality*, *recovery* and *conjunctive overlap*, then it satisfies *partial anti-monotony*.
6. *closure*, *inclusion*, *recovery*, and *partial anti-monotony*, then it satisfies *conjunctive overlap*.
7. *closure*, *inclusion*, *vacuity*, *extensionality*, and *recovery*, then it satisfies *conjunctive trisection* if and only if it satisfies *conjunctive overlap*.
8. *closure*, *inclusion*, *success*, *failure*, *recovery*, *conjunctive inclusion* and *conjunctive overlap*, then it satisfies *conjunctive factoring*.

(9.) *closure*, *inclusion*, *vacuity*, *extensionality*, *recovery*, and *conjunctive factoring*, then it satisfies *conjunctive inclusion* and *conjunctive overlap*.

The following observations [51] summarize the conformity of partial-meet, full-meet and maxi-choice contractions to the properties of idealized contraction detailed above.

(2.3.1) Observation:
Partial-meet contraction satisfies: success, inclusion, relevance, *and* uniformity. *Because it satisfies* relevance, *it also satisfies* relative closure *and* core-retainment. *Because it satisfies* inclusion *and* core-retainment, *it also satisfies* failure *and* vacuity. *Because it satisfies* inclusion *and* relative closure, *it also satisfies* closure. *Because it satisfies* uniformity, *it also satisfies* extensionality.

(2.3.2) Observation:
Partial-meet contraction of logically closed set $\mathcal{B}$ satisfies: closure, inclusion, vacuity, success, extensionality *and* recovery. *Because it satisfies* closure, inclusion, vacuity, *and* recovery, *it satisfies* relevance. *Because it satisfies* extensionality *and* vacuity, *it also satisfies* uniformity.

(2.3.3) Observation:
Full-meet contraction satisfies: closure, inclusion, vacuity, success, extensionality, recovery, *and* meet-identity. *Because it satisfies* extensionality *and* vacuity, *it also satisfies* uniformity. *Because it satisfies* closure, inclusion, vacuity *and* recovery, *it also satisfies* relevance.

(2.3.4) Observation:
Maxi-choice contraction satisfies: success, inclusion, fullness, *and* uniformity. *Because it satisfies* uniformity, *it also satisfies* extensionality.

In order to further ensure the rationality of partial-meet contractions, it is possible to impose constraints on the selection function. One such constraint is to require the selection function to be *relational*[51].

(2.3.6) Definition (Relationality):
A selection function γ for a set $\mathcal{B}$ is relational *if and only if there is a relation $\sqsubseteq$ such that for all sentences β, if $\mathcal{B}\perp\!\!\!\perp\beta$ is non-empty, then*

$$\gamma(\mathcal{B}\perp\!\!\!\perp\beta) = \{\mathcal{B}' \in \mathcal{B}\perp\!\!\!\perp\beta \mid \mathcal{B}'' \sqsubseteq \mathcal{B}' \text{ for all } \mathcal{B}'' \in \mathcal{B}\perp\!\!\!\perp\beta\}$$

This constraint is sufficient to ensure the resulting partial-meet contraction operator satisfies conjunctive overlap.

For a relation, on which a relational selection function is based, to satisfy the rationality constraints of a preference relation, it is necessary for it to satisfy *transitivity* [51].

(2.3.7) Definition (Transitivity):

If $\mathcal{B} \sqsubseteq \mathcal{B}'$ *and* $\mathcal{B}' \sqsubseteq \mathcal{B}''$ *then* $\mathcal{B} \sqsubseteq \mathcal{B}''$

To ensure that a relation, on which a relational selection function is based, is compatible with the intuition of minimal change, it must be *maximizing*[51].

(2.3.8) Definition (Maximizing Property):

If $\mathcal{B} \sqsubset \mathcal{B}'$, *then* $\mathcal{B} \sqsubset \mathcal{B}'$

Partial-meet contractions based on *transitively* and *maximizing relational* selection functions additionally satisfy[51]:

(1.) conjunctive overlap, and

(2.) conjunctive inclusion and conjunctive factoring provided the set of sentences $\mathcal{B}$ to which it will be applied is finite and disjunctively closed[19], and

(3.) conjunctive inclusion, conjunctive factoring and conjunctive trisection provided the set of sentences $\mathcal{B}$ to which it will be applied is logically closed.

Problems arise when iterated contraction is considered when utilizing a partial-meet contraction operator. These problems stem from the definition of the selection function required to construct the contraction operator. As per definition 2.3.2, each selection function is dependent on the belief base $\mathcal{B}$ to which it will be applied[20]. Because the belief base resulting from a contraction will be, under normal conditions, different from the original belief base, the original selection function is not applicable to it and thus the contraction operator constructed from it is not either. To remedy this it is necessary to introduce *unified* selection functions from which to construct *unified* contraction operations [51].

[19] A set of sentences $\mathcal{B}$ is *disjunctively closed* (closed under disjunction) if and only if for all sentences β and β': if $\beta \in \mathcal{B}$ and $\beta' \in \mathcal{B}$, then $\beta \vee \beta' \in \mathcal{B}$.

[20] See the second condition which mentions $\mathcal{B}$ explicitly.

2.3.9 Definition (Unified Selection Function):
A selection function γ is unified *if and only if for all subsets $\mathcal{B}$ and $\mathcal{B}'$ of $\mathcal{L}_\mathcal{B}$ and all elements β and β' of $\mathcal{L}_\mathcal{B}$: If $\mathcal{B} \perp\!\!\!\perp \beta = \mathcal{B}' \perp\!\!\!\perp \beta' \neq \varnothing$, then $\bigcap \gamma(\mathcal{B}, \mathcal{B} \perp\!\!\!\perp \beta) = \bigcap \gamma(\mathcal{B}', \mathcal{B}' \perp\!\!\!\perp \beta')$*[21].

2.3.10 Definition (Perfectly Unified Selection Function):
A selection function γ is perfectly unified *if and only if for all subsets $\mathcal{B}$ and $\mathcal{B}'$ of $\mathcal{L}_\mathcal{B}$ and all elements β and β' of $\mathcal{L}_\mathcal{B}$: if $\mathcal{B} \perp\!\!\!\perp \beta = \mathcal{B}' \perp\!\!\!\perp \beta' \neq \varnothing$, then $\gamma(\mathcal{B}, \mathcal{B} \perp\!\!\!\perp \beta) = \gamma(\mathcal{B}', \mathcal{B}' \perp\!\!\!\perp \beta')$.*

Given suitable definitions for expansion and partial-meet contraction ($\sim$) based on a selection function (γ) it is trivial, by way of the *Levi identity* or the *reversed Levi identity*, to define a corresponding *internal* revision ($\mp_\gamma$) or *external* revision ($\pm_\gamma$) operator. As can be seen in definition 2.3.11, an internal revision refers to a revision constructed by contracting prior to extending the belief base. Similarly, definition 2.3.12 demonstrates an external revision which is defined such that the expansion is applied before contraction.

2.3.11 Definition (Levi Identity[22]):

$$\mathcal{B} \mp_\gamma \beta = (\mathcal{B} \sim_\gamma \neg\beta) + \beta$$

2.3.12 Definition (Reversed Levi Identity[23]):

$$\mathcal{B} \pm_\gamma \beta = (\mathcal{B} + \beta) \sim_\gamma \neg\beta$$

Conversely, it is possible to generate a contraction operator from a revision operator by way of the *Harper identity*.

2.3.13 Definition (Harper Identity):
Let $\mathcal{B}$ be any set and γ a selection function for $\mathcal{B}$. Then $\mathcal{B} \sim_\gamma \beta = \mathcal{B} \cap (\mathcal{B} \mp_\gamma \neg\beta)$ where $\sim_\gamma$ is the partial-meet contraction operator constructed from selection function γ and $\mp_\gamma$ is an internal partial-meet revision operator constructed from γ.

[21] Each selection function accepts two arguments: the belief base to return in the case the input remainder set is empty, and the remainder set from which a selection is to be made.

[22] The general Levi identity makes no reference to the type of contraction required: $\mathcal{B} * \beta = (\mathcal{B} - \neg\beta) + \beta$.

[23] The general reversed Levi identity makes no reference to the type of contraction required: $\mathcal{B} * \beta = (\mathcal{B} + \beta) - \neg\beta$.

Revision "$*$" is an operation of change that is consistency preserving. Unless the original belief base is inconsistent, the consistency of the original belief base will be preserved by a revision operation. The same cannot be claimed for a primitive expansion operator.

In the same manner it was possible to postulate the properties of ideal contractions, so to it is possible to postulate guiding principles for idealized revisions of consistent belief bases $\mathcal{B}$[41].

(Success) : $\beta \in \mathcal{B} * \beta$.

(Consistency) : $\mathcal{B} * \beta$ is consistent if β is consistent.

(Inclusion) : $\mathcal{B} * \beta \subseteq \mathcal{B} \cup \{\beta\}$.

(Relevance) : If $\beta' \in \mathcal{B}$ and $\beta' \notin \mathcal{B} * \beta$, then there is some $\mathcal{B}'$ such that $\mathcal{B} * \beta \subseteq \mathcal{B}' \subseteq \mathcal{B} \cup \{\beta\}$, $\mathcal{B}'$ is consistent but $\mathcal{B}' \cup \{\beta'\}$ is inconsistent.

(Inconsistent Expansion) : If $\neg\beta \in Cn(\varnothing)$ then $\mathcal{B} * \beta = \mathcal{B} \cup \{\beta\}$[24].

(Vacuity) : If $\neg\beta \notin Cn(\mathcal{B})$, then $\mathcal{B} * \beta = \mathcal{B} \cup \{\beta\}$.

(Uniformity) : If for all $\mathcal{B}' \subseteq \mathcal{B}$, $\mathcal{B}' \cup \{\beta\}$ is inconsistent if and only if $\mathcal{B}' \cup \{\beta'\}$ is inconsistent, then $\mathcal{B} \cap (\mathcal{B} * \beta) = \mathcal{B} \cap (\mathcal{B} * \mathcal{B}')$.

(Weak uniformity) : If β and β' are elements of $\mathcal{B}$ and it holds for all $\mathcal{B}' \subseteq \mathcal{B}$ that $\mathcal{B}' \cup \{\beta\}$ is inconsistent if and only if $\mathcal{B}' \cup \{\beta'\}$ is inconsistent, then $\mathcal{B} \cap (\mathcal{B} * \beta) = \mathcal{B} \cap (\mathcal{B} * \beta')$.

(Pre-expansion) : $\mathcal{B} + \beta * \beta = \mathcal{B} * \beta$.

(Post-expansion) : $\mathcal{B} * \beta + \beta = \mathcal{B} * \beta$.

(Redundancy) : If β is consistent, and $\neg\beta \in Cn(\beta')$ for each $\beta' \in \mathcal{B}'$, then $\mathcal{B} * \beta = (\mathcal{B} \cup \mathcal{B}') * \beta$.

(Tenacity) : For all $\beta' \in \mathcal{B}$, either $\beta' \in \mathcal{B} * \beta$ or $\neg\beta' \in Cn(\mathcal{B} * \beta)$.

[24] $Cn(\varnothing)$ represents the set of tautologies, i.e. the sentences that can be concluded without any premises.

(Inertness) : If $\beta' \in \mathcal{B} * \beta$, then it holds for all sets $\mathcal{B}'$ that if $\beta \in \mathcal{B}' \subseteq \mathcal{B} \cup \{\beta\}$, then $\mathcal{B}$ is consistent if and only if $\mathcal{B}' \cup \{\beta'\}$ is consistent.

(Superexpansion) : $\mathcal{B} * (\beta \wedge \beta') \subseteq (\mathcal{B} * \beta) + \beta'$.

(Subexpansion) : If $\neg\beta' \notin Cn(\mathcal{B} * \beta)$, then $(\mathcal{B} * \beta) + \beta' \subseteq \mathcal{B} * (\beta \wedge \beta')$.

(Disjunctive overlap) : $(\mathcal{B} * \beta) \cap (\mathcal{B} * \beta') \subseteq \mathcal{B} * (\beta \vee \beta')$.

(Disjunctive inclusion) : If $\neg\beta \notin Cn(\mathcal{B} * (\beta \vee \beta'))$, then $\mathcal{B} * (\beta \vee \beta') \subseteq \mathcal{B} * \beta$.

(Disjunctive factoring) : Either $\mathcal{B} * (\beta \vee \beta') = \mathcal{B} * \beta$, $\mathcal{B} * (\beta \vee \beta') = \mathcal{B} * \beta'$, or $\mathcal{B} * (\beta \vee \beta') = (\mathcal{B} * \beta) \cap (\mathcal{B} * \beta')$.

(Idempotence) : If $\beta \in \mathcal{B}$, then $\mathcal{B} * \beta = \mathcal{B}$.

(Monotonicity) : If $\mathcal{B} \subseteq \mathcal{B}'$, then $\mathcal{B} * \beta \subseteq \mathcal{B}' * \beta$.

(Cut) : If $\beta' \in \mathcal{B} * \beta$, then $\mathcal{B} * (\beta \wedge \beta') \subseteq \mathcal{B} * \beta$.

(Cautious monotony) : If $\beta' \in \mathcal{B} * \beta$, then $\mathcal{B} * \beta \subseteq \mathcal{B} * (\beta \wedge \beta')$.

(Reciprocity) : $\mathcal{B} * \beta = \mathcal{B} * \beta'$ if and only if $\beta' \in \mathcal{B} * \beta$ and $\beta \in \mathcal{B} * \beta'$.

Of these *closure*, *success*, *inclusion*, *vacuity*, *consistency* and *extensionality* are considered basic and the minimum to meet rationality requirements.

There are a number of logical connections between the postulates such that if $*$ satisfies:

1. *success*, *inclusion*, and *relevance*, then it satisfies *vacuity*.
2. *success*, *inclusion*, and *relevance*, then it satisfies *inconsistent expansion*.
3. *post-expansion*, it does so if and only if it satisfies *success*.
4. *success*, *inclusion*, and *inertness*, then it satisfies *consistency*.
5. *inclusion*, *consistency*, *inconsistent expansion*, and *tenacity*, then it satisfies *relevance*.

Additionally, if an operator $*$ for a logically closed set $\mathcal{B}$ satisfies:

(1.) *extensionality*, *closure*, and *success*, then it satisfies *disjunctive overlap* if and only if it satisfies *superexpansion*.

(2.) *extensionality*, *closure*, and *success*, then it satisfies *disjunctive inclusion* if an only if it satisfies *subexpansion*.

(3.) *extensionality*, *closure*, *success*, and *consistency*, then it satisfies *disjunctive factoring* if an only if it satisfies both *disjunctive overlap* and *disjunctive inclusion*.

(4.) *consistency*, *closure*, *success*, and *subexpansion*, then it satisfies *cautious monotony*.

(5.) *closure* and *success*, then it satisfies *reciprocity* if and only if it satisfies both *cut* and *cautious monotony*.

The postulates of idealized revision are related to the operations of revision discussed thus far according to the following observations:

(2.3.5) Observation:
Internal partial-meet revision for a belief base $\mathcal{B}$ satisfies: consistency, inclusion, relevance, success *and* uniformity. *Because it satisfies* success, inclusion, *and* relevance *it also satisfies* vacuity *and* inconsistent expansion. *Because it satisfies* success *it also satisfies* post-expansion.

(2.3.6) Observation:
External partial-meet revision satisfies: consistency, inclusion, relevance, success, weak uniformity, *and* pre-expansion. *Because it satisfies* success, inclusion, *and* relevance *it also satisfies* vacuity *and* inconsistent expansion. *Because it satisfies* success *it also satisfies* post-expansion.

(2.3.7) Observation:
Unified *internal partial-meet revision for a belief base $\mathcal{B}$ satisfies:* consistency, inclusion, relevance, success, uniformity, *and* redundancy. *Because it satisfies* success, inclusion, *and* relevance *it also satisfies* vacuity *and* inconsistent expansion. *Because it satisfies* success *it also satisfies* post-expansion.

(2.3.8) Observation:
Unified *external partial-meet revision satisfies:* consistency, inclusion, relevance, success, weak uniformity, pre-expansion, *and* redundancy. *Because it satisfies* success, inclusion, *and* relevance, *it also satisfies* vacuity *and* inconsistent expansion. *Because it satisfies* success, *it also satisfies* post-expansion.

(2.3.9) Observation:
Internal maxi-choice *revision for a belief base $\mathcal{B}$ satisfies:* consistency, inclusion, success, inconsistent expansion, uniformity, *and* tenacity. *Because it satisfies* inclusion, consistency, inconsistent expansion, *and* tenacity, *it also satisfies* relevance. *Because it satisfies* success, inclusion, *and* relevance, *it also satisfies* vacuity. *Because it satisfies* success *it also satisfies* post-expansion.

(2.3.10) Observation:
Internal full-meet *revision for a belief base $\mathcal{B}$ satisfies* inclusion, relevance, success, uniformity, *and* inertness. *Because it satisfies* success, inclusion, *and* relevance, *it also satisfies* vacuity *and* inconsistent expansion. *Because it satisfies* success *it also satisfies* post-expansion. *Because it satisfies* success, inclusion, *and* inertness, *it also satisfies* consistency.

(2.3.11) Observation:
Partial-meet revision of logically closed set $\mathcal{B}$ satisfies: closure, success, inclusion, vacuity, consistency, *and* extensionality. *Because it satisfies* success, *it also satisfies* post-expansion.

(2.3.12) Observation:
Transitively relational partial-meet revision *of a logically closed set $\mathcal{B}$ satisfies:* closure, success, inclusion, vacuity, consistency, extensionality, superexpansion, *and* subexpansion. *Because it satisfies* success, *it also satisfies* post-expansion. *Because it satisfies* extensionality, closure, success, *and* superexpansion, *it also satisfies* disjunctive overlap. *Because it satisfies* extensionality, closure, success, *and* subexpansion, *it also satisfies* disjunctive inclusion. *Because it satisfies* extensionality, closure, success, consistency, disjunctive overlap, *and* disjunctive inclusion, *it also satisfies* disjunctive factoring. *Because it satisfies* consistency, closure, success, *and* subexpansion, *it also satisfies* cautious monotony.

For revision operators constructed from expansion and contraction by way of the *(reversed) Levi identity*, the satisfaction of the postulates for idealized revision follows from the satisfaction of the postulates from idealized contraction according to the following observations[51]:

(2.3.13) Observation:
Let $\mathcal{B}$ be a logically closed set and $-$ an operator for $\mathcal{B}$ that satisfies the contraction-postulates inclusion, vacuity, success, *and* extensionality. *Then $*$ is a revision operator for $\mathcal{B}$ constructed from $-$ that satisfies the revision-postulates* closure, success, inclusion, vacuity, consistency, *and* extensionality.

Conversely:

(2.3.14) Observation:
Let $\mathcal{B}$ be a logically closed set and $*$ an operator for $\mathcal{B}$ that satisfies the revision-postulates closure, success, vacuity, consistency, *and* extensionality. *Then – is a contraction operator for $\mathcal{B}$ constructed from $*$ that satisfies the contraction-postulates* closure, inclusion, vacuity, success, extensionality, *and* recovery.

The correction of belief is a central operation of an agent in a complex environment in which the accuracy of its perception cannot be guaranteed. Operations of correction have been discussed at both the knowledge and symbolic level[72]. Such operations were discussed in an idealized fashion by describing properties that perfect operators of change will satisfy. The requirements of actual operators for satisfying these postulates were discussed, as was the extent to which the operators presented satisfy these requirements. It was shown that correction is decomposable into operations of expansion and contraction. The relationships between contractions and revisions presented and correspondences between the rationality of each when used in defining the other outlined.

2.3.2 Modal Belief Revision

When the domain of an agent's belief extends beyond the sentences it holds true to a set of worlds that it considers possible, the intuitions pertaining to rational change differ. As discussed in section 2.1.2, modal logic is a formal logic that holds possible worlds and relations between these as its semantics. Thus, as was the case for BDI logic, an agent will hold a belief at a given world in all sentences it considers to hold at all worlds regarded as possible from that world. This captures uncertainty[25] by way of multiple worlds, each holding different facets true of the environment. The more worlds considered possible from a given world, the more uncertainty is held in the state of affairs at that world[106, page 55]. Thus, when modelling an expansion in a modal context, the required operation is to restrict the set of possible worlds (accessible to each world) to those where the formula by which the expansion is being conducted holds. Formally, this operation is known as relativisation (adapted from [106, Definition 3.28]).

(2.3.14) Definition (Relativisation):
Given a countable set of propositions $\mathcal{B}$, a set of states S, an accessibility relation between states R*, a valuation function $V^{\mathcal{B}} : \beta \to 2^S$ that for every believable proposition*

[25] In addition to other modalities of truth, as discussed in section 2.1.2.

$\beta \in \mathcal{B}$ yields the set of states in which β is true, and a state ω, the relativisation *of* R *by β' to the set of worlds where β' holds ($\{\omega \mid \beta' \in Cn(\omega)\}$) yields a world $\omega' = \omega$ and accessibility relation* R'*, where* $\mathrm{R}' = \mathrm{R} \cap (S \times \{\omega \mid \beta' \in Cn(\omega)\})$.

As in the classical case, contraction is a more difficult operator to define. Intuition indicates that contraction corresponds to the addition of worlds to the accessibility relation. The difficulty arises in the selection of the worlds to add. In order to contract β from a given world ω it is sufficient to ensure that at least one world accessible from ω satisfies $\neg\beta$. However, there may be multiple worlds that satisfy this condition and to satisfy the principle of *minimal change* the world which differs least must be selected (adapted from [106, Definition 3.30]).

(2.3.15) Definition (Modal Contraction):
Given a countable set of propositions $\mathcal{B}$, a set of states S, an accessibility relation between states R*, a valuation function $V^{\mathcal{B}} : \beta \to 2^S$ that for every believable proposition $\beta \in \mathcal{B}$ yields the set of states in which β is true, a state ω, and a system of spheres $\circledcirc = \{\odot \mid \odot \supseteq \{\omega' \mid \omega \mathrm{R} \omega'\}\}$ that are linearly ordered by $\supseteq$, a* semantic contraction *of* R *by β' yields a world $\omega = \omega'$ and accessibility relation* R'*, where* $\mathrm{R}' = \min\{\odot \in \circledcirc \mid \odot \cap \{\omega'' \mid \beta' \in Cn(\omega'') \neq \varnothing\}\}$ *and for all $\omega''' \neq \omega$, $\{\omega''' \mathrm{R}' \omega''''\} = \{\omega''' \mathrm{R} \omega''''\}$.*

The intuition behind modal contraction is that there is a set of spheres, where each sphere is a set of worlds, of increasing size from the set of worlds currently related to the current world $\{\omega' \mid \omega \mathrm{R} \omega'\}$ to the set of all worlds S. As the number of worlds in each sphere increases so does the uncertainty about the true state of the world and the number of sentences that differ in truth value from the current world. Because each world assigns a truth value to every sentence, eventually there will be a sphere that contains a world that satisfies the negation of the formula by which the beliefs are to be contracted. Once such a world is related to the current world, the formula will no longer be believed. Thus, by extending the current accessibility relation to relate the worlds in this minimal sphere that contains a world with such a negation the contraction can be made. Utilizing the *Levi identity* a corresponding revision operator can be derived.

Although the above procedure does generate a revision operator for modal logic, the result is not without issue. The addition of introspective capabilities, namely beliefs about belief, introduces a number of incompatibilities with the intuitions surrounding belief change as presented thus far.

(2.3.1) Example:
*Now [*sic*] suppose we have two of such belief sets $\mathcal{B}$ and $\mathcal{B}'$ [each closed under* ***KD45****-*

*consequences], and suppose that $\mathcal{B} \subset \mathcal{B}'$. Then there is some β in $\mathcal{B}'$ that is not in $\mathcal{B}$. By positive introspection, we have $(\mathsf{Bel}\ \beta) \in \mathcal{B}'$ and, by negative introspection, $\neg(\mathsf{Bel}\ \beta) \in \mathcal{B}$. Since $\mathcal{B} \subset \mathcal{B}'$, we also have $\neg(\mathsf{Bel}\ \beta) \in \mathcal{B}'$. Belief sets are closed under propositional logical consequence, so that we conclude $\bot \in \mathcal{B}'$, which contradicts the fact that beliefs satisfy axiom **D**, which says that beliefs are consistent! .*

(van Ditmarsch, van der Hoek, and Kooi [106, Page 59])

2.3.2 Example:

Suppose $\mathcal{B}$ is to be revised by β ($\beta \in \mathcal{L}$) such that $\neg\beta \notin Cn(\varnothing)$[26] *so that by the* consistency *postulate $\mathcal{B} * \beta$ is consistent. Suppose, also, that both $\neg(\mathsf{Bel}\ \beta) \in \mathcal{B}$ and $\neg(\mathsf{Bel}\ \neg\beta) \in \mathcal{B}$. By the* success *postulate and positive introspection, $(\mathsf{Bel}\ \beta) \in \mathcal{B} * \beta$. By* vacuity *$\neg(\mathsf{Bel}\ \beta) \in \mathcal{B} * \beta$ leading to $\mathcal{B} * \beta$ being inconsistent.*

(adapted from van Ditmarsch, van der Hoek, and Kooi [106, Page 59])

2.3.3 Example:

*Let β be a **KD45**-consistent formula, that is not KD45-valid. $\beta \wedge \neg(\mathsf{Bel}\ \beta)$ is thus satisfiable. Suppose that $\mathcal{B}$ is to be revised by $\beta \wedge \neg(\mathsf{Bel}\ \beta)$. By*[27]

$$(\neg(\mathsf{Bel}\ \bot) \rightarrow \neg(\mathsf{Bel}\ \phi \wedge \neg(\mathsf{Bel}\ \phi)))$$

it is the case that $\beta \wedge \neg(\mathsf{Bel}\ \beta) \notin \mathcal{B}$. Thus, through the application of vacuity

$$\mathcal{B} * (\beta \wedge \neg(\mathsf{Bel}\ \beta) = \mathcal{B} + (\beta \wedge \neg(\mathsf{Bel}\ \beta))$$

By success *$(\mathsf{Bel}\ \beta \wedge \neg(\mathsf{Bel}\ \beta)) \in \mathcal{B}$. Because $\neg(\mathsf{Bel}\ \beta \wedge \neg(\mathsf{Bel}\ \beta))$ is derivable in **KD45**, this demonstrates how revising a consistent belief set with a consistent formula can lead to an inconsistent belief set.*

(adapted from van Ditmarsch, van der Hoek, and Kooi [106, Page 60])

Formal study of belief revision in modal contexts, extending beyond the classical intuitions presented here, takes place within dynamic doxastic logic[94]. This formalism introduces modalities (normal modal operators) for the expansion, contraction and revision operators by which the dynamics of belief can be discussed. In this way the operators of change are introduced into the object language of the logic. Thus the semantics of change is integrated into the logic and the consequences of axioms proposed detailing the properties of the change operators analysed.

2.4 Research Questions

While the progress on the many fronts presented above is extensive, many areas remain unexplored. There remains divergences between the conditions of rational

[26] Again, note that $Cn(\varnothing)$ represents the set of tautologies.

[27] This formula is known as Moore's principle.

intentionality as argued for in the planing theory of intention and those captured in the BDI logic developed. One such example condition is that of "no regret" (see condition 2.1.1). Thus, there remains work on updating BDI logic to capture the more recent philosophical developments. Additionally, there remains unanswered questions pertaining to the revision of BDI theories, its relationship to the classical postulates of theory change and the applicability of these postulates to the revision of desires and intentions. Also, there remains a wide disparity between the practical reasoning advocated in the BDI logic and that implemented in AgentSpeak. The content of the mental attitudes utilized in the AgentSpeak approach are so restricted as to make equivalences with their logical counterparts difficult to justify. Consequently, work remains in finding efficient yet more expressive representations of the mental attitudes within AgentSpeak systems so as to render these equivalences clear.

While the above avenues of future work are rich with challenge, another theme is obvious from the above presentation. The integration of maintenance and effect conditions of behaviours into the practical reasoning of intentional systems is surprisingly absent. Given the clear reference to such a need in the philosophy of practical reasoning, the lack of equivalent notions in the logic and implementation remains a fundamental issue to be addressed. Of particular interest is the application of the technologies for the reasoning with actions and effects into the richer context afforded by the BDI approach. Given such facilities, their interaction with the dynamics of the mental attitudes to which they pertain offers significant scope for contribution. In particular, a number of questions are raised:

1. what additional reasoning facilities are necessary when the validity requirements and outcome circumstances of behaviours are known?

2. what effect does knowing the conditions under which behaviours are valid, successful or failed have on the way beliefs are managed?

3. what additional reasoning opportunities towards the management of intentions arise when the validity, success or failure conditions of behaviour are known?

These questions lie in the intersection of the topics discussed as the background of the contributions this thesis attempts to make. The integration of representations and reasoning procedures with actions into the BDI methodology requires an in depth understanding of the problems such reasoning involves and sufficient insight into practical reasoning to know which of these issues are pertinent, how

they can be avoided and which techniques are most appropriate. It also needs sufficient familiarity with an appropriate embodiment of the practical reasoning ethos into which these representations and reasoning techniques may be integrated. Similarly, understanding how knowledge pertaining to the validity, success or failure of behaviour may influence the revision of belief requires both insight into the role beliefs play in such behavioural knowledge and the necessities of rational belief change. Crucial also is an awareness of how such an integration can benefit either the reasoning or behaviour of the agent and the trade-offs inherent in balancing of the two. Lastly, understanding the interactions of intentions over time necessitates an appreciation of the guiding principles of the planning theory of practical reasoning in addition to recognition of the practical embodiments of intention.

These questions lead directly to the hypotheses under investigation in this thesis. The particular additional reasoning problem necessitated by knowledge of behaviour validity and outcomes that will be the focus of the early chapters of this thesis is that of monitoring these conditions to ensure the validity of the behaviours under execution and to detect their completion. These conditions will then be utilized as a guiding mechanism for the selection of rational belief revisions. In so doing, this will provide a direct example of a technique by which the management of the beliefs of an agent is influenced by the intentions held and the behavioural knowledge they contain. By utilizing the validity and outcome conditions pertaining to intentions to ensure the consistency of the intentions concurrently held by an agent, an understanding of intention dynamics and dependencies is facilitated. These dependencies then provide the information necessary to propagate intention change and deduce the commitment held in given intentions.

2.5 Summary

This chapter provides the context in which the proposals to follow were developed and against which they should be compared. This background encompasses intention, desires and belief, three elements crucial to the practical reasoning of intelligent agents. The discussion of intention was divided into: the theoretical and philosophical elements that shape our intuitions of intention, practical reasoning, and bounded rationality; the embodiment of these ideas into formal mathematical logic; and finally, the techniques by which they are realized in practice. The discussion of the reasoning and representation approaches developed towards the knowledge of effects and causality followed, including the problems identified and progress towards their resolution. Finally, the state of the art in the dynamics of

belief was presented. This presentation was from both the perspective of unstructured belief sets and bases in a classical setting and more structured beliefs states enabled by modal approaches. While no claims of exhaustivity can be made, it is hoped that the above presentation is sufficient to ensure that the fundamental techniques, problems and achievements of each topic were introduced. Having done so, a number of outstanding issues were identified. Of these, a clear theme emerged regarding the role of behavioural knowledge in the wider practical reasoning of intelligent agents. Within this theme, a number of pressing questions arose and it was shown how these relate to the hypotheses under investigation in this thesis.

Luck, that's when preparation and opportunity meet.

P. E. Trudeau

3 Representation

In contrast to the representations presented in section 2.2, the primary purpose of the representation of effects presented in this chapter is the monitoring and maintenance of the validity of the behaviours to which the effects pertain. To this end, the representation will be concerned with capturing two distinct constraints on the validity of behaviour. Firstly, means for the specification of conditions that must hold throughout the entire, or set period of, the execution of behaviour are necessary. Secondly, conditions that indicate the realization of the effects associated with the behaviour, and conditions that indicate the failure of the behaviour to produce its listed effects must be specifiable. In particular, it will be necessary to motivate the representation as fit for purpose for each of the hypotheses outlined in chapter 1.

3.1 Maintenance Conditions

Maintenance conditions provide checks on behaviours to ensure their correct execution. They facilitate the early detection of (potential) failure and consequently opportunity for early correction or adaptation. In agents of limited resources such early detection and salvage of allocated resources can be the distinguishing factor between the success and failure of its wider intentions.

3.1.1 Example:
Tom is an intentional agent and money is tight. He has a large mortgage and an investment account. He believes that if interest rates rise, then his savings will increase, but he will also have to pay more on his mortgage. His beliefs are as follows:

$$\mathcal{B} = \left\{ \begin{array}{c} \neg \text{`}\textit{Savings increase}\text{'}, \neg \text{`}\textit{Mortgage increase}\text{'}, \\ \text{`}\textit{Interest rate rise}\text{'} \rightarrow \text{`}\textit{Savings increase}\text{'}, \\ \text{`}\textit{Interest rate rise}\text{'} \rightarrow \text{`}\textit{Mortgage increase}\text{'} \end{array} \right\}$$

Tom intends to pay only the minimum back on his mortgage necessary:

$$\mathcal{I} = \{[\text{`}\textit{Minimal repayments}\text{'} : \texttt{while}\,(\neg\,\text{`}\textit{Mortgage increase}\text{'})]\}$$

It is announced that interest rates have been increased. He must now integrate this into his beliefs ($\mathcal{B} * \text{`}\textit{Interest rate rise}\text{'}$). *He has a number of ways of doing so:*

$$\mathcal{B}_1 = \left\{ \begin{array}{c} \text{`}\textit{Interest rate rise}\text{'}, \neg\,\text{`}\textit{Mortgage increase}\text{'}, \\ \text{`}\textit{Interest rate rise}\text{'} \rightarrow \text{`}\textit{Savings increase}\text{'} \end{array} \right\}$$

$$\mathcal{B}_2 = \left\{ \begin{array}{c} \text{`}\textit{Interest rate rise}\text{'}, \neg\,\text{`}\textit{Savings increase}\text{'}, \\ \text{`}\textit{Interest rate rise}\text{'} \rightarrow \text{`}\textit{Mortgage increase}\text{'} \end{array} \right\}$$

$$\mathcal{B}_3 = \left\{ \begin{array}{c} \text{`}\textit{Interest rate rise}\text{'} \rightarrow \text{`}\textit{Mortgage increase}\text{'}, \\ \text{`}\textit{Interest rate rise}\text{'} \rightarrow \text{`}\textit{Savings increase}\text{'}, \\ \text{`}\textit{Interest rate rise}\text{'} \end{array} \right\}$$

$$\mathcal{B}_4 = \left\{ \begin{array}{c} \text{`}\textit{Interest rate rise}\text{'}, \neg\,\text{`}\textit{Savings increase}\text{'}, \\ \neg\,\text{`}\textit{Mortgage increase}\text{'} \end{array} \right\}$$

We can see that $\mathcal{B}_1$ *and* $\mathcal{B}_4$ *most support Tom's intentions. Thus, one, the other or some combination of both should constitute Tom's selected belief revision.*

example 3.1.1, upon initial examination, may intuitively appear to argue against the revision of belief based on the intentions currently held. Revision $\mathcal{B}_1$ discards the rule that an interest rate rise will increase Tom's mortgage. $\mathcal{B}_4$, on the other hand, discards both rules pertaining to the effect of interest rate increases on Tom's savings and mortgage. The admissibility of such convenient selections arises as a consequence of the treatment of beliefs as bases or sets. When considering beliefs in such circumstances, rules are no more fundamental than facts. Both are subject to the same scepticism as to their objective truth. Thus, the agent may be equally incorrect in its belief in a rule (such as interest rates increasing one's savings) as it is in its belief in a fact (such as an increase in interest rates). Thus, unless one is to adopt a philosophy in which some beliefs are a priori more fundamental than others [35, 36], or one revises with a preference towards informational content [42, 65], the argued revisions are as valid as any other.

As demonstrated in example 3.1.1, maintenance conditions can be captured via constraints on the agent's beliefs. An agent's beliefs aim to reflect the environment in which it is situated. Thus, by querying beliefs, maintenance conditions provide an ongoing balancing connection between the state of the world and the behaviours an agent uses to manipulate it.

(3.1.1) Definition (Maintenance Condition):
Maintenance conditions are captured by the following syntactic schema:

$$\iota : \texttt{while}\,(\{\beta_1, \ldots, \beta_n\})$$

where: ι is a behaviour that an agent is undertaking as specified in the language of the agent's intentions ($\mathcal{L}_{\mathcal{I}}$); and $\{\beta_1, \ldots, \beta_n\}$ is a set of formulæ in the language in the logic of the agent's beliefs ($\mathcal{L}_{\mathcal{B}}$) that must hold of the agent's beliefs while ever the agent holds ι as an intention.

The requirement expressed by such a condition is that, should the listed conditions fail to hold of the agent's beliefs, the associated behaviour is to be modified, dropped or, possibly, an alternative adopted in its stead. The use of a set of conditions, rather than a single condition, which, assuming the language of the logic of belief is sufficiently expressive to conjoin multiple conditions into a single equivalent formula, is to ease the dynamics of the maintenance condition.

Conceptually, behaviours can be complex, of significant duration, hierarchically structured or any combination of these. Thus, maintenance conditions can be applicable to the entire duration of a behaviours execution, or one or more portions thereof. One motivating case for dynamic maintenance conditions is their use to capture causal dependencies between goals in a plan. Causal dependencies arise when a goal in an early part of a plan establishes a necessary condition for the success of a subsequent goal in that plan. In order to ensure the maintenance of such dependencies, the establishment of a maintenance condition on the successful establishment of the condition in question is necessary. Subsequent to the successful completion of the dependent goal, the condition can be dropped, having served its purpose. There are a number of advantages in expressing maintenance conditions within the plan to which they apply. It obviates the need of every plan that may be utilized throughout the duration of the constraint from manually having to respect it. Such plans will not be consistent with the intended constraint and subsequently discarded from adoption as valid intentions. Further, because such plans may be adopted in circumstances in which the constraint is inapplicable, associating the constraint with these plans may invalidate them in situations in which they are adoptable. Finally, it avoids declarations of constraints for all the contexts for which such plans may be used.

In the context of AgentSpeak, each semantic object representing a plan in the agent's plan library $\mathcal{P}$ requires an additional element: the current set of conditions to monitor. The grammar of AgentSpeak plans figure 3.1 require no modification

to support this. Maintenance conditions are added and subtracted from the currently monitored set of maintenance conditions ($\varkappa$) as specified through the set of modifiers (*modifiers*) preceding and following the execution of each goal ($\eth$).

(3.1.2) Definition (Plan with Maintenance Conditions):

$$\tau : \varpi \parallel \varkappa \leftarrow \pi$$

where:

(τ, ϖ, π) *: are as definition 2.1.3.*

($\varkappa$) *: is the current set of maintenance conditions that are required to hold to ensure the validity of the plan. Each maintenance condition (μ) is represented as a query on the agent's belief state.*

Figure 3.1 outlines the modifications required of the syntax of AgentSpeak necessary to facilitate the specification of maintenance conditions. These maintenance conditions are dynamic. They are not fixed at plan construction time. Additional conditions are added and others removed as the execution of the plan progresses. This separates the lifetime of an in-condition from that of the plan in which it is embedded.

Goals are processed in AgentSpeak according to the following steps:

(1.) generate and insert into the set of events an event that corresponds to the goal in question;

(2.) the aforementioned event is eventually selected from the event set, a means for handling it is found and the result returned to the intentions of the agent;

(3.) the resulting intention is eventually selected for execution sufficiently often such that the event has been handled;

(4.) the plan containing the goal is updated;

(5.) and finally, at some later stage, the plan is again selected for execution and the next goal is scheduled for execution.

Between each step in this process an unknown number of reasoning cycles may be undertaken by the agent due to execution of its other, unrelated, intentions. Consequently, to provide sufficient granularity on the modification of maintenance conditions it is necessary to have access both directly prior to (before item 1), and after

(after item 4) the execution of a goal. This is achieved through the extension of the syntax for goals to provide modifiers at both initialization and completion. Each modifier is a set of additions or retractions from the current set of maintenance conditions. An addition is constituted by a literal prefixed with the "+" symbol. Similarly, a retraction is constituted by a literal prefixed with the "-" symbol. While such a scheme is verbose, the specification of maintenance condition modifiers is optional and is likely to be largely automated via causal dependency analysis[108, 98].

$$
\begin{array}{lcll}
\textit{agent} & ::= & \mathcal{B}\ \mathcal{P} & \\
\mathcal{B} & ::= & \beta_1, \ldots, \beta_n & (n \geq 0) \\
\mathcal{P} & ::= & \rho_1, \ldots, \rho_n & (n \geq 1) \\
\rho & ::= & \tau : \omega \leftarrow \pi & \\
\tau & ::= & +atom \mid -atom \mid +\eth \mid -\eth & \\
\omega & ::= & condition \mid \top & \\
condition & ::= & atom \mid \neg atom \mid condition\ \ \&\ \ condition & \\
\pi & ::= & sequent; \top \mid \top & \\
sequent & ::= & \alpha \mid \eth' \mid update \mid sequent; sequent & \\
atom & ::= & \mathsf{P}(\vec{t}) & \\
\alpha & ::= & \mathsf{A}(\vec{t}) & \\
\eth' & ::= & \{modifiers\}\ \eth\ \{modifiers\} & \\
modifiers & ::= & modifier_1, \ldots, modifier_n & (n \geq 1) \\
modifier & ::= & +condition \mid -condition & \\
\eth & ::= & !atom \mid ?atom & \\
update & ::= & +\beta \mid -atom & \\
\beta & ::= & \mathsf{P}(\vec{g}) & \\
term & ::= & \mathsf{F}(\vec{t}) \mid \mathsf{variable} \mid ground & \\
ground & ::= & \mathsf{F}(\vec{g}) \mid \mathsf{string} \mid \mathsf{number} & \\
\vec{t} & ::= & term_1, \ldots, term_n & (n \geq 0) \\
\vec{g} & ::= & ground_1, \ldots, ground_n & (n \geq 0)
\end{array}
$$

Figure 3.1: Formal Grammar for AgentSpeak extended with Maintenance Condition Modifiers

Maintenance conditions have been integrated into rational agent architectures previously[33, 34]. However, such facilities have not been available in AgentSpeak systems. The closest analogue is that of the *maintenance goal* pattern provided by

Jason[10]. There are fundamental differences between the two. Maintenance goals are separately intended and have a life-cycle independent of any given plan. Maintenance conditions are not intentions in their own right and their life-cycles are tied to the plans to which they pertain. On the failure of a maintenance goal, one or more plans may be adopted to re-establish the condition to be maintained. On the failure of a maintenance condition, the containing plan fails and an alternative is adopted in its stead.

3.2 Outcome Conditions

In order to execute purposeful behaviour, an intentional agent needs to reason about the success and failure of its past endeavours. In order to be reasoned with, instances of success and failure must be monitored for and detected within the environment. Observing the environment for indications of success and failure are interlinked. Signs of success imply the absence failure and those of failure the absence of success. However, the absence of failure does not imply success nor does the absence of success imply failure. Existing models of intentional behaviour focus on the verification of success without regard to detecting failure. Consequently, these formalisms assume failure whenever success is unable to be verified. Given the non-equivalence between failure and the absence of success this assumption is invalid and that instances of failure must be detected independently.

Failure is manifest in two ways: failure to complete and failure to produce the desired outcomes. Intentional agents monitoring for failure must critique their own behaviour with these sources of failure in mind. Auditing for failure to complete is less challenging than checking for instances where behaviours do not produce their intended effects. For instance, detecting the failure of an action to complete can be achieved using proprioception and interoception[1]. Similarly, detecting the failure of a plan to complete can be reduced to detecting the failure of one of the sub-goals of the plan. Monitoring for the failure of a particular behaviour to produce its desired outcomes is more complicated. Some of the complexities associated with this form of monitoring are revealed in the following scenario:

(3.2.1) Scenario:
John makes a cup of tea every morning. To do so he uses an old kettle which has no

[1] *Proprioception* is the perception of stimuli arising internally within an agent, in particular, pertaining to the location and relative position of the body. *Interoception* is the perception of the internal stimuli pertaining to pain and the internal organs. *Exteroception* is the mode of perception by which the external environment perceived.

light to indicate when it is on. In order to achieve his goal of making tea he must boil water and he turns the kettle on with this goal in mind. He can tell that he has succeeded to boil the water when the kettle whistles or he sees steam flowing from it. There are a number of things that can prevent this action from achieving his goal: a blackout, the kettle shorts or the kettle is empty. He can detect the occurrence of a blackout and infer from this that the water will not actually boil. However, in the case of either a short or an empty kettle his suspicions are only aroused once an inordinately long time for a kettle to boil has passed and he is yet to notice it doing so.

$$\mathcal{I} = \left\{ \left[\text{`boil water'} : \begin{array}{l} \texttt{suc}(\{\text{`whistle'}, \text{`steaming'}\}), \\ \texttt{fail}(\{\text{`blackout'}, \text{`too long'}\}) \end{array} \right] \right\}$$

Aside from the inequality of failure and the lack of indications of success, there are a number of additional issues exposed by scenario 3.2.1. The success or failure of an action or plan is not, necessarily, immediately apparent. There may be a delay between the time at which the behaviour is completed and the time at which its effects become observable. Additionally, the effect for which a behaviour was executed may itself not be directly observable. However, there may be beliefs which are syntactically unrelated to the intended effect, which, when observed provide evidence for asserting that the behaviour has produced or will not produce the intended effect. Furthermore, it may be necessary to separate the inferences from these indicator conditions to the causation of the effect from the agent's wider beliefs. This is due to the sensitivity of the inference to the context in which it is to be made. Referring again to scenario 3.2.1, an inference from a 'whistle' to 'boiling kettle' may only be valid provided the agent is undertaking a behaviour to which this inference pertains. In other contexts, such as playing a sport, the inference from 'whistle' to 'stop play' may be valid, whereas the inference from 'whistle' to 'boiling kettle' is unlikely to be. We can summarize the requirements an agent must satisfy in order to successfully reason about success and failure.

(3.2.1) Requirement:
A monitoring agent must have a means by which observable elements of the environment which indicate the success or failure of an effect to be realized can be associated with its causing effect.

(3.2.2) Requirement:
A monitoring agent must have means by which it can determine if an action/plan has produced the effect for which it was executed.

(3.2.3) Requirement:
A monitoring agent must have means by which it can determine that an action/plan will not produce the effect for which it was executed.

(3.2.4) Requirement:
A monitoring agent must remain uncertain of the outcome of an action/plan until it has evidence by which it may conclude that the outcome has been or will not be achieved.

(3.2.5) Requirement:
A monitoring agent must have means by which inferences from indications of success or failure can be context dependent. Because they are not valid context independent inferences, they must remain distinct from the agent's beliefs proper.

(3.2.6) Requirement:
A monitoring agent must have strategies by which simultaneous evidence of both success and failure of a given behaviour can be resolved.

The necessity of requirement 3.2.6 is made evident by the following example:

(3.2.1) Example:
Tom is informed of another interest rate rise. His potential revisions are the same as those outlined in example 3.1.1. Because of this, Tom decides it is time to save money. He will be successful if he comes to believe his savings have increased and fail if his mortgage increases:

$$\mathcal{I} = \{[\text{`}\textit{Save money}\text{'} : \texttt{suc}(\text{`}\textit{Savings increase}\text{'}), \texttt{fail}(\text{`}\textit{Mortgage increase}\text{'})]\}$$

If Tom is optimistic, then he will assume the success of his intention in light of uncertainty. Given that $\mathcal{B}_1$ allows Tom to conclude the success of his intention without also supporting its failure, Tom should accept $\mathcal{B}_1$. However, if Tom prefers to identify his potential mistakes early, he may prefer revisions $\mathcal{B}_2$ or $\mathcal{B}_3$. If Tom is cautious, he will attempt to minimize the effects on his intentions and prefer $\mathcal{B}_4$.

(3.2.2) Example:
Tom and his wife have decided to have a baby. Since it will be their first baby they may be eligible for a government subsidy called the 'baby bonus' depending on their current economic situation. Tom's beliefs are as in example 3.2.1. As part of intending to have a baby, Tom also intends to gain the 'baby bonus'. Since the 'baby bonus' is

worth much more than his investments will earn, he prefers[2] *to get the 'baby bonus' over a savings increase. However, should his investments increase in value then that will elevate him over the threshold making him ineligible to receive it. His intentions and preferences can be formalised as follows:*

$$\mathcal{I} = \left\{ \begin{array}{c} \left[\text{'Save money'} : \begin{array}{l} \texttt{suc}(\text{'Savings increase'}), \\ \texttt{fail}(\text{'Mortgage increase'}) \end{array} \right], \\ [\text{'Baby bonus'} : \texttt{suc}(\text{'Have baby'}), \texttt{fail}(\text{'Savings increase'})] \end{array} \right\}$$

$$\text{'Save money'} \lhd \text{'Baby bonus'}$$

Tom is told of another interest rate rise. The effect on his beliefs is as in example 3.2.1. We can see that both $\mathcal{B}_1$ and $\mathcal{B}_3$ are not ideal revisions as they will require Tom to drop his intention of gaining the 'baby bonus'. $\mathcal{B}_2$ and $\mathcal{B}_3$, if adopted, would result in Tom's having to drop his intention to 'save money'. Thus, the best revision considering Tom's intentions is $\mathcal{B}_4$. Had Tom's preferences towards his intentions been reversed ('Baby bonus' < 'Save money') then the picture is more complicated. If Tom selects $\mathcal{B}_1$ then although his intention to save money has been achieved he must relinquish his intention to gain the 'baby bonus'. Alternatively, he could adopt $\mathcal{B}_4$ and retain both intentions but also risk his most preferable intention ('Save money') failing later.

Examples 3.2.1 and 3.2.2 demonstrate an approach to capture success, failure and uncertainty in the outcome of intentional behaviour. In particular, they showed the need for both success and failure conditions for effects and that, like maintenance conditions, success and failure conditions can be captured via queries on the agent's beliefs. These examples also demonstrated how preferences towards intentions can and should influence the outcome of belief revision. Subsequently, we generalize the approaches of examples 3.2.1 and 3.2.2 to produce definition 3.2.1.

[2] Tom's intention preference relation is as specified by $\lhd$.

(3.2.1) Definition (Effective Intention):
Success and failure conditions are captured by the following syntactic schema:

$$\iota : \texttt{suc}(\{\beta\}), \texttt{fail}(\{\beta'\})$$

where:

(ι) *: is a behaviour that an agent is undertaking as represented as a formula in the language of the logic of the agent's intentions ($\mathcal{L}_{\mathcal{I}}$);*

$(\texttt{suc}(\{\beta\}))$ *is a set of formulæ in the language of the agent's beliefs ($\mathcal{L}_{\mathcal{B}}$) that once a consequence of the agent's belief indicate the intention has achieved its purpose; and, correspondingly,*

$(\texttt{fail}(\{\beta\}))$ *is a set of formulæ in the language of the agent's beliefs ($\mathcal{L}_{\mathcal{B}}$) that once a consequence of the agent's belief indicate the environment will not evolve so as to satisfy a success condition and, thus, has failed.*

One of the primary differences between the approach advocated above and those outlined in section 2.2 is that the derivation of consequences, both logical and causal, is not of primary concern. Thus the frame problem, ramification problem and other related issues are not applicable. By avoiding these problems, the solutions that require the extension of the deduction mechanism to facilitate second order reasoning or the limitations imposed to avoid such an extension are unnecessary. Although the qualification problem is not applicable to the reasoning of a monitoring agent, it does apply to the designer in the specification of the outcomes to monitor for. Thus the monitoring agent assumes the completeness of the specification with which it is working. Because it is assumed that the specification is static and the agent uses no techniques to update this specification, it is necessary to appeal to a form of bounded rationality as justification for an agent's behaviour and reasoning. The best the agent can achieve is optimality with respect to this specification, regardless of its accuracy given an evolving environment. Compounding this, an agent may not reach even this level of optimality given the accuracy and completeness concessions required to remain responsive.

The semantic component that reflects the extensions to the syntax to facilitate the expression of outcome conditions is defined as follows:

(3.2.2) Definition (Effective Plan):

$$\tau : \omega \leftarrow \pi.\epsilon$$

where:

(τ, ω, π) *: are as definition 2.1.3*

(ϵ) *: are a set of effects where each effect is defined according to definition 3.2.3*

Rather than specifying effects (ϵ) as a set of literals that must become a consequence of the agent's beliefs, each effect (ξ) is a compound entity:

(3.2.3) Definition (Effect):

$$\xi = \lambda : \langle \{\mathtt{suc}_0, \ldots, \mathtt{suc}_n\}, \{\mathtt{fail}_0, \ldots, \mathtt{fail}_m\} \rangle$$

where:

(λ) *: the name by which to refer to the effect as a whole;*

$(\mathtt{suc}_0, \ldots, \mathtt{suc}_n)$ *: the conditions on the agent's beliefs that indicate that the effect (ξ) was realized; and,*

$(\mathtt{fail}_0, \ldots, \mathtt{fail}_m)$ *: the conditions on the agent's beliefs that indicate that the effect (ξ) will not be realized.*

The changes necessary to the AgentSpeak grammar are highlighted in figure 3.2. Each plan has a set of effects specified. Each effect specifies the conditions under which a plan execution is complete and successful. Each effect has success and failure conditions. The success conditions define the sufficient properties of the world to allow the inference that the effect has been realized. Each effect can be realized at different times and once the agent has determined that an effect has been realized, it no longer monitors the conditions of that effect. Once the agent has detected that all the effects have been successfully realized the plan is complete. If any failure effect of an event follows from the agent's beliefs, the agent can infer that the event cannot or will not be realized. Once a single effect has been classified as failed, the plan too has failed.

The open-source AgentSpeak implementation Jason[10] offers an alternative approach to the representation and reasoning with declarative achievement goals.

$$
\begin{array}{lcll}
\mathit{agent} & ::= & \mathcal{B}\ \mathcal{P} & \\
\mathcal{B} & ::= & \beta_1, \ldots, \beta_n & (n \geq 0) \\
\mathcal{P} & ::= & \rho_1, \ldots, \rho_n & (n \geq 1) \\
\rho & ::= & \tau : \omega \leftarrow \pi.\epsilon & \\
\tau & ::= & +\mathit{atom} \mid -\mathit{atom} \mid +\eth \mid -\eth & \\
\omega & ::= & \mathit{condition} \mid \top & \\
\mathit{condition} & ::= & \mathit{atom} \mid \neg\mathit{atom} \mid \mathit{condition}\ \ \&\ \ \mathit{condition} & \\
\pi & ::= & \mathit{sequent}; \top \mid \top & \\
\mathit{sequent} & ::= & \alpha \mid \eth' \mid \mathit{update} \mid \mathit{sequent}; \mathit{sequent} & \\
\mathit{atom} & ::= & \mathsf{P}(\vec{t}) & \\
\alpha & ::= & \mathsf{A}(\vec{t}) & \\
\epsilon & ::= & \xi_1, \ldots, \xi_n & (n \geq 0) \\
\xi & ::= & \mathit{atom} : \mathtt{suc}(\mathit{conditions}), \mathtt{fail}(\mathit{conditions}) & \\
\mathit{conditions} & ::= & \mathit{condition}_1, \ldots, \mathit{condition}_n & (n \geq 1) \\
\eth' & ::= & \{\mathit{modifiers}\}\ \eth\ \{\mathit{modifiers}\} & \\
\mathit{modifiers} & ::= & \mathit{modifier}_1, \ldots, \mathit{modifier}_n & (n \geq 1) \\
\mathit{modifier} & ::= & +\mathit{condition} \mid -\mathit{condition} & \\
\eth & ::= & !\mathit{atom} \mid ?\mathit{atom} & \\
\mathit{update} & ::= & +\beta \mid -\mathit{atom} & \\
\beta & ::= & \mathsf{P}(\vec{g}) & \\
\mathit{term} & ::= & \mathsf{F}(\vec{t}) \mid \mathsf{variable} \mid \mathit{ground} & \\
\mathit{ground} & ::= & \mathsf{F}(\vec{g}) \mid \mathsf{string} \mid \mathsf{number} & \\
\vec{t} & ::= & \mathit{term}_1, \ldots, \mathit{term}_n & (n \geq 0) \\
\vec{g} & ::= & \mathit{ground}_1, \ldots, \mathit{ground}_n & (n \geq 0)
\end{array}
$$

Figure 3.2: Formal Grammar for AgentSpeak extended with Outcome Conditions

This leverages some of the additional *internal actions/goal types* of the Jason interpreter. A number of *plan patterns* capture the monitoring and commitment strategies of individual declarative achievement goals. This approach is advantageous in a number of ways. It leverages existing facilities of the interpreter, requiring only a pre-processor to transform the goal macros into valid AgentSpeak syntax. Monitoring is achieved through the existing event generation and plan adoption mechanisms, leading to low cost monitoring. Finally, plan patterns facilitate the specification of commitment strategies on a goal by goal basis rather than globally for all goals. However, it is limited compared to the approach advocated here. For

instance, the effect for which the plan was adopted must become a consequence of the agent's beliefs prior to the completion of its execution. No facility is made for the delays in the realization of outcomes. Thus, remedial plans and subsequent wasted effort and reasoning may ensue. Additionally, no means for the specification of multiple effects for a given plan is provided. Further, it is assumed that the content of the achievement event is always directly perceivable and, as such, is always the only success condition to monitor. Finally, no mechanism for the specification of failure conditions is available. While it is trivial to extend the definition of the plan patterns to accommodate alternative sentences to monitor beyond those within the achievement event, the means by which the approach could be extended to overcome the additional limitations is less clear.

The representation of outcome conditions bears similarities with that of the STRIPS planning approach. However, there are significant differences. While the STRIPS approach defines operators with add and delete lists of literals, the approach advocated here uses a list of literals to both represent the success and failure conditions of a given effect. The difference, however, is the reasoning task to which they are applied. The STRIPS methodology utilizes the add and delete lists to directly modify the "beliefs" of the executing agent. Each operator is assumed to be instantaneous and the add and delete lists complete and accurate. The effect conditions, alternatively, are used as the focus of monitoring. They are not applied to the beliefs of the agent to modify them in any way. They are checked intermittently, according to the adopted monitoring strategy, and there is no assumption of completeness, only of appropriateness.

Similar representations to those posed above have been applied to goals. One such representation is that of [108] in which goals are defined to be of the form $\text{Goal}(s, P, f)$. In this representation s is a logical formula over the agent's beliefs that defines the success states of the goal, P is a set of guarded plans that are designed to achieve s and f is a logical formula over the agent's beliefs that defines the failure conditions of the goal. Such a representation allows for the consistency of the goals adopted by an agent to be maintained. A number of levels of consistency were defined given such a representation: necessary consistency, necessary inconsistency, possible consistency and, finally, possible inconsistency. Additionally, the identification of possible and necessary positive interactions between goals was facilitated. An operational semantics defining the life cycle of goals with respect to an agent's beliefs was provided. It was shown that such a representation given the provided operational semantics satisfied a number of rationality properties of goals. In particular it was shown that goals are persistent; are only retained while unachieved; and, are only adopted and maintained when possible, consistent

and known.

Another such representation is that of [99], this time applied to both plans and goals. Goals are represented as tuples containing a label, a set of direct and desired effects described by logical formulae and a set of plan types that achieve the goal. Plans are structures containing a label, a logical formula that defines the preconditions of the plan, a logical formula representing the maintenance condition of the plan and a final logical formula representing the direct and desired effects of the plan. From this representation of plans and goals it was shown how, through the use of goal-plan trees, both the interactions and dependencies between goals and plans can be detected. It was then demonstrated how these interactions and dependencies could be used to ensure the consistent adoption of both plans and goals.

3.3 Summary

This chapter has presented syntactic means for the representation of both maintenance and outcome conditions. Three primary extensions were advocated to this end: maintenance conditions on plans; effects on plans to be monitored; and, progressive maintenance condition modifiers in the bodies of plans. These means are applicable both generally and particularly to the AgentSpeak architecture to be developed in the chapters to follow. These syntactic forms were motivated by numerous examples and the philosophy of section 2.1.1. The relationship of these representations to those of section 2.2 was discussed and the applicability of the problems those approaches entailed determined. A number of challenges were posed by the requirement of compatibility with AgentSpeak. These significantly shaped the resulting approach proposed. The development of these representations facilitate the reasoning mechanisms crucial to the investigations of the three primary hypotheses of this thesis.

History is on our side (as long as we can control the historians).

Unknown.

4 Monitoring

Hypothesis:
The successful monitoring of the effects and validity of intentional behaviour requires mechanisms to: (1.) resolve the conflict for the success or failure of a behaviour given evidence for both; and (2.) balance the time spent monitoring against other mental tasks.

In order to test the hypothesis above, it will be necessary to advocate a representation that defines appropriate aspects of intentional behaviour to monitor. It will then be necessary to integrate the aforementioned representation into the syntax and semantics of an intentional agent architecture. Further, given the known issue of balancing monitoring and other mental processes, the identification of strategies for the balance of monitoring and other behaviour are required. Having integrated the monitoring process into an agent architecture, it must then be generalized to utilize such strategies as well. The representations will be those described, motivated and analysed in chapter 3, and integrated as per definitions 3.1.2 and 3.2.2 and figures 3.1 and 3.2. A consequence of the chosen representation, in particular the structure by which effects are described, is that conflicts will arise during the monitoring process. Such conflicts arise in scenarios in which conditions indicative of both success and failure are satisfied. These conflicts will require strategies to resolve. A number of such strategies will be proposed. The chosen architecture and semantics will then require generalization for their integration. The semantics into which these representations are to be integrated are those of AgentSpeak as discussed in section 2.1.3. This integration will be presented as modifications with respect to the abstract semantics of definition 2.1.2 and rules 2.1.1 to 2.1.21.

4.1 The Semantics of Monitoring

The intuition motivating the following changes to the AgentSpeak interpreter is that a necessary precondition for the monitoring of outcome conditions is the need for them to remain active for monitoring once they have completed execution[21, 23, 22].

4.1.1 Definitions

An AgentSpeak transition system configuration is redefined as a tuple $\langle agent, C, T, S\rangle$ where:

$(agent)$: is a tuple $\langle \mathcal{P}, \gamma_{\mathcal{E}}, \gamma_{\mathcal{O}}, \gamma_{\mathcal{I}}, \Phi_{\epsilon}, \Phi_{\mu}, \Phi_{\chi}\rangle$ representing the constant elements of an agent program where:

- $(\gamma_{\mathcal{E}}, \gamma_{\mathcal{O}}, \gamma_{\mathcal{I}})$: retain their existing definition (see definition 2.1.4).
- (Φ_{ϵ}) : is a function representing the current effect monitoring strategy.
- (Φ_{μ}) : is a function representing the current maintenance condition checking strategy.
- (Φ_{χ}) : is a function representing the conflict resolution strategy.

(C) : is as per definition 2.1.4.

(T) : is a tuple $\langle \mathcal{R}, \mathcal{O}, \sigma_{\iota}, \sigma_{\varepsilon}, \sigma_{\omega}, \epsilon, \varkappa\rangle$ where:

- $(\mathcal{R}, \mathcal{O}, \sigma_{\iota}, \sigma_{\varepsilon}, \sigma_{\omega})$: again, retain their original meaning (definition 2.1.4).
- (ϵ) : represents the set of effects that we should monitor this reasoning cycle as defined by F_{ϵ} (see definition 4.1.2).
- $(\varkappa)$: represents the set of maintenance conditions we should check this reasoning cycle as defined by $F_{\varkappa}$ (see definition 4.1.3).

(S) : is the set of annotations applied to each reasoning rule so as to specify the transition relation.

$$S = \left\{ \begin{array}{c} \mathsf{Init}, \mathsf{SelEv}, \mathsf{RelPl}, \mathsf{ApplPl}, \mathsf{SelAppl}, \mathsf{AddIM}, \\ \mathsf{SelInt}, \mathsf{UpdPreMnt}, \mathsf{GenMnt}, \mathsf{MonMnt}, \mathsf{ExecInt}, \mathsf{UpdPostMnt}, \\ \mathsf{GenMnt}_2, \mathsf{MonMnt}_2, \mathsf{GenEff}, \mathsf{MonEff} \end{array} \right\}$$

4.1.2 Auxiliary Functions

Auxiliary functions are built directly into the semantics. An agent designer cannot over-ride these and need not even be aware of their existence. However, they are necessary in order to express computation not easily captured within the transition system based operational semantics methodology.

The intention filtering function ($f_{\mathcal{I}}$) returns the sub-set of the current set of intentions ($C_{\mathcal{I}}$) such that each included intention has elements remaining in the body. This function is used to prevent intentions that have completed executing, but have effects outstanding, from being candidates for processing this reasoning cycle.

4.1.1 Definition (Intention Filter Function):

$$f_{\mathcal{I}}(\mathcal{I}) = \{\iota \in \mathcal{I} \mid \iota = \iota'[\tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon]\iota'' \wedge \pi \neq \varnothing\}$$

The effect generation function (f_{ϵ}) generates the set of effects that should be monitored this reasoning cycle. This is achieved by iterating over each intention in the set of intentions, each plan on each intention stack and each effect within each plan. This combination, in addition to the current state of the agent, is passed onto the user defined effect monitoring strategy function. This function defines whether this effect should be checked in this cognitive cycle. The return value of this function is the set of intention (ι), plan (ρ), effect (ξ) tuples $\langle \iota, \rho, \xi \rangle$ for which the strategy returns true.

4.1.2 Definition (Effect Generation Function):

$$f_{\epsilon}(T, C, agent) = \left\{ \langle \iota, \rho, \xi \rangle \;\middle|\; \begin{array}{c} \iota \in C_{\mathcal{I}} \wedge \iota = \iota'[\rho]\iota'' \wedge \rho = \tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon \wedge \\ \xi \in \epsilon \wedge agent_{\Phi_{\epsilon}}(C, T, \iota, \rho, \xi) = \top \end{array} \right\}$$

Similar to the effect generation function (f_{ϵ}), the maintenance condition generation function ($f_{\varkappa}$) iterates over each intention, each plan within each intention stack and each maintenance condition within each plan. This function returns the set of intention (ι), plan (ρ), maintenance condition (μ) tuples $\langle \iota, \rho, \mu \rangle$ such that the maintenance condition monitoring strategy returns true.

4.1.3 Definition (Maintenance Condition Generation Function):

$$f_{\varkappa}(C, T, agent) = \left\{ \langle \iota, \rho, \mu \rangle \;\middle|\; \begin{array}{c} \iota \in C_{\mathcal{I}} \wedge \iota = \iota'[\rho]\iota'' \wedge \rho = \tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon \wedge \\ \mu \in \varkappa \wedge agent_{\Phi_{\mu}}(C, T, \iota, \rho, \mu) = \top \end{array} \right\}$$

Because the set of effects to monitor is regenerated each reasoning cycle, it is necessary to remove those effects that have sufficient evidence for the agent to conclude their success from their encompassing plan. Otherwise, the agent will continue to monitor for these effects and the plan will never be considered complete. This is achieved using the update effects function. This function maps an intention, plan and effect into a new intention. Because an agent can monitor for and consequently conclude the success of an effect in any level of an intention stack, this function retrieves the input plan ($\tau : \bar{\omega} \parallel \varkappa \leftarrow \pi.\epsilon$) from the intention ($\iota$), removes the input effect (ξ) from the input plan ($\tau : \bar{\omega} \parallel \varkappa \leftarrow \pi.\epsilon$) and reconstructs the intention stack (ι') to be returned.

(4.1.4) Definition (Updating Effects):

$$\text{update}(\iota, \tau : \bar{\omega} \parallel \varkappa \leftarrow \pi.\epsilon, \xi) = \iota'[\tau : \bar{\omega} \parallel \varkappa \leftarrow \pi.\epsilon']\iota'' \quad \text{where: } \epsilon' = \epsilon \setminus \{\xi\}.$$

4.1.3 Rules

The first rule altered from the original (as presented in section 2.1.3) is that of **SelInt**. This is in order to integrate the intention filtering function (as described above) into the intention selection process. This also introduces the maintenance condition checking functionality (defined below) into the transition system. The first rule is applicable in the case where there are intentions for which further processing is required. The second handles the case where there are not.

$$\frac{f_{\mathcal{I}}(C_{\mathcal{I}}) \neq \varnothing}{\langle agent, C, T, \mathsf{SelInt}\rangle \longrightarrow \langle agent, C, T', \mathsf{UpdPreMnt}\rangle} \quad \text{where: } T'_{\sigma_\iota} = \gamma_{\mathcal{I}}(f_{\mathcal{I}}(\mathcal{I}))$$

Rule 4.1.1 **SelInt$_1$**

$$\frac{f_{\mathcal{I}}(C_{\mathcal{I}}) = \varnothing}{\langle agent, C, T, \mathsf{SelInt}\rangle \longrightarrow \langle agent, C, T, \mathsf{GenMnt}\rangle}$$

Rule 4.1.2 **SelInt$_2$**

Rule 4.1.3 updates the maintenance conditions of the selected intention. This involves the addition of all the conditions prefixed with "+" to, and the removal of those prefixed with "-" from, the current set of maintenance conditions. This must be done in such a way to minimize the size of the resulting condition size. To achieve this, the additions to the maintenance conditions are made prior to the removals. This is necessary to handle the unlikely case in which the same condition is both added and removed in the same modifier. Furthermore, the set of processed maintenance condition modifiers is removed from the body of the plan.

$$\frac{T_{\sigma_\iota} = \iota[\tau : \varpi \parallel \varkappa \leftarrow \{in\}\ \eth\ \{in'\}\,;\pi.\epsilon]}{\langle agent, C, T, \mathsf{UpdPreMnt}\rangle \longrightarrow \langle agent, C', T', \mathsf{GenMnt}\rangle}$$

Rule 4.1.3
UpdPreMnt

where: $T'_{\sigma_\iota} = \iota[\tau : \varpi \parallel \varkappa' \leftarrow \pi'.\epsilon]$

$$C'_{\mathcal{I}} = (C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\}) \cup \{T_{\sigma_\iota}\}'$$
$$\varkappa' = (\varkappa \cup \varkappa'') \setminus \varkappa'''$$
$$\varkappa'' = \{\mu \mid +\mu \in \{in\}\}$$
$$\varkappa''' = \{\mu \mid -\mu \in \{in\}\}$$
$$\pi' = \eth\ \{in'\}\,;\pi$$

Following the update of the maintenance conditions of the selected intention, it is necessary to then validate the resulting conditions. There is a requirement to wait until after the application of the intention selection function to ensure the most up-to-date set of conditions is available to monitor and to allow the generator function to make decisions based on the resulting selection. The first rule pertaining to this validation generates the conditions to monitor during the current reasoning cycle. These conditions are collected in a temporary element of the agent's circumstance ($T_\varkappa$). The validation process will then remove and process each condition individually, progressing to the next stage in the reasoning cycle once the temporary collection of conditions is empty.

$$\frac{C_{\mathcal{I}} \neq \varnothing}{\langle agent, C, T, \mathsf{GenMnt}\rangle \longrightarrow \langle agent, C, T', \mathsf{MonMnt}\rangle}$$

Rule 4.1.4 **GenMnt**$_1$

where: $T'_{\varkappa} = f_{\varkappa}(T, C, agent)$

The second rule handles the case where there are no maintenance conditions to be validated this reasoning cycle.

$$\frac{C_{\mathcal{I}} = \varnothing}{\langle agent, C, T, \mathsf{GenMnt}\rangle \longrightarrow \langle agent, C, T, \mathsf{ExecInt}\rangle}$$

Rule 4.1.5 **GenMnt**$_2$

Given that there are maintenance conditions to check[1], the rule (MonMnt) selects an maintenance condition and checks it against the agent's current beliefs. If the maintenance condition is a consequence of the agent's beliefs then the maintenance condition is removed from the set of maintenance conditions and the rule processes the next maintenance condition.

$$\frac{\langle \iota, \rho, \mu\rangle \in T_{\varkappa} \wedge \mu \in Cn(C_{\mathcal{B}})}{\langle agent, C, T, \mathsf{MonMnt}\rangle \longrightarrow \langle agent, C, T', \mathsf{MonMnt}\rangle}$$

Rule 4.1.6 **MonMnt**$_1$

where: $T'_{\varkappa} = T_{\varkappa} \setminus \{\langle \iota, \rho, \mu\rangle\}$

However, if the maintenance condition is not a consequence of the agent's beliefs then the plan has failed. In such circumstances all maintenance conditions within the same plan and intention are removed from the set of maintenance conditions remaining. This is because the plan has already failed and there is no point checking any of its other maintenance conditions. Additionally, an event is added

[1] These conditions are generated via $f_{\varkappa}(T, C, agent)$ as described in rule 4.1.4.

to the set of events indicating the failure of the plan in question. This event is generated so as to utilize the current failure handling facilities of AgentSpeak. Thus, such events are only generated for achievement goals. It is also necessary to remove all plans from above the point of failure within the failed intention. This is necessary so that processing can continue at the point of failure once the failure itself has been handled. Finally, the intention is removed from the set of intentions.

$$\frac{\langle \iota[\tau : @ \parallel \varkappa \leftarrow \pi.\epsilon]\iota', \tau : @ \parallel \varkappa \leftarrow \pi.\epsilon, \mu\rangle \in T_\varkappa \wedge \mu \notin Cn(C_\mathcal{B})}{\langle agent, C, T, \mathsf{MonMnt}\rangle \longrightarrow \langle agent, C', T', \mathsf{MonMnt}\rangle}$$

Rule 4.1.7
MonMnt$_2$

where: $C'_\mathcal{E} = C_\mathcal{E} \cup \{\langle -!atom, \iota\rangle\}$ if $\tau = +!atom$
$T'_\varkappa = T_\varkappa \setminus \{\langle \iota[\tau : @ \parallel \varkappa \leftarrow \pi.\epsilon]\iota', \tau : @ \parallel \varkappa \leftarrow \pi.\epsilon, \mu'\rangle\}$
$C'_\mathcal{I} = C_\mathcal{I} \setminus \{\iota[\tau : @ \parallel \varkappa \leftarrow \pi.\epsilon]\iota'\}$

Having monitored each maintenance condition and dealt with each appropriately, it is necessary to execute the selected intention.

$$\frac{T_\varkappa = \varnothing}{\langle agent, C, T, \mathsf{MonMnt}\rangle \longrightarrow \langle agent, C, T, \mathsf{ExecInt}\rangle}$$

Rule 4.1.8
MonMnt$_3$

The rules for executing the topmost plan of the selected intention remain fundamentally the same as those outlined previously (section 2.1.3). However, once the intention has finished execution, it is necessary to update its maintenance conditions. Like, rule 4.1.3, it is necessary to minimize the size of the maintenance condition that results. Complicating issues further is the fact that once the plan is complete, all the associated maintenance conditions must be removed. Once the plan has completed execution, its associated maintenance conditions are no longer meaningful and must be retracted.

$$\frac{T_{\sigma_\iota} = \iota[\tau : \varpi \parallel \varkappa \leftarrow \eth\{in\}; \pi.\epsilon]}{\langle agent, C, T, \mathsf{UpdPostMnt}\rangle \longrightarrow \langle agent, C', T', \mathsf{GenMnt}_2\rangle}$$

Rule 4.1.9
UpdPostMnt$_1$

where: $T'_{\sigma_\iota} = \iota[\tau : \varpi \parallel \varkappa' \leftarrow \pi.\epsilon]$
$C'_{\mathcal{I}} = C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\} \cup \{T'_{\sigma_\iota}\}$
$\varkappa' = (\varkappa \cup \varkappa'') \setminus \varkappa'''$
$\varkappa'' = \{\mu \mid +\mu \in in\}$
$\varkappa''' = \{\mu \mid -\mu \in in\}$

$$\frac{T_{\sigma_\iota} = \iota[\tau : \varpi \parallel \varkappa \leftarrow \top.\epsilon]}{\langle agent, C, T, \mathsf{UpdPostMnt}\rangle \longrightarrow \langle agent, C', T', \mathsf{GenMnt}_2\rangle}$$

Rule 4.1.10
UpdPostMnt$_2$

where: $T'_{\sigma_\iota} = \iota[\tau : \varpi \parallel \varkappa' \leftarrow \top.\epsilon]$
$C'_{\mathcal{I}} = C_{\mathcal{I}} \setminus \{T_{\sigma_\iota}\} \cup \{T'_{\sigma_\iota}\}$
$\varkappa' = \varnothing$

Because the rules **GenMnt**$_3$ – **GenMnt**$_4$ and **MonMnt**$_4$ – **MonMnt**$_6$ are identical to **GenMnt**$_1$ – **GenMnt**$_2$ and **MonMnt**$_1$ – **MonMnt**$_3$ except that the next states to which the configuration is transitioned are SelEv_2 and GenEff respectively, there is no reason to repeat them here. It is necessary to ensure the validity of the agent's plans both prior to and after plan execution because an agent can exercise conscious control over its beliefs. The agent may adopt or drop beliefs between the first and second validation of a given plan in a particular reasoning cycle. Compounding this need is the fact that the maintenance conditions of the topmost plan on the selected intention are themselves updated as part of its execution. This change may lead to maintenance conditions failing that, were it not for the second validation process, would go undetected for an indeterminate amount of time dependent on the utilized monitoring strategy.

The next process is that of generating the set of effects to check. Two rules facilitate this. The first handles the case where the set of intentions is empty. Under

such circumstances, the agent has no effects to monitor and progresses to select an event to handle.

$$\frac{C_{\mathcal{I}} = \varnothing}{\langle agent, C, T, \mathsf{GenEff}\rangle \longrightarrow \langle agent, C, T, \mathsf{SelEv}\rangle}$$

Rule 4.1.11
GenEff$_1$

However, should there be elements in the set of intentions the second rule passes the current set of intentions as a parameter to the effect generation function and stores the result in the set of effects to be checked. The generated set of effects to monitor is collected in a temporary element in the agent's circumstance (T_ϵ). Like the processing of the maintenance conditions, each effect to monitor will be removed from the temporary collection and its conditions checked. The process repeats until the temporary collection is empty and all effects monitored.

$$\frac{C_{\mathcal{I}} \neq \varnothing}{\langle agent, C, T, \mathsf{GenEff}\rangle \longrightarrow \langle agent, C, T', \mathsf{MonEff}\rangle}$$

where: $T'_\epsilon = F_\epsilon(T, C, agent)$

Rule 4.1.12
GenEff$_2$

Five rules are required to process the current effects. The first four are repeatedly applied until there are no elements left in the effect set. The first is applicable when there are outstanding effects to be checked and the selected effect is deemed successful by the conflict resolution strategy. It removes the effect from the set of effects awaiting monitoring and from the plan from which it originated. It also updates the intention using definition 4.1.4.

$$\frac{\langle \iota, \rho, \xi \rangle \in T_\epsilon \wedge \Phi_\chi(C_\mathcal{B}, \xi) = \top \wedge \quad \iota'[\tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon]\iota'' = \text{update}(\iota, \rho, \xi) \wedge \epsilon \neq \varnothing}{\langle \mathit{agent}, C, T, \mathsf{MonEff} \rangle \longrightarrow \langle \mathit{agent}, C', T', \mathsf{MonEff} \rangle}$$

Rule 4.1.13 **MonEff**$_1$

where: $T'_\epsilon = T_\epsilon \setminus \{\langle \iota, \rho, \xi \rangle\}$
$C'_\mathcal{I} = (C_\mathcal{I} \setminus \{\iota\}) \cup \{\iota'[\tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon]\iota''\}$

The second rule is necessary to handle the case in which the update of the effects of the monitored plan leads to the realization of all the effects for that plan. This requires the removal of all elements of the intention that fall above the plan in question, in addition to the plan itself. Additionally, it is necessary to remove the goal for which the now completed plan was adopted to achieve. Any bindings generated in satisfying the goal are applied to the remaining intention and the intention returned to the set of intentions for further processing.

$$\frac{\langle \iota, \rho, \xi \rangle \in T_\epsilon \wedge \Phi_\chi(C_\mathcal{B}, \xi) = \top \wedge \quad \iota' = \text{update}(\iota, \rho, \xi)}{\langle \mathit{agent}, C, T, \mathsf{MonEff} \rangle \longrightarrow \langle \mathit{agent}, C', T', \mathsf{MonEff} \rangle}$$

Rule 4.1.14 **MonEff**$_2$

where: $T'_\epsilon = T_\epsilon \setminus \{\langle \iota, \rho, \xi \rangle\}$
$\iota' = \iota''[\tau' : \varpi' \parallel \varkappa' \leftarrow \eth; \pi'.\epsilon'][\tau : \varpi \parallel \varkappa' \leftarrow \pi.\varnothing]\iota'''$
$C'_\mathcal{I} = (C_\mathcal{I} \setminus \{\iota\}) \cup \{\iota''[\tau' : \varpi' \parallel \varkappa' \leftarrow \pi'.\epsilon']\iota'''\theta\}$
$\theta = \mathit{agent}_{\gamma_\theta}(\{\theta \mid \tau\theta = \eth\theta\})$

The third rule is applicable when there are outstanding effects to be checked, but the conflict resolution strategy has judged the selected effect to have failed. This rule creates the appropriate failure event and adds it to the event set. Then the intention is removed from the set of intentions to prevent its further processing until the failure is handled.

$$\frac{\langle \iota, \tau : \bar{\omega} \parallel \varkappa \leftarrow \pi.\epsilon, \xi \rangle \in T_\epsilon \wedge \Phi_\chi(C_\mathcal{B}, \xi) = \bot}{\langle agent, C, T, \mathsf{MonEff} \rangle \longrightarrow \langle agent, C', T', \mathsf{MonEff} \rangle}$$

Rule 4.1.15 **MonEff**$_3$

where: $C'_\mathcal{E} = C_\mathcal{E} \cup \{\langle -!atom, \iota \rangle\}$ if $\tau = +!atom$
$C'_\mathcal{I} = C_\mathcal{I} \setminus \{\iota\}$
$T'_\epsilon = T_\epsilon \setminus \{\langle \iota, \rho, \xi \rangle\}$

The fourth rule handles the case where there are effects to audit, one of these is selected, yet the conflict resolution strategy cannot conclude that the effect has been satisfied or invalidated. Under these circumstances the conflict resolution strategy will return ($\dashv$) to indicate that the effect is neither successful, nor failed. All that must be done in this case is to remove the effect from the effect set so that it will not be tested again.

$$\frac{\langle \iota, \rho, \xi \rangle \in T_\epsilon \wedge \dashv = \Phi_\chi(C_\mathcal{B}, \epsilon)}{\langle agent, C, T, \mathsf{MonEff} \rangle \longrightarrow \langle agent, C, T', \mathsf{MonEff} \rangle}$$

Rule 4.1.16 **MonEff**$_3$

where: $T'_\epsilon = T_\epsilon \setminus \{\langle \iota, \rho, \xi \rangle\}$

Finally, the last rule handles the case where there are no elements in the set of effects to review.

$$\frac{T_\epsilon = \varnothing}{\langle agent, C, T, \mathsf{MonEff} \rangle \longrightarrow \langle agent, C, T, \mathsf{SelEv} \rangle}$$

Rule 4.1.17 **MonEff**$_4$

Given the fact that plans are only removed from the set of intentions upon the realization of all their effects or their failure, the rules in the original semantics pertaining to the clearing of intentions (rules 2.1.19 to 2.1.21) are no longer necessary.

The removal of intended means with completed effects is achieved via the application of rules 4.1.13 to 4.1.17. Plans remain intended beyond the completion of the execution of all their goals pending the realization or failure of their effects. A state of the world may never arise such that an agent can declare a plan successful or failed. In such circumstances the intended plan will be retained and monitored for the lifetime of the agent. All plans in which the intended plan is embedded will also be delayed until the outcome of the intended plan is known. Thus, it is necessary that care be taken in the definition of the success and failure conditions of the effects of plans to ensure the timely and accurate detection of their outcome. For example, assume an agent has an intention of the form:

$$\iota[\tau : \varpi \leftarrow !a; !b; \pi.\epsilon][+!a : \top \leftarrow \pi'.a : \langle \{p\}, \{q\} \rangle]$$

Once the agent has completed the execution of its topmost plan ($\pi' = \top$), it will retain:

$$[+!a : \top \leftarrow \top.a : \langle \{p\}, \{q\} \rangle]$$

as the topmost plan of intention (ι) until either p or q is a consequence of its beliefs. Thus, $!b$ will not be executed until the agent believes that its plan to achieve a has succeeded or failed. This is important as the achievement of b may be dependent on the realization of a.

Alternatively, assume that an agent has the following intention:

$$\iota[\tau : \varpi :\leftarrow \tau'; \pi.\epsilon][\tau' : \varpi' \leftarrow !a; !b; \pi'.\tau' : \langle \{p\}, \{q\} \rangle][+!a : \top \leftarrow \pi''.\epsilon']$$

If the agent comes to believe either p or q prior to the completion of any dependent plans (such that $\pi' \neq \top$ or $\pi'' \neq \top$) then all plans including and above:

$$[\tau' : \varpi' \leftarrow !a; !b; \pi'.\tau' : \langle \{p\}, \{q\} \rangle]$$

can be dropped. This results in the following intention:

$$\iota[\tau : \varpi :\leftarrow \pi.\epsilon]$$

Were any of the above plans to contain multiple effects, it would be necessary for all the success conditions to be realized (though not at the same instant) prior to dropping it. In contrast, only one effect needs to have failed before a plan should be abandoned.

The extended semantics give rise to the architecture depicted in figure 4.1 and the state transition system of figure 4.2.

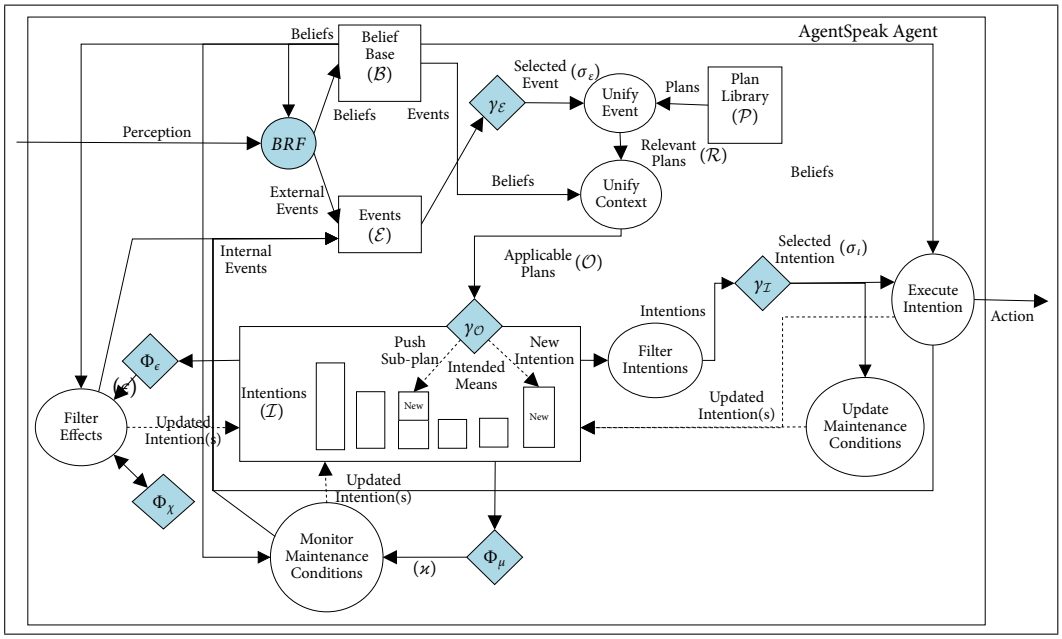


Figure 4.1: Extended AgentSpeak Architecture

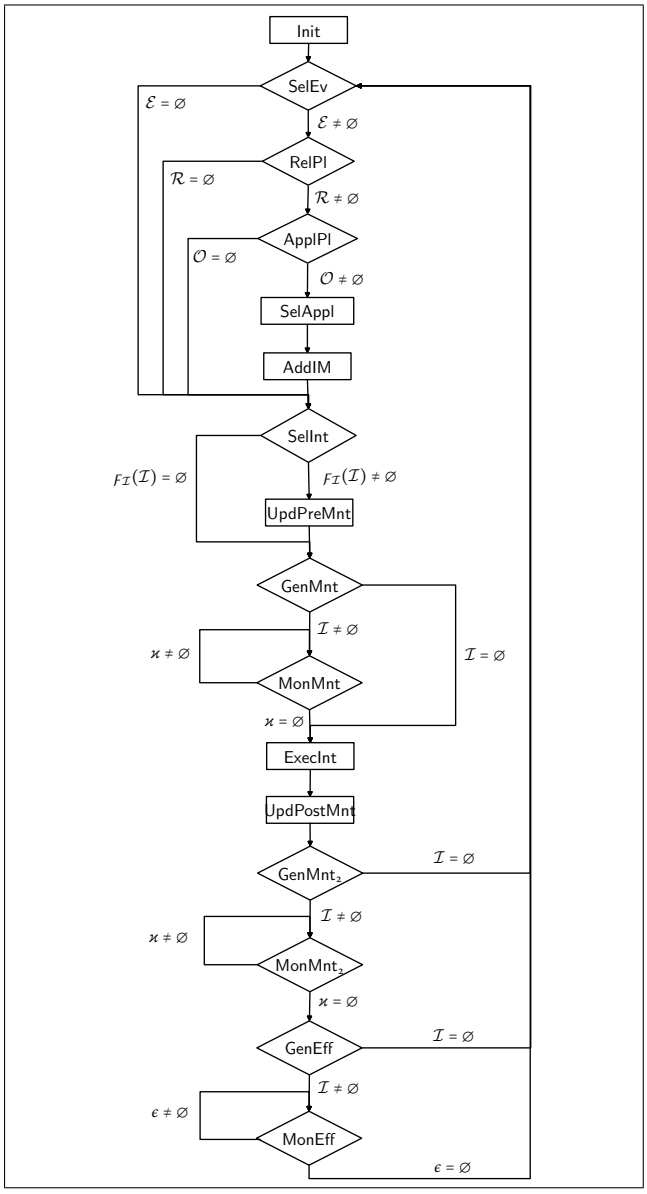


Figure 4.2: Extended AgentSpeak State Transition Diagram

4.2 Monitoring Strategies

In the semantics above, monitoring strategies are required for two tasks: effect monitoring and maintenance condition checking. An effect monitoring strategy was defined as a function that maps the current state of an agent and a particular effect within a particular plan within a particular intention to whether or not this effect should be checked. Similarly, a maintenance condition checking strategy was defined as a function that maps the current state of an agent and a particular maintenance condition in a particular plan within a particular intention to whether or not this maintenance condition should be checked. This particular definition is justified by demonstrating how it captures a number of simple strategies. Example strategies that an agent designer may wish to utilize include:

(4.2.1) Definition (Monitor on Plan Execution Completion):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \begin{cases} \top & \textit{if } \rho = \tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon \wedge \pi = \varnothing \wedge T_{\gamma_\iota} = \iota = \iota'[\rho] \\ \bot & \textit{otherwise} \end{cases}$$

Definition 4.2.1 facilitates the monitoring of the outcomes of a plan whenever the plan in question completes execution. In combination with plans specified with no outcome conditions, definition 4.2.1 can capture the behaviour of AgentSpeak agents without monitoring facilities. Each plan will only be monitored when it completes execution, which in combination with the empty set of effects will cause the application of rule 4.1.14. This will result in the removal of the plan from the set of intentions. This strategy is not applicable to the monitoring of maintenance conditions because at the time of plan completion, the maintenance conditions contained therein cease to have any meaning.

(4.2.2) Definition (Monitor Post Plan Execution Completion):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \begin{cases} \top & \textit{if } \tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon = \rho \wedge \pi = \varnothing \wedge \iota = \iota'[\rho] \\ \bot & \textit{otherwise} \end{cases}$$

By monitoring only on the completion of a plan (definition 4.2.1), the temporal window for the detection of either the success or failure of a plan is limited. This creates a substantial risk that one of the conditions may be satisfied temporarily prior to, or significantly after the agent tests it. In such circumstances the plan will be held indefinitely as no further monitoring that can remove it from the set of intentions will be conducted. Definition 4.2.2 overcomes this limitation by extending

monitoring of completed plans into the future as far as necessary to determine a result.

(4.2.3) Definition (Monitor Intention on Plan Execution Completion):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \begin{cases} \top & \textit{if } \tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon = \rho \wedge \pi = \varnothing \wedge \iota = \iota'[\rho]\iota'' = T_{\iota} \\ \bot & \textit{otherwise} \end{cases}$$

Definition 4.2.3 extends the reach of the monitoring of definition 4.2.1 by monitoring, not only the plan just executed, but all plans in the intention stack in which that plan is stored. This form of monitoring allows an agent to detect the success or failure of plans deeper in the intention stack and shortcut the execution of the intervening plans. Again, like definition 4.2.1, this monitoring strategy is invalid for the monitoring of maintenance conditions due to the limitation of monitoring to after the plans completion at which point the condition ceases to be meaningful. Additionally, the issues with regard to the monitoring window of definition 4.2.1 apply to this rule as well. Definition 4.2.4 overcomes this limitation.

(4.2.4) Definition (Monitor Intention Post Plan Execution Completion):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \begin{cases} \top & \textit{if } \tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon = \rho \wedge \pi = \varnothing \wedge \iota = \iota'[\rho]\iota'' \\ \bot & \textit{otherwise} \end{cases}$$

(4.2.5) Definition (Monitor on Sub-Goal Completion):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \begin{cases} \top & \textit{if } \iota = T_{\sigma_{\iota}} = \iota'[\rho] \\ \bot & \textit{otherwise} \end{cases}$$

$$\Phi_{\mu}(C, T, \iota, \rho, \mu) = \begin{cases} \top & \textit{if } \iota = T_{\sigma_{\iota}} = \iota'[\rho] \\ \bot & \textit{otherwise} \end{cases}$$

Definition 4.2.5 requires the monitoring of the top-most plan of the selected intention after each goal execution within the plan. This is more extensive monitoring and facilitates the recognition of the early success or failure of the plan prior to completion.

4.2.6 Definition (Monitor Intention on Sub-Goal Completion):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \begin{cases} \top & \text{if } \iota = T\sigma_{\iota} = \iota'[\rho]\iota'' \\ \bot & \text{otherwise} \end{cases}$$

$$\Phi_{\mu}(C, T, \iota, \rho, \mu) = \begin{cases} \top & \text{if } \iota = T\sigma_{\iota} = \iota'[\rho]\iota'' \\ \bot & \text{otherwise} \end{cases}$$

By monitoring all levels of an intention using definition 4.2.6, an agent not only facilitates the early detection of the success or failure of the currently executing plan, but all the plans that led to its adoption.

4.2.7 Definition (Monitor each Reasoning Cycle):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) \begin{cases} \top & \text{if } \iota[\rho] \in C_{\mathcal{I}} \\ \bot & \text{otherwise} \end{cases}$$

$$\Phi_{\mu}(C, T, \iota, \rho, \mu) \begin{cases} \top & \text{if } \iota[\rho] \in C_{\mathcal{I}} \\ \bot & \text{otherwise} \end{cases}$$

Monitoring each reasoning cycle (definition 4.2.7) selects the top-most plan of all intention stacks for monitoring. This moves the focus of the agent from the context of the most recently executed plan to the most recently executed plans across all its intentions. This allows for shallow detection of beneficial interactions between the execution of intentions, without resorting to the additional cost of complete monitoring.

4.2.8 Definition (Always Monitor):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \top$$
$$\Phi_{\mu}(C, T, \iota, \rho, \mu) = \top$$

Complete monitoring (definition 4.2.8) provides an agent with a strategy aimed to maximize the detection of the early success or failure of its plans. It does so without concern for the costs of doing so. The complement of always monitoring is to never monitor (definition 4.2.9).

(4.2.9) Definition (Never Monitor):

$$\Phi_{\epsilon}(C, T, \iota, \rho, \xi) = \bot$$
$$\Phi_{\mu}(C, T, \iota, \rho, \mu) = \bot$$

While not an exhaustive list, definitions 4.2.1 to 4.2.8, provide a number of examples of strategies that trade monitoring accuracy against the cost of such monitoring.

4.3 Conflict Resolution Strategies

The modified AgentSpeak semantics define a conflict resolution function by which an agent designer may supply a strategy for resolving conflicting evidence within a particular effect of a plan. This function returns one of three values: $\top$ indicating that the effect (ξ) was shown to have evidence for success, $\bot$ indicating there is evidence that this effect has failed, and $\dashv$ indicating that there was evidence neither for nor against the success or failure of this effect. By presenting a number of simple implementations of this function we demonstrate the different personalities an agent can display through modification of this function. The first such example is the pessimism displayed by definition 4.3.1. It is pessimistic in that failure is concluded given any evidence for doing so, success is only declared in the complete absence of any failure evidence, and uncertainty results from an absence of both.

(4.3.1) Definition (Pessimistic Resolution):

$$\Phi_{\chi}(\mathcal{B}, \lambda : \langle \texttt{suc}, \texttt{fail} \rangle) = \begin{cases} \top & \textit{if } \exists \beta \in \texttt{suc} \,.\, \beta \in Cn(\mathcal{B}) \wedge \forall \beta' \in \texttt{fail} \,.\, \beta' \notin Cn(\mathcal{B}) \\ \bot & \textit{if } \exists \beta \in \texttt{fail} \,.\, \beta \in Cn(\mathcal{B}) \\ \dashv & \textit{otherwise} \end{cases}$$

The dual of definition 4.3.1 is definition 4.3.2. Using such a strategy, an agent disregards all evidence indicative of the failure of its behaviour in circumstances in which there is any evidence for its success.

(4.3.2) Definition (Optimistic Resolution):

$$\Phi_{\chi}(\mathcal{B}, \lambda : \langle \texttt{suc}, \texttt{fail} \rangle) = \begin{cases} \top & \textit{if } \exists \beta \in \texttt{suc} \,.\, \beta \in Cn(\mathcal{B}) \\ \bot & \textit{if } \exists \beta \in \texttt{fail} \,.\, \beta \in Cn(\mathcal{B}) \wedge \forall \beta' \in \texttt{suc} \,.\, \beta' \notin Cn(\mathcal{B}) \\ \dashv & \textit{otherwise} \end{cases}$$

Both definitions 4.3.1 and 4.3.2 disregard the amount of evidence relevant to any conclusion. To remedy this, relative resolution (definition 4.3.3) concludes the outcome with strictly the most evidence for its realization. In scenarios with equal evidence for both outcomes, the decision is delayed pending further information. This perspective on conflict allows an agent to remain sceptical of the evidence it has. By making the comparison strict an agent must have unambiguous information for making a conclusion in one direction or the other. One limitation of this particular strategy is that each piece of evidence is weighted equally. However, introduction of unequal weighting on individual conditions is not precluded by the formalism as presented.

(4.3.3) Definition (Relative Resolution):

$$\Phi_{\chi}(\mathcal{B}, \lambda : \langle \texttt{suc}, \texttt{fail} \rangle) = \begin{cases} \top & \textit{if } |cons(\texttt{suc}, \mathcal{B})| > |cons(\texttt{fail}, \mathcal{B})| \\ \bot & \textit{if } |cons(\texttt{suc}, \mathcal{B})| < |cons(\texttt{fail}, \mathcal{B})| \\ \dashv & \textit{otherwise} \end{cases}$$

where:

$$cons(X, Y) = \{ x \in X \mid x \in Cn(Y) \}$$

While remaining sceptical towards the evidence pertaining to a given conflict is conservative and likely to result in the most accurate depiction of the environment, there are costs associated with remaining so. Delaying decisions requires additional monitoring, slower progression of held intentions and an assumption that additional information that will resolve the ambiguity is forthcoming. In some circumstances, such an assumption may be invalid or the additional costs too high. In these cases either definition 4.3.4 or definition 4.3.5 may be most appropriate. Note that in both cases it is necessary to check that there is pertinent evidence available before resolving the conflict. Were this not the case, the optimistic approach would conclude success and, conversely, the pessimistic approach would conclude failure without any evidence for either.

(4.3.4) Definition (Optimistic Relative Resolution):

$$\Phi_{\chi}(\mathcal{B}, \lambda : \langle \texttt{suc}, \texttt{fail} \rangle) = \begin{cases} \top & \textit{if } |cons(\texttt{suc}, \mathcal{B})| \geq |cons(\texttt{fail}, \mathcal{B})| > 0 \\ \bot & \textit{if } 0 < |cons(\texttt{suc}, \mathcal{B})| < |cons(\texttt{fail}, \mathcal{B})| \\ \dashv & \textit{otherwise} \end{cases}$$

where:

$$cons(X, Y) = \{x \in X \mid x \in Cn(Y)\}$$

(4.3.5) Definition (Pessimistic Relative Resolution):

$$\Phi_{\chi}(\mathcal{B}, \lambda : \langle \texttt{suc}, \texttt{fail} \rangle) = \begin{cases} \top & \textit{if } 0 < |cons(\texttt{suc}, \mathcal{B})| > |cons(\texttt{fail}, \mathcal{B})| \\ \bot & \textit{if } |cons(\texttt{suc}, \mathcal{B})| \leq |cons(\texttt{fail}, \mathcal{B})| > 0 \\ \dashv & \textit{otherwise} \end{cases}$$

where:

$$cons(X, Y) = \{x \in X \mid x \in Cn(Y)\}$$

4.4 Summary

This chapter has adopted the representation of maintenance and outcome conditions of chapter 3 and generalized the semantics of AgentSpeak to accommodate them. In doing so it was necessary to provide mechanisms for the balance of the accuracy of the monitoring against the additional costs involved in meeting these accuracy requirements. A number of simple strategies for making this trade-off were proposed and their properties discussed. This led to the observation that the representation adopted suffered issues surrounding conflicting evidence. Again, the AgentSpeak semantics was extended to provide a customisable approach to resolve the issue. Subsequently, a number of simple example strategies were presented to manage conflict resolution and their merits analysed. Given an agent's sensitivity to the environment to which it is deployed, providing additional entry points for customization instead of advocating single approaches to overcome these difficulties is advantageous. It also facilitates the development of more sophisticated or general approaches while remaining consistent with both AgentSpeak and Jason in providing user definable behaviour. Future work will aim to generate empirical evidence towards the assessment of the validity of the approaches detailed.

They're giving bank robbing a bad name.
John Dillinger, on Bonnie and Clyde

5
Belief Revision

Hypothesis:
It is rational for an agent to revise its beliefs in accordance with the intended outcomes and constraints of its behaviours.

The mental attitudes involved in the practical reasoning of an autonomous entity are interdependent. Perception and belief influence the opportunities an agent considers available to it. The behaviours to which an agent is committed are tracked by its ongoing perception and belief revisions. The behaviours an agent pursues affect its place in its environment and subsequently the perceptions and beliefs it holds. In scenarios in which an agent has uncertainty about the true state of the environment, means of belief acceptance are required for the progress of practical reasoning. In this chapter the role of intention dynamics in the dynamics of belief will be investigated. As evolving beliefs change the commitments an agent holds, this evolution can be controlled to ensure that, whenever rational, the agent's beliefs evolve to support its commitments[19]. Three different classes of interaction between intention and belief will be considered:

1. Each intention that an agent holds is dependent on the agent holding a given belief. Should the agent cease to believe a depended upon belief, then all the dependent intentions must be dropped. (See section 5.1).

2. Each intention that an agent holds persists by default, contains a condition which when believed by the agent indicates that the intention has been successful and contains a condition which when believed indicates the failure or invalidity of the intention. (See section 5.2).

3. Additional structure is imposed on the intention base by assuming that there is a priority relation over the intentions. Intentions and beliefs are related as described above. (See section 5.3).

In each of these scenarios the relationship between beliefs and intentions will be formalized. Based on this relationship a preference ordering over potential belief revisions based on their effects on the intentions of the agent will be defined. The validity of this doxastic preference relation will then be assessed. In doing so, the rationality of each with respect to the classical belief revision framework outlined in section 2.3.1 will be established.

5.1 Maintenance Conditions

Below, the mechanisms by which an intentional agent's intentions relate to its beliefs are formalized and a number of auxiliary concepts necessary for establishing the rationality of the preference relations proposed are presented.

Firstly, the conditions under which it is valid to say that the intentions held by an agent are grounded in its beliefs is defined (definition 5.1.1). For an agent to hold belief grounded intentions it is necessary for all of its held intentions to be conditional on some subset of the agent's currently held beliefs. Intentions cannot be adopted without their condition upon the agent's beliefs being satisfied. Should the agent's beliefs be revised so as to no longer support a given intention, the dependent intention must be dropped. Thus, an intention can only be rationally retained while ever its associated belief is held. For the remainder of this section, we will be concerned only with agents whose intentions are grounded in its beliefs at all times.

(5.1.1) Definition (Agent with Belief Grounded Intentions):
An agent's intentions ($\mathcal{I}$) are grounded in (conditional with respect to) its beliefs ($\mathcal{B}$) if and only if:

1. *they are of the form $[\iota : \texttt{while}(\beta)]$ where $\beta \in \mathcal{L}_{\mathcal{B}}$ and $\iota \in \mathcal{L}_{\mathcal{I}}$*
2. *for all $[\iota : \texttt{while}(\beta)] \in \mathcal{I}$, if $\mathcal{B}$ is the agent's current belief base and $\beta \in Cn(\mathcal{B})$, then $[\iota : \texttt{while}(\beta)] \in \mathcal{I}'$ where $\mathcal{I}'$ is the current set of intentions revised relative to the agent's new beliefs.*
3. *Alternatively, for all $[\iota : \texttt{while}(\beta)] \in \mathcal{I}$, if $\beta \notin Cn(\mathcal{B})$ then $[\iota : \texttt{while}(\beta)] \notin \mathcal{I}'$.*

Next, the subset of a set of intentions that are supported by a given set of beliefs is defined (definition 5.1.2). Unlike the definition for agent belief grounded intentions, there are no constraints imposed on valid agent configurations. This definition pertains to arbitrary sets of belief conditional intentions and beliefs, regardless

of the set of beliefs or intentions currently held by any given agent. It should be noted that for agents with belief grounded intentions, at all times $(\mathcal{I}\perp\mathcal{B}) = \mathcal{I}$ provided $\mathcal{B}$ and $\mathcal{I}$ are the agent's beliefs and intentions, respectively, at the beginning of each of its reasoning cycles (before the agent has revised its beliefs in accordance with any new perceptions).

5.1.2 Definition (Belief Supported Intentions):
The set of intentions supported given a belief base $(\mathcal{B})$*, an initial set of intentions* $(\mathcal{I})$ *and the consequence operator used to derive new beliefs from old* (Cn) *is the set* $(\mathcal{I}\perp\mathcal{B})$ *where:*

$$(\mathcal{I}\perp\mathcal{B}) = \{[\iota : \texttt{while}(\beta)] \in \mathcal{I} \mid \beta \in Cn(\mathcal{B})\}$$

Given a pair of belief bases and a set of intentions, a relation that orders the belief bases according to the support they provide for the input intention set is defined. A belief base is more preferred than another provided it provides more support for the input set of intentions than the other. This preference relation will provided the mechanism by which belief bases are compared by the selection function below.

5.1.3 Definition (Minimal Intention Change Relation):
If $\mathcal{B}$ *and* $\mathcal{B}'$ *are belief bases and* $\mathcal{I}$ *is an intention base then* $(\sqsubseteq^{\texttt{dif}}_{\mathcal{I}})$ *is a preference relation over belief bases given* $\mathcal{I}$ *if:*

$$\mathcal{B} \sqsubseteq^{\texttt{dif}}_{\mathcal{I}} \mathcal{B}' \text{ if and only if } (\mathcal{I}\perp\mathcal{B}) \subseteq (\mathcal{I}\perp\mathcal{B}')$$

Now that arbitrary belief bases can be compared according to the support they provide for arbitrary intention sets, a function that will select the subset of these belief bases that most support the aforementioned intention set can be constructed. This function will return a subset of the input set of belief bases. This is necessary as it is possible that different belief bases may support the same number of intentions. For example, if two belief bases differ only in their valuation of beliefs that are unrelated to any of the input intentions then both will support the same intentions though their contents differ. Alternatively, it is possible that two belief bases support the same number of intentions, but different subsets of the input intention set.

5.1.4 Definition (Minimal Intention Change Selection Function):
A selection function $(\gamma^{\mathcal{I}}_{\texttt{dif}})$ *is a* minimal intention change *selection function given*

a set of belief bases $(\mathbb{B})$ *and an intention base* $(\mathcal{I})$ *based on a* minimal intention change *relation if and only if:*

$$\gamma^{\mathcal{I}}_{\texttt{dif}}(\mathbb{B},\mathcal{I}) = \{\mathcal{B} \in \mathbb{B} \mid \mathcal{B}' \sqsubseteq^{\texttt{dif}}_{\mathcal{I}} \mathcal{B} \text{ for all } \mathcal{B}' \in \mathbb{B}\}$$

Finally, it is possible to define a revision based on the selection function above. This revision is defined accordance with the belief revision framework presented in section 2.3.1. In particular, it is an external partial meet revision operator definition 2.3.12. This is necessary so that the selection function has access to the belief added as well as any that must be dropped in order to maintain consistency. Were it an internal partial meet revision then the selection function would have access only to the belief bases with the inconsistencies removed but not the new belief.

(5.1.5) Definition (Minimal Intention Change Belief Revision):
A minimal intention change *belief revision* $\left(\mathcal{B} \pm_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \beta\right)$ *given an intention base* $(\mathcal{I})$ *is the external partial meet of the selected belief bases as selected via a* minimal intention change *selection function* $(\gamma^{\mathcal{I}}_{\texttt{dif}})$ *where each option is an element of the belief remainder set* $(\mathcal{B} \perp\!\!\!\perp \neg\beta)$*[2] union the new belief* (β)*:*

$$\mathcal{B} \pm_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \beta = \left(\mathcal{B} +_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \beta\right) \sim_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \neg\beta \tag{5.1}$$

$$\mathcal{B} +_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \beta = \mathcal{B} \cup \{\beta\} \tag{5.2}$$

$$\mathcal{B} \sim_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \beta = \bigcap \gamma^{\mathcal{I}}_{\texttt{dif}}(\mathcal{B} \perp\!\!\!\perp \beta, \mathcal{I}) \tag{5.3}$$

Applying this belief revision operator to example 3.1.1 it can be seen that:

(1.) Tom's initial beliefs are:

$$\mathcal{B} = \left\{ \begin{array}{c} \neg\text{'Savings increase'}, \neg\text{'Mortgage increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Savings increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Mortgage increase'} \end{array} \right\}$$

(2.) His intentions are:

$$\mathcal{I} = \{[\text{'Minimal repayments'} : \texttt{while}\,(\neg\text{'Mortgage increase'})]\}$$

(3.) He is making the following revision:

$$\mathcal{B} \pm_{\gamma^{\mathcal{I}}_{\texttt{dif}}} \text{'Interest rate rise'}$$

(4.) After expanding his beliefs with 'Interest rate rise' in accordance with equation 5.2 his beliefs are:

$$\mathcal{B} \cup \{\text{'Interest rate rise'}\} = \left\{ \begin{array}{c} \neg\text{'Savings increase'}, \neg\text{'Mortgage increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Savings increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Mortgage increase'} \\ \text{'Interest rate rise'} \end{array} \right\}$$

(5.) In order to generate the remainder set

$$(\mathcal{B} \cup \{\text{'Interest rate rise'}\}) \perp\!\!\!\perp \neg\text{'Interest rate rise'}$$

as required by equation 5.3, it is necessary to find the subsets $\mathcal{B}'$ of

$$\mathcal{B} \cup \{\text{'Interest rate rise'}\}$$

such that:

(a) $\mathcal{B}' \subseteq \mathcal{B} \cup \{\text{'Interest rate rise'}\}$

(b) $Cn(\mathcal{B}') \cap \neg\text{'Interest rate rise'} = \varnothing$

(c) There is no set $\mathcal{B}''$ such that $\mathcal{B}' \subset \mathcal{B}'' \subseteq (\mathcal{B} \cup \{\text{'Interest rate rise'}\})$ and $Cn(\mathcal{B}'') \cap \neg\text{'Interest rate rise'} = \varnothing$

Since $\neg$'Savings increase' $\in Cn(\mathcal{B} \cup \{\text{'Interest rate rise'}\})$ and 'Savings increase' $\in Cn(\mathcal{B} \cup \{\text{'Interest rate rise'}\})$ it is necessary to remove either:

$$\neg\text{'Savings increase'}, \text{'Interest rate rise'} \rightarrow \text{'Savings increase'},$$

or,

$$\text{'Interest rate rise'}$$

from $\mathcal{B} \cup \{\text{'Interest rate rise'}\}$ to restore consistency. This is because from contradiction anything can be derived ($\perp \rightarrow \beta$ for all $\beta \in \mathcal{L}_{\mathcal{B}}$). This includes $\neg$'Interest rate rise' which in turn violates the requirement expressed by item 5b. Similar reasoning applies to the contradiction surrounding the derivability of both 'Mortgage increase' and $\neg$'Mortgage increase'. Consequently, the set $(\mathcal{B} \cup \{\text{'Interest rate rise'}\}) \perp\!\!\!\perp \neg\text{'Interest rate rise'}$ is the set containing:

$$\mathcal{B}_1 = \left\{ \begin{array}{c} \text{'Interest rate rise'}, \neg\text{'Mortgage increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Savings increase'} \end{array} \right\}$$

$$\mathcal{B}_2 = \left\{ \begin{array}{c} \text{'Interest rate rise'}, \neg\text{'Savings increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Mortgage increase'} \end{array} \right\}$$

$$\mathcal{B}_3 = \left\{ \begin{array}{c} \text{'Interest rate rise'} \rightarrow \text{'Mortgage increase'}, \\ \text{'Interest rate rise'} \rightarrow \text{'Savings increase'}, \\ \text{'Interest rate rise'} \end{array} \right\}$$

$$\mathcal{B}_4 = \left\{ \begin{array}{c} \text{'Interest rate rise'}, \neg\text{'Savings increase'}, \\ \neg\text{'Mortgage increase'} \end{array} \right\}$$

(6.) The intention remainder sets given the above belief remainder sets are:

$$\mathcal{I} \perp\!\!\!\perp \mathcal{B}_1 = \{[\text{'Minimal repayments'} : \texttt{while}\,(\neg\text{'Mortgage increase'})]\}$$

since $\neg$'Mortgage increase' $\in Cn(\mathcal{B}_1)$. By the same reasoning:

$$\mathcal{I} \perp\!\!\!\perp \mathcal{B}_2 = \varnothing, \qquad \mathcal{I} \perp\!\!\!\perp \mathcal{B}_3 = \varnothing,$$
$$\mathcal{I} \perp\!\!\!\perp \mathcal{B}_4 = \{[\text{'Minimal repayments'} : \texttt{while}\,(\neg\text{'Mortgage increase'})]\}$$

(7.) The above intention remainder sets satisfy the following subset relationships:

$$\mathcal{I} \perp\!\!\!\perp \mathcal{B}_2 = \mathcal{I} \perp\!\!\!\perp \mathcal{B}_3 \subset \mathcal{I} \perp\!\!\!\perp \mathcal{B}_1 = \mathcal{I} \perp\!\!\!\perp \mathcal{B}_4$$

and therefore satisfy the following minimum intention change relationships:

$$\mathcal{I} \perp\!\!\!\perp \mathcal{B}_2 \sqsubseteq^{\texttt{dif}}_{\mathcal{I}} \mathcal{I} \perp\!\!\!\perp \mathcal{B}_3 \sqsubset^{\texttt{dif}}_{\mathcal{I}} \mathcal{I} \perp\!\!\!\perp \mathcal{B}_1 \sqsubseteq^{\texttt{dif}}_{\mathcal{I}} \mathcal{I} \perp\!\!\!\perp \mathcal{B}_4$$

(8.) The minimum intention change selection function based on the above ordering of intention remainder sets will select $\mathcal{B}_1$ and $\mathcal{B}_4$:

$$\gamma^{\mathcal{I}}_{\texttt{dif}}(\{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}, \mathcal{I}) = \{\mathcal{B}_1, \mathcal{B}_4\}$$

(9.) Finally, to generate the resulting revision it is necessary to take the intersection of the selected belief remainder sets. Thus, Tom's revised beliefs will be:

$$\begin{aligned} \mathcal{B} \pm_{\gamma^{\mathcal{I}}_{\texttt{dif}}} & \text{'Interest rate rise'} \\ & = \bigcap \gamma^{\mathcal{I}}_{\texttt{dif}}((\mathcal{B} \cup \{\text{'Interest rate rise'}\}) \perp\!\!\!\perp \neg\text{'Interest rate rise'}, \mathcal{I}) \\ & = \bigcap \{\mathcal{B}_1, \mathcal{B}_4\} \\ & = \mathcal{B}_1 \cap \mathcal{B}_4 \\ & = \{\text{'Interest rate rise'}, \neg\text{'Mortgage increase'}\} \end{aligned}$$

Consequently, Tom will be rational in retaining his intention towards making minimal mortgage repayments.

Notice that if the set of intentions is empty, then all belief revisions are equally preferred. If all revisions are equally preferred then the selection function will return all the revisions generated by the remainder set operator. Thus, when there are no intentions the agent will execute a full meet revision (with all the problems of over cautiousness that full meet entails).

5.1.1 Observation:
If the set of intentions is empty then all potential revisions are equally preferred by $\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$.

Proof. It is necessary to show that if $\mathcal{I} = \varnothing$ then for all belief bases $\mathcal{B}$ and $\mathcal{B}'$, $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}$. By definition 5.1.3, $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ if and only if $(\mathcal{I} \perp \mathcal{B}) \subseteq (\mathcal{I} \perp \mathcal{B}')$. If $\mathcal{I} = \varnothing$ then by definition 5.1.2 $(\mathcal{I} \perp \mathcal{B}) = \varnothing$ for all belief bases. Since $\varnothing \subseteq \varnothing$ the proof is complete. □

It must now be verified that $\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$ provides a valid doxastic preference relation. To do so it is necessary to verify that it is transitive and weakly, if not completely, maximizing. These properties are important for the rationality of preference relations. Recall that, as discussed in section 2.3.1 on page 68, selection functions based on relations that are transitive and maximizing satisfy conjunctive inclusion, conjunctive factoring and conjunctive trisection[1] in addition to those rationality constraints satisfied by arbitrary partial-meet revisions (observation 2.3.12).

5.1.1 Theorem:

$\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$ *is transitive.*

Proof. It is crucial to prove that if $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}''$, then $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}''$. By definition 5.1.3 if $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ then $(\mathcal{I} \perp \mathcal{B}) \subseteq (\mathcal{I} \perp \mathcal{B}')$ and similarly if $\mathcal{B}' \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}''$ then $(\mathcal{I} \perp \mathcal{B}') \subseteq (\mathcal{I} \perp \mathcal{B}''))$. Therefore, $(\mathcal{I} \perp \mathcal{B}) \subseteq (\mathcal{I} \perp \mathcal{B}'')$ and $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}''$ as required. □

5.1.2 Theorem:

$\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$ *is weakly maximizing.*

Proof. It must be shown that if $\mathcal{B} \subset \mathcal{B}'$, then $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$. If $\mathcal{B} \subset \mathcal{B}'$ then $Cn(\mathcal{B}) \subset Cn(\mathcal{B}')$ by the monotonicity of Cn. Thus:

$$\{[\iota : \mathtt{while}(\beta)] \in \mathcal{I} \mid \beta \in Cn(\mathcal{B})\} \subseteq \{[\iota : \mathtt{while}(\beta)] \in \mathcal{I} \mid \beta \in Cn(\mathcal{B}')\}$$

By definitions 5.1.2 and 5.1.3, $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ follows as required. □

[1] Provided some additional conditions are met.

It is not possible to strengthen this to show that $\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$ is maximizing.

5.1.3 Theorem:

$\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$ is not maximizing.

Counter-Example. $\mathcal{B}$ and $\mathcal{B}'$ must be defined such that $\mathcal{B} \subset \mathcal{B}'$ but $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}$. Let:

$$\mathcal{B} = \{\beta\} \qquad \mathcal{B}' = \{\beta, \beta'\} \qquad \beta' \notin Cn(\beta) \qquad \mathcal{I} = \{[\iota : \mathtt{while}(\beta)]\}$$

As can be seen from the definitions of $\mathcal{B}$ and $\mathcal{B}'$ that $\mathcal{B} \subset \mathcal{B}'$ and the same number of intentions will be retained given either belief base by definition 5.1.2. Thus, both $(\mathcal{I} \perp\!\!\!\perp \mathcal{B}) \subseteq (\mathcal{I} \perp\!\!\!\perp \mathcal{B}')$ and $(\mathcal{I} \perp\!\!\!\perp \mathcal{B}') \subseteq (\mathcal{I} \perp\!\!\!\perp \mathcal{B})$ and definition 5.1.3 establishes $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}$ as required. □

5.1.4 Theorem (Hansson [51]):
$\sim_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$ satisfies the properties of success, inclusion, relevance, uniformity, failure, closure, relative closure, core-retainment, vacuity, extensionality and conjunctive overlap by virtue of being a transitive weakly maximizingly relational partial meet contraction.

5.1.5 Theorem (Hansson [52]):
$\pm_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$ satisfies the properties of consistency, inclusion, relevance, success, weak uniformity and pre-expansion by virtue of being an external partial meet revision.

The relationship between intentions and beliefs as advocated in this section is not the traditional relationship between beliefs and intentions. It does not capture the inherent stability of intentions as each intention is only as stable as the belief it utilizes as its maintenance condition.

5.2 Outcome Conditions

In this section the case where the beliefs and intentions of an agent are related via success and failure conditions is investigated. Intentions persist by default. Each intention has a success and failure condition associated with it. The success condition is a condition on the agent's beliefs that when satisfied indicates that the intention has been successful. At this point the agent can drop the intention and continue in the knowledge that the intention achieved its ends. Similarly, the failure condition of an intention indicates the conditions under which the intention has failed. Thus,

the agent may take remedial steps, execute a recovery procedure or accept the failure of its intention as an outright loss and move on to other behaviours. Given that the beliefs of an agent, at a minimum, encapsulate its information regarding the environment, this relationship between intentions and beliefs ensures that the agent retains a realistic perspective on its progress towards its ends.

Example 3.2.1 motivates the following relationship between an agent's intentions and beliefs and its preferences towards the effect of belief revision on its intentions.

5.2.1 Definition (Belief Grounded Intentions):
An agent's intentions ($\mathcal{I}$) are grounded in (conditional with respect to) its beliefs ($\mathcal{B}$) if and only if:

1. *they are of the form $[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')]$ where $\beta \in \mathcal{L}_{\mathcal{B}}$, $\beta' \in \mathcal{L}_{\mathcal{B}}$ and $\iota \in \mathcal{L}_{\mathcal{I}}$*

2. *for all $[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I}$ if $\beta \in Cn(\mathcal{B})$ or $\beta' \in Cn(\mathcal{B})$, then $\iota \notin \mathcal{I}'$ where $\mathcal{I}'$ is the intention base revised to account for the changes in the agent's beliefs.*

An agent can select the belief revision that, given a set of intentions:

1. maximizes the number of successful intentions, then within those maximally successful intentions minimizes the number of failed intentions $(>_{\mathtt{suc}}) > (<_{\mathtt{fail}})$. If both the success and failure conditions are satisfied by the agent's beliefs then the intention is considered successful, or

2. minimizes the number of failed intentions prior to maximizing the successful intentions within the minimally failing intentions $(<_{\mathtt{fail}}) > (>_{\mathtt{suc}})$. If both the success and failure conditions are satisfied by the agent's beliefs then the intention is considered to have failed, or

3. minimizes the change required of its intentions in the hope that further information will become available so that it need not decide prematurely $(<_{\mathtt{dif}})$.

Alternative 3 may appear to be a strange strategy for an agent to adopt. However, there may be a number of situations in which it is the most appropriate. In environments in which perception is noisy, placing a priority on the stability of the agents intentions over success may prevent over-optimism and buffer the agent against faulty changes in its intentions due to incorrect perception. In situations in which an agent must interact and cooperate with other agents, the additional

predictability of the agents behaviour subsequent to prioritizing the stability of intention over success may be the determinant factor in whether cooperation is attempted or successfully achieved.

It is necessary to define the sets of intentions rendered a success/failure by a given belief base, the duals of these sets, the set of beliefs relevant for a given intention base, preference relations over belief bases, selection functions based on these preference relations and belief revision operators in terms of these selection functions:

5.2.2 Definition (Intention Remainder Operators):
A successful/failure intention remainder operator $\left(\frac{\perp}{\div}_{\mathtt{suc}}\right)$ / $\left(\frac{\perp}{\div}_{\mathtt{fail}}\right)$ *is an infix operator that requires an intention base* ($\mathcal{I}$) *and a belief base* ($\mathcal{B}$) *as input and returns the elements of the intention base for which the success/failure condition do not follow from the belief base:*

$$\left(\mathcal{I}\frac{\perp}{\div}_{\mathtt{suc}}\mathcal{B}\right) = \{[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I} \mid \beta \notin Cn(\mathcal{B})\}$$

and

$$\left(\mathcal{I}\frac{\perp}{\div}_{\mathtt{fail}}\mathcal{B}\right) = \{[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I} \mid \beta' \notin Cn(\mathcal{B})\}$$

The dual of these operators are the successful/failure intention operators:

$$\left(\mathcal{I}\frac{\perp}{\div}_{\mathtt{suc}}\mathcal{B}\right) = \{[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I} \mid \beta \in Cn(\mathcal{B})\}$$

and

$$\left(\mathcal{I}_{\substack{\mathtt{fail}\\ \perp}}\mathcal{B}\right) = \{[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I} \mid \beta' \in Cn(\mathcal{B})\}$$

5.2.3 Definition (Intention Relevant Beliefs):
The beliefs relevant to the success of the agent's intentions $\mathcal{I}$[2]*:*

$$\mathcal{B}^{\mathcal{I}}_{\mathtt{suc}} = \{\beta \mid [\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I}\}$$

and, to their failure:

$$\mathcal{B}^{\mathcal{I}}_{\mathtt{fail}} = \{\beta' \mid [\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I}\}$$

[2]The set of success and failure conditions are each being treated as a single condition to simplify their presentation. Extending the definitions to cater for sets of conditions is trivial, however.

5.2.4 Definition (Successful Intentions):
$\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ if the successful intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is a superset of the successful intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$. Otherwise, $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ if the successful intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is equal to the successful intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$ and the failure intention remainder set resultant from the successful intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is a subset or equal to the failure intention remainder set applied to the successful intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$.

$$\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}' = \begin{cases} \top & \text{if } \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right) \supset \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}'\right) \\ \top & \text{if } \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right) = \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}'\right) \wedge \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right)\frac{\perp}{\mathtt{fail}}\mathcal{B} \subseteq \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}'\right)\frac{\perp}{\mathtt{fail}}\mathcal{B}' \\ \perp & \text{otherwise} \end{cases}$$

This captures the intuition that $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ if the set of successful intentions given $\mathcal{B}$ is a strict subset of those rendered a success by $\mathcal{B}'$, or, the set of successful intentions is the same for both belief bases and the set of failed intentions among those remaining after removing the successful ones for $\mathcal{B}$ is a superset of those for $\mathcal{B}'$. Conversely:

5.2.5 Definition (Failed Intentions):
$\mathcal{B} \sqsubseteq^{\mathtt{fail}}_{\mathcal{I}} \mathcal{B}'$ if the failure intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is a subset of the failure intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$. Otherwise, $\mathcal{B} \sqsubseteq^{\mathtt{fail}}_{\mathcal{I}} \mathcal{B}'$ if the failure intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is equal to the failure intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$ and the successful intention remainder set resultant from the failure intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is a subset or equal to the successful intention remainder set applied to the failure intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$.

$$\mathcal{B} \sqsubseteq^{\mathtt{fail}}_{\mathcal{I}} \mathcal{B}' = \begin{cases} \top & \text{if } \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}\right) \subset \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}'\right) \\ \top & \text{if } \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}\right) = \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}'\right) \wedge \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}\right)\frac{\perp}{\mathtt{suc}}\mathcal{B} \supseteq \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}'\right)\frac{\perp}{\mathtt{suc}}\mathcal{B}' \\ \perp & \text{otherwise} \end{cases}$$

5.2.6 Definition (Minimal Intention Change Ralation):
$\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ if the failure intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is a subset or is equal to the failure intention remainder set given $\mathcal{B}'$ and $\mathcal{I}$ and the successful intention remainder set given $\mathcal{B}$ and $\mathcal{I}$ is a subset or is equal to the successful intention remainder

set given $\mathcal{B}'$ and $\mathcal{I}$.

$$\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}' = \begin{cases} \top & \text{if } \left(\mathcal{I}^{\perp}_{\overline{\mathtt{fail}}}\mathcal{B}\right) \subseteq \left(\mathcal{I}^{\perp}_{\overline{\mathtt{fail}}}\mathcal{B}'\right) \wedge \left(\mathcal{I}^{\perp}_{\overline{\mathtt{suc}}}\mathcal{B}\right) \subseteq \left(\mathcal{I}^{\perp}_{\overline{\mathtt{suc}}}\mathcal{B}'\right) \\ \bot & \text{otherwise} \end{cases}$$

This captures the fact that $\mathcal{B} \sqsubseteq^{\mathtt{dif}}_{\mathcal{I}} \mathcal{B}'$ if both the set of successful and failed intentions for $\mathcal{B}'$ is a subset of those for $\mathcal{B}$.

5.2.7 Definition (Intention Change-based Selection Function):
A selection function $(\gamma_\star)$ is an intention change *selection function given a set of belief bases $(\mathbb{B})$ and an intention base $(\mathcal{I})$ based on an* intention change *relation $(\sqsubseteq_\star)$[3] if:*

$$\gamma_\star(\mathbb{B}, \mathcal{I}) = \{\mathcal{B} \in \mathbb{B} \mid \mathcal{B}' \sqsubseteq_\star \mathcal{B} \text{ for all } \mathcal{B}' \in \mathbb{B}\}$$

5.2.8 Definition (Intention Change-based Belief Revision):
An intention change *belief revision $(\mathcal{B} \pm_\star \beta)$ given an intention base $(\mathcal{I})$ is the partial meet of the selected belief bases as selected via an* intention change *selection function $(\star)$ where each option is an element of the belief remainder set $(\mathcal{B} \perp\!\!\!\perp \neg\beta)$ union the new belief (β)[4]:*

$$\mathcal{B} \pm_\star \beta = (\mathcal{B} +_\star \beta) \dot{-}_\star \neg\beta \tag{5.4}$$

$$\mathcal{B} +_\star \beta = \mathcal{B} \cup \{\beta\} \tag{5.5}$$

$$\mathcal{B} \dot{-}_\star \beta = \bigcap \star(\mathcal{B} \perp\!\!\!\perp \beta, \mathcal{I}) \tag{5.6}$$

To demonstrate the above revision operators, it is necessary to apply them to an example. Consider example 3.2.1 in which Tom is attempting to maximize the success of his intentions. Tom's intentions in this case are:

$$\mathcal{I} = \{[\text{'Save money'} : \mathtt{suc}(\text{'Savings increase'}), \mathtt{fail}(\text{'Mortgage increase'})]\}$$

Again, Tom is revising his beliefs to integrate the fact that interest rates have risen:

$$\mathcal{B} \pm_{\gamma^{\mathcal{I}}_{\mathtt{suc}}} \text{'Interest rate rise'}$$

The belief remainder sets are the same as for the minimum intention change revision operator $(\pm_{\gamma^{\mathcal{I}}_{\mathtt{dif}}})$ discussed prior.

$$(\mathcal{B} \cup \{\text{'Interest rate rise'}\}) \perp\!\!\!\perp \neg\text{'Interest rate rise'} = \{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}$$

[3] Where $\star$ is either $\mathtt{suc}$ $(\mathcal{I})$, $\mathtt{fail}$ $(\mathcal{I})$ or $\mathtt{dif}$ $(\mathcal{I})$.
[4] Where $\star$ is $\gamma^{\mathcal{I}}_{\mathtt{suc}}$, $\gamma^{\mathcal{I}}_{\mathtt{fail}}$ or $\gamma^{\mathcal{I}}_{\mathtt{dif}}$.

The successful intention remainder sets are:

$$\mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_1 = \varnothing, \quad \mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_2 = \mathcal{I}, \quad \mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_3 = \varnothing, \quad \mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_4 = \mathcal{I}$$

The failure intention remainder sets given the successful intention remainder sets are:

$$\left(\mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_1\right)\overset{\perp}{\underset{\texttt{fail}}{-}}\mathcal{B}_1 = \varnothing, \qquad \left(\mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_2\right)\overset{\perp}{\underset{\texttt{fail}}{-}}\mathcal{B}_2 = \varnothing,$$
$$\left(\mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_3\right)\overset{\perp}{\underset{\texttt{fail}}{-}}\mathcal{B}_3 = \varnothing, \qquad \left(\mathcal{I}\overset{\perp}{\underset{\texttt{suc}}{-}}\mathcal{B}_4\right)\overset{\perp}{\underset{\texttt{fail}}{-}}\mathcal{B}_4 = \mathcal{I}$$

Based on the intention remainder sets above it can be seen that the successful intention maximizing relation will order the belief remainder sets as below. Note that $=^{\texttt{suc}}_{\mathcal{I}}$ is a shorthand for $\mathcal{B} \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}' \wedge \mathcal{B}' \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}$ for arbitrary belief bases $\mathcal{B}$ and $\mathcal{B}'$.

$$\mathcal{B}_2 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_1, \qquad \mathcal{B}_1 =^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_3, \qquad \mathcal{B}_4 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_1,$$
$$\mathcal{B}_2 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_3, \qquad \mathcal{B}_2 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_4, \qquad \mathcal{B}_4 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_3$$

Which can be more compactly expressed:

$$\mathcal{B}_2 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_4 \sqsubseteq^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_1 =^{\texttt{suc}}_{\mathcal{I}} \mathcal{B}_3$$

Thus, the intention success maximizing selection function will select $\mathcal{B}_1$ and $\mathcal{B}_3$:

$$\gamma^{\mathcal{I}}_{\texttt{suc}}(\{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}, \mathcal{I}) = \{\mathcal{B}_1, \mathcal{B}_3\}$$

Finally, to generate the resulting revision it is necessary to take the intersection of the selected belief bases:

$$\begin{aligned}\mathcal{B} \pm_{\gamma^{\mathcal{I}}_{\texttt{suc}}} \text{`Interest rate rise'} &= \bigcap \gamma^{\mathcal{I}}_{\texttt{suc}}((\mathcal{B} \cup \{\text{`Interest rate rise'}\}) \amalg \neg\text{`Interest rate rise'}, \mathcal{I}) \\ &= \bigcap \{\mathcal{B}_1, \mathcal{B}_3\} \\ &= \mathcal{B}_1 \cap \mathcal{B}_3 \\ &= \{\text{`Interest rate rise'}\}\end{aligned}$$

Notice that whenever the agent does not have any intentions all belief revisions are equally preferred.

5.2.1 Observation:
If the set of intentions is empty then all belief bases are equally preferred by $\sqsubseteq^{\texttt{suc}}_{\mathcal{I}}$, $\sqsubseteq^{\texttt{fail}}_{\mathcal{I}}$ *and* $\sqsubseteq^{\texttt{dif}}_{\mathcal{I}}$.

Proof. It is required to show that if $\mathcal{I} = \varnothing$ then for all belief bases $\mathcal{B}$ and $\mathcal{B}'$, $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$. By definition 5.2.4 there are two ways that $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ can hold and vice versa. Only one case need hold for the relation to hold:

Case 1: $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ if $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)$. Since $\mathcal{I} = \varnothing$ then $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right) = \varnothing$ for any $\mathcal{B}$. Since $\varnothing \not\supset \varnothing$ it is known that the first case does not hold for $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$. Through symmetry it is also established that $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$ does not hold via $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right)$.

Case 2: $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ if $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B} \subseteq \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B}'$. Since $\mathcal{I} = \varnothing$

$$\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B} = \varnothing$$

for any $\mathcal{B}$. Since $\varnothing \subseteq \varnothing$ then $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ by this case. Through symmetry it is the case that $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$ as required.

□

The proofs for $\sqsubseteq^{\mathtt{fail}}_{\mathcal{I}}$ and $\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$ are analogous.

Now it is necessary to demonstrate that the above relations are transitive and at least weakly, if not, completely maximizing.

5.2.1 Theorem:

$\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$ *is transitive.*

Proof. It need be shown that if $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ then $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$. There are four cases to consider:

Case 1: $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ by $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ by $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}''\right)$. The transitivity of $\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$ follows from the transitivity of $\supset$.

Case 2: $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ by $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ by $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B}' \subseteq \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}''\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B}''$. Given definition 5.2.4 $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B}' \subseteq \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}''\right)\overset{\perp}{\underset{\mathtt{fail}}{=}}\mathcal{B}''$ is only used in the case that $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right) = \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}''\right)$ (otherwise $\mathcal{B}'' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$). Since $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right)$ and $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}'\right) = \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}''\right)$, it follows that $\left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}\right) \supset \left(\mathcal{I}\overset{\perp}{\underset{\mathtt{suc}}{=}}\mathcal{B}''\right)$ and $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ as required.

Case 3: $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ by $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B} \subseteq \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ by $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right) \supset \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}''\right)$. Given definition 5.2.4 $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B} \subseteq \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}'$ is used only in the case that $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) = \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)$ (otherwise $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$). Since $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) = \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)$ and $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right) \supset \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}''\right)$ it is established that $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) \supset \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}''\right)$ and $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ as required.

Case 4: $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ by $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B} \subseteq \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}''$ by $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}'$ $\subseteq \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}''\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}''$. The transitivity of $\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$ follows from the transitivity of $\subseteq$.

□

5.2.2 Theorem:

$\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$ *is not weakly maximizing.*

Counter-Example. It must be shown that if $\mathcal{B} \subset \mathcal{B}'$ then it need not be the case that $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$. To do so it is necessary to construct $\mathcal{B}$ and $\mathcal{B}'$ such that $\mathcal{B} \subset \mathcal{B}'$ and $\mathcal{B}' \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$ but not $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$. Since $\mathcal{B} \subset \mathcal{B}'$, $Cn(\mathcal{B}) \subset Cn(\mathcal{B}')$ follows by the monotonicity of Cn. By definition 5.2.2, it follows that $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) \supseteq \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)$. By definition 5.2.4, if $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) \supset \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)$ then $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$. Therefore, it is necessary to construct $\mathcal{B}$ and $\mathcal{B}'$ so that $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) = \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)$ in order to prevent $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ yet maintain $\mathcal{B} \subset \mathcal{B}'$. Since it is required that $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right) = \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)$, $\mathcal{B}$ and $\mathcal{B}'$ must be such that $\left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}'\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}' \subset \left(\mathcal{I}^{\stackrel{\perp}{=}}_{\mathtt{suc}}\mathcal{B}\right)^{\stackrel{\perp}{=}}_{\mathtt{fail}}\mathcal{B}$ by definition 5.2.4 (it cannot be $\subseteq$ lest it also satisfy the condition for $\mathcal{B} \sqsubseteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$). This means that there must be at least one $\beta' \in \mathcal{B}'$ such that: $\beta' \notin \mathcal{B}$ and $[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')] \in \mathcal{I}$ where $\beta \notin \mathcal{B}$. Let:

$$\mathcal{B} = \{\beta\} \qquad \mathcal{B}' = \{\beta, \beta'\} \qquad \beta' \notin Cn(\beta) \qquad \mathcal{I} = \{[\iota : \mathtt{suc}(\beta''), \mathtt{fail}(\beta')]\}$$

It can easily be seen that the above satisfies the constraints required to demonstrate that $\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$ is not a weakly maximizing relation. □

It can be seen that $\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$ is anti-monotonic with respect to failure. By adding beliefs which cause intentions to fail (without affecting the success of any intentions), the belief base is made less preferred.

5.2.3 Theorem:

$\sqsubseteq_{\mathcal{I}}^{\mathtt{fail}}$ *is transitive but not weakly maximizing.*

Proof. Proofs are analogous to theorems 5.2.1 and 5.2.2. □

5.2.4 Theorem:

$\sqsubseteq_{\mathcal{I}}^{\mathtt{dif}}$ *is transitive.*

Proof. The proof is analogous to theorem 5.1.1. □

5.2.5 Theorem:

$\sqsubseteq_{\mathcal{I}}^{\mathtt{dif}}$ *is not weakly maximizing.*

Counter-Example. It is necessary to define $\mathcal{B}$ and $\mathcal{B}'$ such that $\mathcal{B} \subset \mathcal{B}'$ yet $\mathcal{B}' \sqsubseteq_{\mathcal{I}}^{\mathtt{dif}} \mathcal{B}$ but not $\mathcal{B} \sqsubseteq_{\mathcal{I}}^{\mathtt{dif}} \mathcal{B}'$. If $\mathcal{B} \subset \mathcal{B}'$ then $Cn(\mathcal{B}) \subset Cn(\mathcal{B}')$ by the monotonicity of Cn. By definitions 5.2.2 and 5.2.6 it is clear that it is necessary to construct $\mathcal{B}$ and $\mathcal{B}'$ such that $\mathcal{B}'$ results in the success or failure of more intentions than $\mathcal{B}$. Thus, let:

$$\mathcal{B} = \varnothing \qquad \mathcal{B}' = \{\beta\} \qquad \mathcal{I} = \{[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta')]\}$$

From the above constructions it can be seen that $\mathcal{B} \subset \mathcal{B}'$. Now it is necessary to show that these definitions lead to $\mathcal{B}' \sqsubseteq_{\mathcal{I}}^{\mathtt{dif}} \mathcal{B}$. From definition 5.2.2 it must be shown $\left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}'\right) \subseteq \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right)$ and $\left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}'\right) \subseteq \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}\right)$. Since $\mathcal{B} = \varnothing$ it is known that $\left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}\right) = \mathcal{B}^{\mathcal{I}}_{\mathtt{fail}}$ and $\left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right) = \mathcal{B}^{\mathcal{I}}_{\mathtt{suc}}$. As can be seen from the construction that $\left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}'\right) = \mathcal{B}^{\mathcal{I}}_{\mathtt{fail}}$ since $\mathcal{B}'$ contains no beliefs relevant to the failure conditions of the agent's intentions. However, since $\mathcal{B}'$ does contain elements relevant to the success conditions of the agent's intentions it follows that $\left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}'\right) \subset \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right)$. Thus, $\left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}'\right) \subseteq \left(\mathcal{I}\frac{\perp}{\mathtt{suc}}\mathcal{B}\right)$ and $\left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}'\right) \subseteq \left(\mathcal{I}\frac{\perp}{\mathtt{fail}}\mathcal{B}\right)$ are established as required. □

The intuition underlying the above proof is that $\sqsubseteq_{\mathcal{I}}^{\mathtt{dif}}$ prefers belief revisions that result in belief bases that are less relevant to the agent's current intentions. Each intention relevant belief added to the agent's beliefs will result in a change in the agent's intentions.

(5.2.6) Theorem (Hansson [51]):
$\sim_{\gamma^{\mathcal{I}}_{\mathtt{suc}}}$, $\sim_{\gamma^{\mathcal{I}}_{\mathtt{fail}}}$ *and* $\sim_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$ *satisfy the properties of success, inclusion, relevance, uniformity, failure, closure, relative closure, core-retainment, vacuity and extensionality by virtue of being transitive relational partial meet contractions.*

(5.2.7) Theorem (Hansson [52]):
$\pm_{\gamma^{\mathcal{I}}_{\mathtt{suc}}}$, $\pm_{\gamma^{\mathcal{I}}_{\mathtt{fail}}}$ *and* $\pm_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$ *satisfy the properties of consistency, inclusion, relevance, success, weak uniformity and pre-expansion by virtue of being external partial meet revisions.*

5.3 Prioritized Intentions

Although the above formalization captures the relationship between intentions and beliefs in a more flexible way, it fails to address the relationships between intentions themselves. Typically, an intention base is not simply just a set of intentions but has more structure. This additional structure will be captured through the use of a preference relation over intentions. Example 3.2.2 motivates the need for such additional structure.

Definitions 5.2.2, 5.2.3, 5.2.7 and 5.2.8 and the constraints imposed by definition 5.2.1 are retained but it is necessary to modify definitions 5.2.4 and 5.2.5 to integrate the preference relation over intentions.

(5.3.1) Definition (Preferential Intentional Agent):
A preferential intentional agent is a tuple: $\langle \mathcal{L}_{\mathcal{B}}, \mathcal{B}, Cn, \mathcal{L}_{\mathcal{I}}, \mathcal{I}, \trianglelefteq, \leq^{\mathcal{I}}_{\trianglelefteq} \rangle$ *where:*

$(\mathcal{L}_{\mathcal{B}},\ \mathcal{B},\ Cn,\ \mathcal{L}_{\mathcal{I}}$ *and* $\mathcal{I})$ *are as definition 2.1.2.*

$(\trianglelefteq)$ *is a transitive and complete preference order*[5] *over the agent's intentions.*

$(\leq^{\mathcal{I}}_{\trianglelefteq})$ *is a function that given an intention base and an ordering over intentions produces an ordering*[6] *over all subsets of the intention base.*

(5.3.2) Definition (Prioritized Success Maximizing Relation):
$\mathcal{B} \trianglelefteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ *if the set of intentions that* $\mathcal{B}'$ *renders successful is more preferred than the set of intentions rendered successful by* $\mathcal{B}$ *or, the set of intentions rendered successful by* $\mathcal{B}$ *and* $\mathcal{B}'$ *are equally preferred and the set of intentions deemed a failure by* $\mathcal{B}'$ *(disregarding those intentions that have been rendered successful) is less preferred*

[5] This preference order is reflexive, transitive and antisymmetric.
[6] The resulting generalized order is also reflexive, transitive and antisymmetric.

than the set of intentions deemed a failure by $\mathcal{B}$ (again disregarding those intentions that had been rendered successful)[7].

$$\mathcal{B} \trianglelefteq^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}' = \begin{cases} \top & \text{if } \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}\right) \lessdot^{\mathcal{I}}_{\trianglelefteq} \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}'\right), \\ \top & \text{if } \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}\right) \doteq^{\mathcal{I}}_{\trianglelefteq} \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}'\right) \wedge \left(\left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}'\right)_{\mathtt{fail}}\mathcal{B}'\right) \leq^{\mathcal{I}}_{\trianglelefteq} \left(\left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}\right)_{\mathtt{fail}}\mathcal{B}\right) \\ \bot & \text{otherwise} \end{cases}$$

5.3.3 Definition (Prioritized Failure Minimizing Relation):
$\mathcal{B} \trianglelefteq^{\mathtt{fail}}_{\mathcal{I}} \mathcal{B}'$ if the set of intentions that $\mathcal{B}'$ renders a failure is less preferred than the set of intentions rendered a failure by $\mathcal{B}$ or, the set of intentions rendered a failure by $\mathcal{B}$ and $\mathcal{B}'$ are equally preferred and the set of intentions deemed successful by $\mathcal{B}$ (disregarding those intentions that have been rendered a failure) is less preferable to the set of intentions deemed successful by $\mathcal{B}'$ (again disregarding those intentions that had been rendered a failure).

$$\mathcal{B} \trianglelefteq^{\mathtt{fail}}_{\mathcal{I}} \mathcal{B}' = \begin{cases} \top & \text{if } \left(\mathcal{I}^{\perp}_{\mathtt{fail}}\mathcal{B}'\right) \lessdot^{\mathcal{I}}_{\trianglelefteq} \left(\mathcal{I}^{\perp}_{\mathtt{fail}}\mathcal{B}\right), \\ \top & \text{if } \left(\mathcal{I}^{\perp}_{\mathtt{fail}}\mathcal{B}'\right) \doteq^{\mathcal{I}}_{\trianglelefteq} \left(\mathcal{I}^{\perp}_{\mathtt{fail}}\mathcal{B}\right) \wedge \left(\left(\mathcal{I}^{\perp}_{\mathtt{fail}}\mathcal{B}\right)^{\perp}_{\mathtt{suc}}\mathcal{B}\right) \leq^{\mathcal{I}}_{\trianglelefteq} \left(\left(\mathcal{I}^{\perp}_{\mathtt{fail}}\mathcal{B}'\right)^{\perp}_{\mathtt{suc}}\mathcal{B}'\right) \\ \bot & \text{otherwise} \end{cases}$$

The above definitions 5.3.2 and 5.3.3 assume that the importance of success and failure is equal. That is, for all intentions, an agent attempts to ensure their success and avoid their failure with equal ferocity. However, there may be situations in which this assumption is invalid. In particular, scenarios in which there are behaviours such that the cost of failure is negligible yet the reward is significant or the price of failure is dramatic yet the benefits of success slight. Removing this assumption will not be attempted in this thesis, however, it remains a topic of future work.

Notice, again, that if the intention base is empty then all belief revisions are equally preferred.

5.3.1 Observation:
If the set of intentions is empty then all belief bases are equally preferred by $\trianglelefteq^{\mathtt{suc}}_{\mathcal{I}}$ and $\trianglelefteq^{\mathtt{fail}}_{\mathcal{I}}$.

Proof. The proof for $\trianglelefteq^{\mathtt{suc}}_{\mathcal{I}}$ and $\trianglelefteq^{\mathtt{fail}}_{\mathcal{I}}$ are analogous to observation 5.2.1. □

[7] Note that $\doteq^{\mathcal{I}}_{\trianglelefteq}$ is shorthand for $\mathcal{I} \leq^{\mathcal{I}}_{\trianglelefteq} \mathcal{I}' \wedge \mathcal{I}' \leq^{\mathcal{I}}_{\trianglelefteq} \mathcal{I}$ for arbitrary $\mathcal{I}$ and $\mathcal{I}'$.

5.3.1 Theorem:

$\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ *is transitive.*

Proof. The proof is analogous to theorem 5.2.1 □

5.3.2 Theorem:

$\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ *is not weakly maximizing.*

Counter-Example. To see this, it is sufficient to construct a scenario where $\mathcal{B} \subset \mathcal{B}'$ but it is not the case that $\mathcal{B} \overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}'$ and $\mathcal{B}' \overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$. It is necessary to construct $\mathcal{B}$, $\mathcal{B}'$ and $\leqslant^{\mathcal{I}}_{\lhd}$ such that $\mathcal{B} \subset \mathcal{B}'$ and $\left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}'\right) \leqslant^{\mathcal{I}}_{\lhd} \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}\right)$ to show that $\mathcal{B}' \overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$. Let:

$$\mathcal{B} = \{\beta\} \qquad \mathcal{B}' = \{\beta, \beta'\}$$
$$\mathcal{I} = \{[\iota : \mathtt{suc}(\beta), \mathtt{fail}(\beta'')], [\iota' : \mathtt{suc}(\beta'), \mathtt{fail}(\beta''')]\}$$
$$\mathcal{I}' \leqslant^{\mathcal{I}}_{\lhd} \mathcal{I}'' = \begin{cases} \top & \text{if } \mathcal{I}' \supseteq \mathcal{I}'' \\ \top & \text{if } \exists \iota \in \mathcal{I}' \,.\, \iota \notin \mathcal{I}'' \wedge \forall \iota' \in \mathcal{I}'' \,.\, \iota' \notin \mathcal{I}'(\iota \lhd \iota') \\ \bot & \text{otherwise} \end{cases}$$

It can be seen that $\leqslant^{\mathcal{I}}_{\lhd}$ is a transitive and complete generalized preference relation over $2^{\mathcal{I}}$, that orders, firstly, according to intention set size and, secondly, according to the ordering of the least preferred intention that is not shared in both sets. It is clear from the construction of $\mathcal{B}$ and $\mathcal{B}'$ that $\mathcal{B} \subset \mathcal{B}'$. Furthermore it is known from the construction that $\mathcal{B}'$ renders at least one intention successful beyond the intentions that $\mathcal{B}$ renders successful. Since $\left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}\right) \subset \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}'\right)$, it follows that $\left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}'\right) \lessdot^{\mathcal{I}}_{\lhd} \left(\mathcal{I}^{\perp}_{\mathtt{suc}}\mathcal{B}\right)$ by the first clause of definition 5.3.2. Thus, $\mathcal{B}' \overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}} \mathcal{B}$ follows as required. □

The counter-example used to demonstrate that $\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ is not *weakly maximizing* (theorem 5.3.2) relies on a conflict between the preferences outlined in the definition of $\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ and the preference relation generated by $\leqslant^{\mathcal{I}}_{\lhd}$. This arises due to $\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ aiming towards the revision that maximizes the number of success conditions that follow from the resulting belief base and, therefore, a larger decrease in the number of intentions held and $\leqslant^{\mathcal{I}}_{\lhd}$ generating a preference relation which prefers larger intention bases and thus minimum change in the number of intentions held. Thus, the rationality behind such an example is questionable. Whether requiring consistency between $\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ and $\leqslant^{\mathcal{I}}_{\lhd}$ is sufficient to ensure the *weak maximizability* of $\overset{\lhd}{\sqsubseteq}^{\mathtt{suc}}_{\mathcal{I}}$ remains an open question.

5.3.3 Theorem:

$\trianglelefteq^{\mathtt{fail}}_{\mathcal{I}}$ *is transitive but not weakly maximizing.*

Proof. Proofs are analogous to theorems 5.3.1 and 5.3.2. □

5.3.4 Theorem (Hansson [51]):
$\sim_{\gamma^{\mathcal{I}}_{\mathtt{suc}}}$ *and* $\sim_{\gamma^{\mathcal{I}}_{\mathtt{fail}}}$ *satisfy the properties of success, inclusion, relevance, uniformity, failure, closure, relative closure, core-retainment, vacuity and extensionality by virtue of being a transitive weakly maximizingly relational partial meet contraction.*

5.3.5 Theorem (Hansson [52]):
$\pm_{\gamma^{\mathcal{I}}_{\mathtt{suc}}}$ *and* $\pm_{\gamma^{\mathcal{I}}_{\mathtt{fail}}}$ *satisfy the properties of consistency, inclusion, relevance, success, weak uniformity and pre-expansion by virtue of being an external partial meet revision.*

The proofs outlined in this chapter have demonstrated that it is possible to utilize an agent's intentions to guide the revision of it's beliefs. Through the particular constructions employed, an analysis of the rationality of such revisions utilising the rationality postulates of classical belief revision was conducted. Traditionally, however, the informational content of the belief bases in question have provided the guiding principle by which beliefs are revised. This informational content has been measured using multiple approaches. The most widely known technique is known as *epistemic entrenchment*. This approach aims to retain the beliefs with the maximal explanatory power for reasoning and planning problems. Two measures of such explanatory power have been advocated: *Gärdenfors's entrenchment*[40, chapter 1] and *Rott's entrenchment* [89] orderings. The difference lies in the reversibility of the revisions resultant to the two approaches. Another rational approach is that of *Levi's contraction operators* based on saturatable subsets. These contractions, like Rott's entrenchment, are not reversible by simply expanding the belief base by the formulæ by which it was contracted. Further approaches advocate the selection of the beliefs to be removed, not the beliefs to be retained. Such approaches include *safe contraction* and *kernel contraction*.

Of the aforementioned techniques for selecting rational revisions, the intention guided approach articulated above shares most in spirit with entrenchment orderings. Entrenchment aims to retain the beliefs most useful for problem solving, regardless of the particularities of the problems to be solved. The approach utilizing intentions and their behavioural knowledge, however, maximizes the retainment of the beliefs that are most valuable in the practical reasoning of the agent. This value is based on maximizing the retainment of the existing intentions, or maximizing their success while minimizing their failure. By specializing the revisions

according to the agent's current intentions, there is a risk that beliefs that are valuable for future decisions will be sacrificed. Whether this trade-off is, on-average, beneficial remains an open question. Additionally, approaches that balance the informational content and, thus, future value against their current value with respect to the agent's intentions deserves future attention and development.

5.4 Summary

For a belief revision operation to be rational it must satisfy the basic postulates as outlined in section 2.3.1. Adherence to additional postulates ensures rational belief change under broader circumstances, in particular, when revisions involve conjunctions or disjunctions. The above proofs demonstrate that, in circumstances in which there are multiple valid revisions of a belief base by a given formula, it is rational to select the intersection of those that maximize the beneficial outcome on the agent's intentions. It was demonstrated that revisions based on maintenance conditions offer additional rationality guarantees than those based on outcome conditions. This is likely due to the tighter relationship that holds between the current contents of the beliefs and the maintenance conditions of an agent's intentions. Maintenance conditions tie the stability of the agent's intentions to the stability of it's beliefs. As more beliefs are adopted the more likely it is that the maintenance conditions are satisfied and their corresponding intentions retained. Conversely, the more beliefs the agent is required to give up, the more likely it is that corresponding intentions will be required to be relinquished. This contrasts with the revisions aiming to maximise success or minimize failure. Belief bases with additional beliefs will not automatically be preferred by these revision operators. Any additional intention failure conditions satisfied by a given revision decreases that revision's desirability. Subsequently, more comprehensive belief bases are not automatically more preferential given the agent's intentions. Because of this, an agent may need to trade the rationality of belief revision against its affect on intention dynamics.

In America, any boy may become president and I suppose that's just one of the risks he takes.

Adlai Stevenson

6 Intention Revision

Hypothesis:
The explicit modelling of the dependencies between intentions that arise as a consequence of consistency maintenance during decision making facilitates both the quantification of commitment and mechanisms for intention reconsideration propagation.

This chapter will detail a theory of intention and commitment based on the intuition that the commitment an agent has towards an intention is proportional to the cost of dropping it. This cost is derived from the dependencies between intentions that arise as a consequence of decision making[20]. These dependencies arise as a product of intention consistency maintenance in light of intentions with expected outcomes and known side-effects (as discussed in chapter 3). The role of this theory in the revision of intentions will then be examined. In particular, the cost attributable to a given intention will be utilized to determine the appropriateness of its reconsideration.

An optimal intention reconsideration strategy grounds the agent's behaviour in its environment while minimizing the reasoning required to do so. If an agent reconsiders its intentions too infrequently then its behaviour can diverge from that required by the environment. If it reconsiders its intentions too often then its behaviour may become undirected, inefficient and opportunities for action may be missed. The structure of an agent's intentions and its commitments to achieve these greatly influence the nature of a rational policy an agent adopts. By quantifying commitment an agent is able to reason about its preferences to retain its intentions in circumstances in which it may revise them.

Firstly, the change in the commitment towards a given intention resulting from a number of identified intention change operations will be detailed. Then a method to approximate the initial costs of individual intentions will be presented. Following this, an approach to calculate the overall commitment an intention entails given its relationship to the other adopted intentions is developed. Based on this theory

of commitment, a number of approaches to evaluate the desirability of reconsidering an intention in light of changing intentions are then presented. Finally, the approach is specialized for integration into the semantics of an AgentSpeak (see section 2.1.3) agent.

6.1 Intention Dynamics

Because the commitment an agent places in a given intention changes with the evolution of its other intentions, it is necessary to identify the operations by which intentions change. Given a comprehensive set of such operations it is then necessary to characterize the change in the commitment placed in all intentions affected by, or related to intentions affected by, the operation in question. A number of such operations have been identified and each discussed, individually, below.

6.1.1 Intention Adoption

(6.1.1) Example:
Imagine an office assistant robot whose jobs include fetching coffee, delivering hard-copies of documents, vacuuming carpets and guiding office guests. One of these robots is unoccupied when it receives a request for coffee. Since the robot is otherwise unoccupied it chooses to deliver the coffee. The robot has two means of satisfying this opportunity: deliver the coffee itself, or cooperate with another robot to do so. The robot chooses to deliver the coffee itself resulting in a commitment towards this behaviour. A short time later, a request to relocate a document is made of the robot. The robot has two means of satisfying this request: deliver the document itself, or, request the services of another agent. Because the robot is already committed to delivering the coffee and the document requires delivery to a distant area of the building, it cannot compatibly deliver both the coffee and the document. Therefore, it negotiates with another agent to jointly handle the task instead. Because the robot's commitment towards delivering coffee influenced its decision to request the services of another robot to handle the document delivery request, there is a dependency between the two commitments.

Every time an intention plays an active role in a decision an agent should increase its commitment towards this intention. Should the agent retract this intention at some future point, doing so would contribute to the reasons why the agent should reconsider decisions in which it affected the outcome. For an existing intention to play an active role in a particular decision it must invalidate one

or more of the options over which the agent is contemplating. However, for each option that is invalidated by the agent's current intentions, multiple intentions may invalidate it. Thus, there is a potentially many-to-many relationship between options and the intentions that invalidate them. The agent should, therefore, distribute the cost of such a decision to each intention in a way that is proportional to its contribution.

The process of adopting an intention to satisfy a goal can be divided into three stages: (1.) Option Generation; (2.) Option Filtering; and (3.) Option Selection. Firstly, an agent generates the set of options for achieving its current goal. This may involve extracting relevant and applicable plans from a fixed plan library or planning from first principles. For the present purposes, it is safe to assume that this is a fixed but arbitrary cost. Regardless of the source of the alternatives, it is necessary to filter the generated options to the subset that is compatible with the agent's existing intentions. This is because it is demonstrably irrational to adopt intentions that are not internally consistent. The filtering process, in the worst case, requires comparing each option against each intention. Again, this is a fixed cost that is, in the worst case, proportional to the number of options and intentions involved[1]. Finally, one of the options consistent with the agent's existing intentions must be selected and adopted as the intention by which the goal is to be achieved. To make such a selection, the set of options consistent with the agent's intentions need to be ordered according to some suitability metric and the most suitable option subsequently selected. The nature of the metric is unimportant. The crucial aspect is that the options consistent with the agent's intentions must be ordered. Such a process will be proportional to the number of options to order. Thus, the more options that are consistent with the agent's intentions that the selection process must order, the more costly the selection process is. Conversely, the higher the proportion of options that are incompatible with the agent's intentions, the easier the selection will be.

Figure 6.1 depicts a scenario in which the cost of adopting a given intention is distributed amongst the involved intentions. The black lines indicate conflicts between intentions and options. ω_4 is the selected option. Two out of the four options are invalidated by the agent's intentions (namely ω_1, ω_2). Thus, $1/2$ of the cost of adopting ω_4 is assignable to these intentions and $1/2$ to the ω_4 itself. Had there been no conflicts between the options available and the agent's intentions, then the agent's intentions would have been irrelevant in the decision to adopt ω_4 and consequently the entire cost of adopting ω_4 is attributable to ω_4. In this example,

[1]Though this fixed cost differs for each adoption.

however, there are three conflicts: between ω_1 and ι_2; ω_2 and ι_2; and, between ω_2 and ι_3. Thus, each conflict is weighted $1/3$ of the $1/2$ assignable to the intentions as a whole. Since ι_2 is involved in two of these conflicts, its cumulative weight is $2 \times 1/6$, i.e. $1/3$. Because ι_3 is involved in only one conflict, only $1/6$ of the total cost of adopting ω_4 is attributable to it.

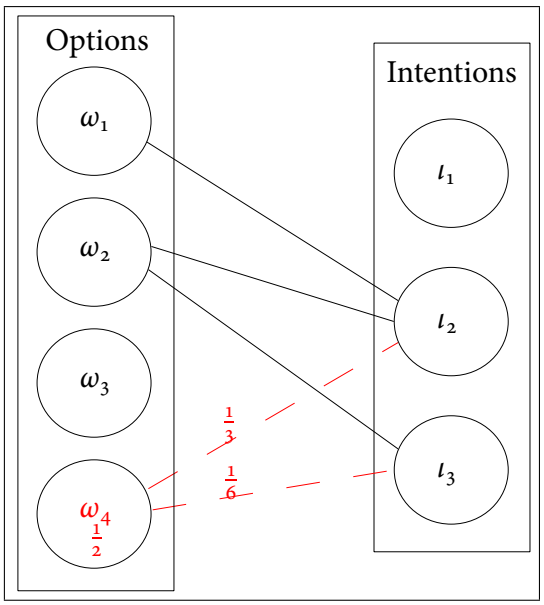


Figure 6.1: Intention-Option Conflicts

Imagine an agent is adopting an intention to satisfy some goal. The agent has a set of options for satisfying this goal and the selected option must be compatible with it's intentions. The agent must check each option for consistency against all currently held intentions. Assume that a subset of these options is found to be incompatible with at least one intention using its conflict recognition function ($\Xi^{\mathcal{O}}_{\mathcal{I}}$).

6.1.1 Definition (Intention Inconsistent Options):
The subset of options that are inconsistent with the agent's currently held intentions:

$$\mathcal{O}_t \not\sim \mathcal{I}_t = \left\{ \omega \in \mathcal{O}_t \;\middle|\; \exists \iota \,.\, \iota \in \mathcal{I}_t \wedge \Xi^{\mathcal{O}}_{\mathcal{I}}(\omega, \iota) = \top \right\}$$

where:

$\mathcal{I}$ *: is the agent's currently held intentions.*

$\mathcal{O}$ *: is the set of options under consideration to satisfy a particular goal.*

$\boxed{t}$ *: the time at which the options were generated.*

$\boxed{\Xi_{\mathcal{I}}^{\mathcal{O}}(\omega, \iota)}$ *: is a function that maps an option and an intention to true (⊤) if they are not simultaneously adoptable, and false (⊥) otherwise.*

6.1.2 Definition (Option Inconsistent Intentions):
The subset of intentions that are in conflict with one or more options under consideration:

$$\mathcal{I}_t \wr \mathcal{O}_t = \left\{ \iota \in \mathcal{I}_t \;\middle|\; \exists \omega \,.\, \omega \in \mathcal{O}_t \wedge \Xi_{\mathcal{I}}^{\mathcal{O}}(\omega, \iota) = \top \right\}$$

where:

$\boxed{\mathcal{I}}$ *: is the agent's currently held intentions.*

$\boxed{\mathcal{O}}$ *: is the set of options under consideration to satisfy a particular goal.*

$\boxed{t}$ *: the time at which the options were generated.*

$\boxed{\Xi_{\mathcal{I}}^{\mathcal{O}}(\omega, \iota)}$ *: is a function that maps an option and an intention to true (⊤) if they are not simultaneously adoptable, and false (⊥) otherwise.*

Assume that an agent makes a selection from the compatible options available to it. The agent has a proportion of the cost of intending the selection to distribute among the intentions involved in the decision. This proportion is the number of inconsistent options given the agent's intentions over the total number of options under consideration. Thus, the initial commitment towards intending the selected option is:

6.1.3 Definition (Base Commitment):
The base *commitment placed in a newly adopted intention σ is the cost (Ω) of adopting σ provided the set of intention inconsistent options is empty. Otherwise, it is 1 - the proportion of options invalidated by the agent's intentions ($|\mathcal{O}_t \wr \mathcal{I}_t| / |\mathcal{O}|$) applied to the cost of adopting it (Ω).*

$$\Pi(\sigma) = \begin{cases} \Omega(\sigma) & \text{if } \mathcal{O}_t \wr \mathcal{I}_t = \varnothing \\ \left(1 - \frac{|\mathcal{O}_t \wr \mathcal{I}_t|}{|\mathcal{O}|}\right) \times \Omega(\sigma) & \text{otherwise} \end{cases}$$

where:

σ : *is the selected option for which the base commitment is being calculated.*

$\Pi(\sigma)$: *is the initial commitment associated with the selected option* σ.

$\mathcal{O}_t, \mathcal{I}_t$: *are the alternative options that were discarded in the decision to adopt* σ *due to conflict with the agent's intentions (see definition 6.1.1).*

$\mathcal{O}$: *is the set of options generated to satisfy the new goal.*

$\Omega(\sigma)$: *is the initial cost of adopting* σ. *An example mechanism to calculate this and further discussion are provided in section 6.2.1.*

In other words, the *base* commitment placed in an intention ι is the cost of adopting the intention (Ω) if the agent's prior intentions played no role in its adoption. If the agent's existing intentions played a role in the adoption of ι, then the *base* commitment placed in ι is the cost of its adoption not attributable to the agent's prior intentions.

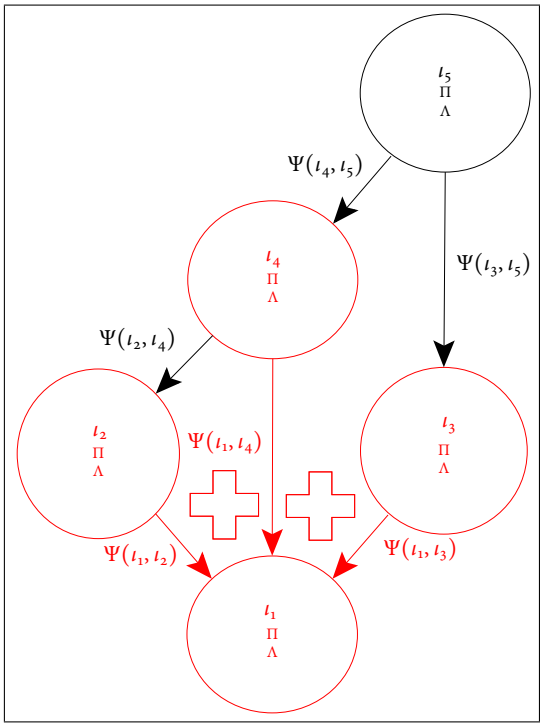


Figure 6.2: Commitment Calculation

Figure 6.2 highlights the role of the intention dependencies in the calculation of the commitment an agent holds towards a given intention. Each intention has a base (Π) and accumulated (Λ) component to this commitment. To calculate the overall commitment placed in intention ι_1, the base and accumulated commitment towards it are utilized in addition to the dependencies between it and ι_2, ι_3 and ι_4. Similarly, ι_2 is depended upon by ι_4 and the commitment held towards ι_4 contributes to the overall commitment ι_2 entails. Thus, ι_4 contributes multiple times towards the commitment held in ι_1. Firstly, it contributes directly through its dependence upon ι_1. Secondly, through the dependencies between ι_4 and ι_2 and ι_2 and ι_1.
The weight of the dependency between each intention $(\mathcal{I}_t \wr \mathcal{O}_t)$ involved in the decision to intend σ is:

6.1.4 Definition (Intention Dependency Weights):

$$\Psi(\sigma, \iota) = \frac{|\mathcal{O}_t \wr \{\iota\}| \times |\mathcal{O}_t \wr \mathcal{I}_t|}{|\mathcal{O}_t| \times \sum_{\omega \in \mathcal{O}_t \wr \mathcal{I}_t} |\mathcal{I}_t \wr \{\omega\}|}$$

where:

σ *: is the option selected from* $\mathcal{O}_t$ *to satisfy the goal in question.*

$\Psi(\sigma, \iota)$ *: is the weight assigned to the dependency between the option selected* (σ) *and the intention* ι.

$|\mathcal{O}_t \wr \{\iota\}|$ *: is the number of options that conflict with intention* ι.

$|\mathcal{O}_t \wr \mathcal{I}_t|$ *: is the number of options that have been invalidated by the agent's intentions (see definition 6.1.1).*

$|\mathcal{O}_t|$ *: is the number of alternative options among which the agent is contemplating.*

$|\mathcal{I}_t \wr \{\omega\}|$ *: is the number of intentions that are in conflict with option* ω.

t *: is the time at which the decision to adopt* σ *was made.*

It is necessary to utilize $\sum_{\omega \in \mathcal{O}_t \wr \mathcal{I}_t} |\mathcal{I}_t \wr \{\omega\}|$ *and not* $\mathcal{I}_t \wr \mathcal{O}_t$ *as* $\mathcal{I}_t \wr \mathcal{O}_t$ *discards duplicate uses of a given intention in the invalidation of one or more options.*

The weight of the dependency between the selected option and each intention is the proportion of the cost inherent in the adoption of the selection attributable to the given intention. This is equivalent to the contribution of the intention in question in making the selection towards the selected option over the agent's other options. Each intention may invalidate one or more options under consideration. Each option may be invalidated my one or more intentions. Each option is considered equally applicable prior to any being invalidated. The proportion of the commitment attributable to all the participating intentions is the number of options invalidated over the total number of intentions. For the given intention, its role is the proportion of the number of options it invalidates over the total number of options invalidated (including duplicates) by the agent's intentions. By multiplying these proportions, the contribution of the intention towards the resulting selection, given the total number of options invalidated, the number of options the given intention invalidates and the number of other intentions which may have also contributed to the invalidation of common options, is derived.

The base commitment, as defined above, is justified by observing that this is the commitment remaining after assigning the appropriate portions to the intentions that played an active role in the decision process. After discarding those options that were in conflict with the agent's existing intentions, for this option to be selected, it must be based on its merits relative to the other remaining valid options. Therefore, the fewer options that are invalidated by the context in which the decision is made, the more commitment is made towards the outcome of the decision making process. Conversely, the more options invalidated by the context the less commitment is attributable to the actual result of the decision process. This means that in situations in which there are few consistent options, the initial commitment towards the selected option will be slight.

There is a competing intuition that the above formalization does not capture. When there are few options between which to deliberate, the commitment should be high. This is due to the assumed stability of the agent's intentions against which the decision is to be made. Given a largely similar set of intentions and few options between which to deliberate, the likelihood is that the same outcome of deliberation will result. Therefore, the commitment should be high so as to avoid the need for such repetitive and unproductive reasoning. However, this analysis fails to consider the cost of deliberation. Making a selection amongst few options is less costly than making a selection amongst many. The commitment attributed to an intention aims to be proportional to the cost in making the selection. Thus, the commitment should be low. Furthermore, since, as argued, such repetitious deliberation will result in the same conclusions, the resultant repeatedly adopted intention will ac-

cumulate commitment as a result (see definition 6.1.5). This obviates the need for a high level of initial commitment. Thus, the approach advocated promotes a preference towards repeated simple decisions over single simple decisions at the cost of repeated difficult decisions.

In order to apply the above formalization to the office robot example, it is necessary to define the agent's conflict function. The following matrix does so.[2]

$\Xi^{\mathcal{O}}_{\mathcal{I}}$	'outsource document'	'deliver document'	'deliver coffee'	'vacuum'	'guide guest'
'guide guest'	$\bot$	$\top$	$\top$	$\bot$	$\bot$
'vacuum'	$\bot$	$\bot$	$\bot$	$\bot$	
'deliver coffee'	$\bot$	$\top$	$\bot$		
'deliver document'	$\top$	$\bot$			
'outsource document'	$\bot$				

When the office assistant robot is faced with the initial decision (at time t_1) as to how to address the need to deliver coffee, it generates the following options:

$$\mathcal{O}_{t_1} = \{\text{'deliver coffee'}, \text{'outsource coffee'}\}$$

Because the office assistant robot is otherwise uncommitted, its initial intentions are empty:

$$\mathcal{I}_{t_1} = \varnothing$$

Subsequently, the set of options that are in conflict with the agent's intentions is empty:

$$\mathcal{O}_t \wr \mathcal{I}_t = \varnothing$$

Additionally, because it was previously unoccupied, the agent chooses to deliver the coffee itself:

$$\sigma = \text{'deliver coffee'}$$

Finally, due to the agent's lack of prior intentions, the agent was not able to discard any of its alternatives for satisfying its goal of delivering coffee. Thus, the agent had to make a selection between all the alternatives available to it. Because of this, the selection was made based solely on the quality of the different alternatives. The subsequent selection was simply the best alternative available and all of the cost of making the selection is attributable to it.

$$\Pi(\text{'deliver coffee'}) = \Omega(\text{'deliver coffee'})$$

[2] For the purposes of this running example, it is assumed that the only documents in question are those explicitly mentioned and going to different physically distant locations. A more realistic approach would parameterize the intentions with the document instances to which they refer and take this parameterization into account when determining conflict.

When selecting a behaviour to handle the request for document delivery at time t_2, the initial set of options are to deliver or outsource the document:

$$\mathcal{O}_{t_2} = \{\text{'deliver document'}, \text{'outsource document'}\}$$

However, at this point, the agent already intends to deliver coffee:

$$\mathcal{I}_{t_2} = \{\text{'deliver coffee'}\}$$

Because of this and the agent's conflict function, the option to deliver the document itself is detected as incompatible with the agent's existing intention to deliver the coffee:

$$\mathcal{O}_{t_2} \wr \mathcal{I}_{t_2} = \{\text{'deliver document'}\},$$
$$\mathcal{I}_{t_2} \wr \{\text{'deliver document'}\} = \{\text{'deliver coffee'}\},$$
$$\mathcal{O}_{t_2} \wr \{\text{'deliver coffee'}\} = \{\text{'deliver document'}\},$$

This leaves outsourcing the document as the only compatible option available for the agent to select to handle the goal of delivering the document:

$$\sigma = \text{'outsource document'}$$

Because the agent's intention towards delivering coffee simplified the decision towards how to handle the need to deliver the document by half, half of the cost of this decision is attributable to it. This is achieved by creating a dependency between the agent's intention to deliver coffee and to outsource the document which has a weight of half.

$$\Psi(\text{'deliver coffee'}, \text{'outsource document'}) = 1/2$$
$$\Psi(\text{'outsource document'}, \text{'deliver coffee'}) = 0$$

As a consequence of attributing half of the cost of adopting the intention to outsource the document to the agent's existing intention, only half remains to attribute to the intention to outsource the document directly:

$$\Pi(\text{'outsource document'}) = \frac{\Omega(\text{'outsource document'})}{2}$$

6.1.2 Intention Revision

6.1.2 Example:
Imagine the robot as per example 6.1.1 has been acting in such a way as to deliver the coffee requested of it. Because it needs to collect the document and deliver it to the robot with whom it is cooperating, it has decided and committed to getting the coffee from the coffee machine between its current location and the meeting place instead of the closer gourmet coffee dispenser in the cafeteria. After making progress towards the coffee machine the other agent informs the robot that it will not be able to meet at the agreed time to accept the document due to unexpected circumstances. Consequently the agent reconsiders its intention to get the coffee from the coffee machine and instead sources the coffee from the cafeteria, committing to meet the robot at a later time. Because this change in behaviour has wasted the agent's time and energy it now commits itself more strongly to its new intention to source the coffee from the cafeteria in order to minimize the chance of further reconsideration and wasted efforts.

Should an agent revise an intention, the commitment the agent adopts towards the replacement should be greater than the commitment it had towards the original. This is because the agent has made a choice to drop the existing intention and accepts all the costs related in doing so (as the cost should be a consideration in the decision to reconsider). This is important in maintaining the stability of the new intention. If it were not to inherit the commitment from the intention it replaced, then it is likely that the new intention would quickly become a candidate for further revision. As a newly adopted intention, the replacement intention would have no dependants. As will be seen in section 6.3 intentions without dependants are considered significantly easier to revise than those with. This is because newly adopted intentions have not had any opportunity to play any role in the agent's decision making. The imposition of such a constraint also has a stabilizing effect on the agent's intentions. An agent is less likely to revise an intention knowing that the commitment it must hold towards the replacement intention will be significant, particularly given the replacement will only accrue additional commitment as it is used in decision making over time.

6.1.5 Definition (Accumulated Commitment):
The commitment towards an intention adopted consequent to the revision of another accumulates the commitment held in the previous intention.

$$\Lambda(\iota') = \begin{cases} \Theta(\iota) & \text{if } \iota \text{ was revised to } \iota' \\ 0 & \text{otherwise} \end{cases}$$

where:

ι' *: is the newly adopted intention.*

ι *: is the original intention that was reconsidered and replaced with* ι'*.*

$\Theta(\iota)$ *: is the total commitment the agent directed towards* ι *(see definition 6.2.4).*

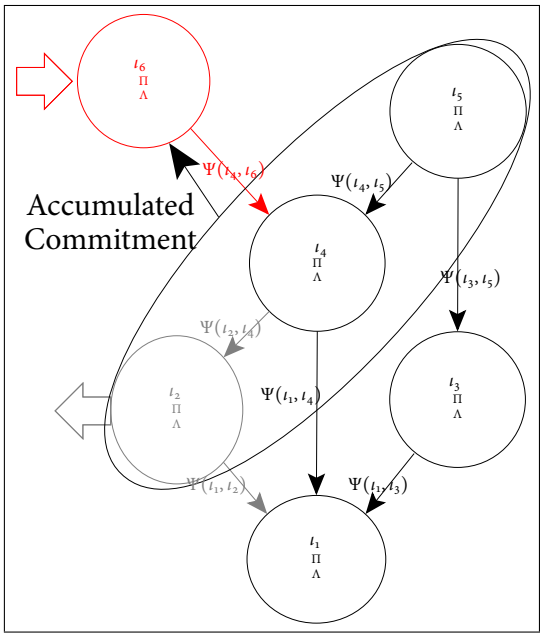


Figure 6.3: Commitment Accumulation subsequent to Intention Revision

Figure 6.3 exemplifies the case where one intention (ι_2) is revised and subsequently replaced with another (ι_6). As can be seen, ι_5 is dependent on ι_4 and ι_4 is dependent on the replaced intention (ι_2). Thus, the commitment the agent holds towards ι_2, including that contributed by the aforementioned dependencies, is transferred to the new intention (ι_6).

Returning to the example above, the agent's intentions during the decision to source the coffee from the coffee machine (at time t_3) are:

$$\mathcal{I}_{t_3} = \{\text{'deliver coffee'}, \text{'outsource document'}\}$$

The set of options for sourcing the coffee are:

$$\mathcal{O}_{t_3} = \{\text{'coffee from machine'}, \text{'coffee from cafeteria'}\}$$

The subset of these options that are incompatible with the agent's existing intentions are:

$$\mathcal{O}_{t_3} \wr \mathcal{I}_{t_3} = \{\text{'coffee from cafeteria'}\}$$

This is due to the incompatibility between the intention to 'outsource document' and 'coffee from cafeteria':

$$\mathcal{I} \wr_{t_3} \{\text{'coffee from cafeteria}\} = \{\text{'outsource document'}\}$$
$$\mathcal{O} \wr_{t_3} \{\text{'outsource document'}\} = \{\text{'coffee from cafeteria'}\}$$

As before, compatibility with agent's intentions have constrained this decision to triviality. There is only one remaining option which the agent may adopt to satisfy its goal of sourcing the coffee:

$$\sigma = \text{'coffee from machine'}$$

Subsequently, the cost of this decision is divided equally among the decision relevant intentions and the resulting selection:

$$\Psi(\text{'outsource document'}, \text{'coffee from machine'}) = 1/2$$
$$\Psi(\text{'coffee from machine'}, \text{'outsource document'}) = 0$$
$$\Pi(\text{'coffee from machine'}) = \frac{\Omega(\text{'coffee from machine'})}{2}$$

Thus, at the point prior to revising its intentions to accommodate the news that the agent to which it outsourced the document delivery to will not meet at the agreed

upon time (time t_4), the agent's intentions are:

$$\mathcal{I}_{t_4} = \{\text{'outsource document'}, \text{'deliver coffee'}, \text{'coffee from machine'}\},$$
$$\Pi(\text{'coffee from machine'}) = \frac{\Omega(\text{'coffee from machine'})}{2}$$
$$\Pi(\text{'outsource document'}) = \frac{\Omega(\text{'outsource document'})}{2}$$
$$\Pi(\text{'deliver coffee'}) = \Omega(\text{'deliver coffee'})$$
$$\Psi(\text{'deliver coffee'}, \text{'outsource document'}) = 1/2$$
$$\Psi(\text{'deliver coffee'}, \text{'coffee from machine'}) = 0$$
$$\Psi(\text{'outsource document'}, \text{'deliver coffee'}) = 0$$
$$\Psi(\text{'outsource document'}, \text{'coffee from machine'}) = 1/2$$
$$\Psi(\text{'coffee from machine'}, \text{'deliver coffee'}) = 0$$
$$\Psi(\text{'coffee from machine'}, \text{'outsource document'}) = 0$$
$$\Lambda(\text{'deliver coffee'}) = 0$$
$$\Lambda(\text{'outsource document'}) = 0$$
$$\Lambda(\text{'coffee from machine'}) = 0$$

The agent's intentions subsequent (at time t_5) to the revision of the intention towards 'coffee from the machine' to a new intention to 'coffee from cafeteria' can be seen below. Given the change in circumstances, it is assumed that the agent now considers sourcing the coffee from the machine incompatible with outsourcing the document and sourcing the coffee from the cafeteria as compatible. As a result of dropping the intention towards sourcing the coffee from the machine, any depen-

dencies it was involved in are reset to zero.

$$\mathcal{I}_{t_5} = \{\text{'outsource document'}, \text{'deliver coffee'}, \text{'coffee from cafeteria'}\},$$
$$\mathcal{O}_{t_5} = \{\text{'coffee from cafeteria'}, \text{'coffee from machine'}\}$$
$$\mathcal{O}_{t_5} \wr \mathcal{I}_{t_5} = \{\text{'coffee from machine'}\}$$
$$\mathcal{I}_{t_5} \wr \mathcal{O}_{t_5} = \{\text{'outsource document'}\}$$
$$\Pi(\text{'coffee from cafeteria'}) = \frac{\Omega(\text{'coffee from cafeteria'})}{2}$$
$$\Pi(\text{'outsource document'}) = \frac{\Omega(\text{'outsource document'})}{2}$$
$$\Pi(\text{'deliver coffee'}) = \Omega(\text{'deliver coffee'})$$
$$\Psi(\text{'deliver coffee'}, \text{'outsource document'}) = 1/2$$
$$\Psi(\text{'deliver coffee'}, \text{'coffee from cafeteria'}) = 0$$
$$\Psi(\text{'deliver coffee'}, \text{'coffee from machine'}) = 0$$
$$\Psi(\text{'outsource document'}, \text{'coffee from cafeteria'}) = 1/2$$
$$\Psi(\text{'outsource document'}, \text{'deliver coffee'}) = 0$$
$$\Psi(\text{'outsource document'}, \text{'coffee from machine'}) = 0$$
$$\Lambda(\text{'deliver coffee'}) = 0$$
$$\Lambda(\text{'outsource document'}) = 0$$
$$\Lambda(\text{'coffee from cafeteria'}) = \frac{\Omega(\text{'coffee from machine'})}{2}$$

6.1.3 Intention Completion

While it is clear that the completion and subsequent removal of an individual intention is an intention changing operation, it is one that plays no role in the dynamics of commitment when the only structure imposed on intentions is that of dependencies arising from consistency maintenance. While the commitment held in a dependent intention is increased as a result of the adoption of dependent intentions, this additional commitment is abandoned with the completion of the dependent intention. This is due to the subsequent decrease in cost of abandoning the previously depended upon intention. The previously depended upon intention no longer needs to be maintained to prevent reconsideration of the now abandoned intention, and consequently the commitment it requires is lessened. No additional

explicit commitment modification is required for this decline in commitment towards previously depended upon intentions, it follows automatically from the definitions for the attribution of commitment (section 6.2.2).

6.2 Calculating Commitment

Given the mechanisms above to handle the effect of the identified operations of intention change on the agent's commitments, it is now necessary to complete the picture by describing how these can be amalgamated to quantify the commitment an agent places in a given intention.

6.2.1 Intention Cost

The theory presented thus far is agnostic to the methodology for calculating the cost of adopting a given intention. To demonstrate the feasibility of such a cost function an example approach is presented below.

When adopting an intention, an agent must make a selection between competing options. In so doing, an agent consciously discourages the pursuit of any desires that conflict with the newly intended option. To pursue such desires would require a reconsideration of the agent's newly adopted intention to accommodate this. It also strengthens any conflicts with desires that are shared with other pre-existing intentions. Intuitively, the cost of adopting a given intention is a measure of the desires that the agent is unable to pursue due to intending it. Multiple intentions may conflict with a given desire and an intention may conflict with multiple desires. Therefore, it is necessary to distribute the cost of each unfulfillable desire among the intentions that prevent the adoption of intentions towards their realization, including the newly adopted intention.

In order to formalize this it is assumed that there is a set of the agent's desires. These need not be consistent. For a given intention, there is a subset of the desires with which it is inconsistent:

6.2.1 Definition (Intention Inconsistent Desires):
The set of desires inconsistent with a given intention:

$$\mathcal{D}_t \wr \mathcal{I}_t = \left\{ \delta \in \mathcal{D} \;\middle|\; \exists \iota \,.\, \iota \in \mathcal{I} \wedge \Xi^{\mathcal{I}}_{\mathcal{D}}(\iota, \delta) = \top \right\}$$

where:

$(\mathcal{D})$ *: are the agent's desires.*

$(\mathcal{I})$ *: is the intentions to calculate the cost of adopting.*

$(\Xi^{\mathcal{I}}_{\mathcal{D}}(\iota, \delta))$ *: is a function that maps an intention and a desire to true ($\top$) if they are not simultaneously adoptable, and false ($\bot$) otherwise.*

6.2.2 Definition (Desire Inconsistent Intentions):
The set of intentions inconsistent with a given desire:

$$\mathcal{I}_t \wr \mathcal{D}_t = \left\{ \iota \in \mathcal{I} \;\middle|\; \exists \delta \,.\, \delta \in \mathcal{D} \wedge \Xi^{\mathcal{I}}_{\mathcal{D}}(\iota, \delta) = \top \right\}$$

where:

$(\mathcal{I})$ *: the agent's intentions.*

$(\mathcal{D})$ *: the desires for which the inconsistent intentions are to be calculated.*

$(\Xi^{\mathcal{I}}_{\mathcal{D}}(\iota, \delta))$ *: is a function that maps an intention and a desire to true ($\top$) if they are not simultaneously adoptable, and false ($\bot$) otherwise.*

Given this set of conflicting desires and the agent's other intentions, the cost of an intention ι is:

6.2.3 Definition (Intention Cost):
The cost of adopting intention ι:

$$\Omega(\iota) = \begin{cases} 0 & \text{if } \mathcal{D}_t \wr \{\iota\} = \varnothing \\ \sum_{\delta \in \mathcal{D}_t \wr \{\iota\}} \frac{1}{|\mathcal{I}_t \wr \{\delta\}|} & \text{otherwise} \end{cases}$$

where:

$(\Omega(\iota))$ *: is the cost associated with intention ι.*

$\mathcal{I}_t \wr \{\delta\}$ *: is the set of intentions in conflict with a given desire* δ *(see definition 6.2.2).*

Each desire is considered equally valid and important to pursue. If the intention is consistent with all of the agent's desires then there is no cost involved in adopting it. Otherwise, the cost is proportional to the number of desires that it renders unrealisable, given that other intentions may do so also.

To apply the above to the example office robot both prior to and after its adoption of the commitment to outsource the delivery of the document, assume the agent's set of desires are:

$$\mathcal{D}_{t_1} = \{\text{'guide guest'}, \text{'vacuum'}, \text{'deliver coffee'}, \text{'deliver document'}\}$$

and its desire conflict function is defined by the following matrix, which mirrors its intention conflict function:

$\Xi^{\mathcal{I}}_{\mathcal{D}}$	'outsource document'	'deliver document'	'deliver coffee'	'vacuum'	'guide guest'
'guide guest'	$\bot$	$\top$	$\top$	$\bot$	$\bot$
'vacuum'	$\bot$	$\bot$	$\bot$	$\bot$	
'deliver coffee'	$\bot$	$\top$	$\bot$		
'deliver document'	$\top$	$\bot$			
'outsource document'	$\bot$				

This results in the following prior (at time t_2) to adopting a commitment to outsource the document delivery:

$\mathcal{I}_{t_2} = \{\text{'deliver coffee'}\}$
$\mathcal{D}_{t_2} = \{\text{'guide guest'}, \text{'vacuum'}, \text{'deliver coffee'}, \text{'deliver document'}\}$
$\mathcal{D}_{t_2} \wr \text{'deliver coffee'} = \{\text{'guide guest'}, \text{'deliver document'}\}$
$\mathcal{I}_{t_2} \wr \text{'guide guest'} = \{\text{'deliver coffee'}\}$
$\mathcal{I}_{t_2} \wr \text{'deliver document'} = \{\text{'deliver coffee'}\}$

and thus:

$$\begin{aligned}
\Omega(\text{'deliver coffee'}) &= \sum_{\delta \in \mathcal{D}_{t_2} \rangle \{\text{'deliver coffee'}\}} \frac{1}{|\mathcal{I}_{t_2} \rangle \{\delta\}|} \\
&= \sum_{\delta \in \{\text{'guide guest'}, \text{'deliver document'}\}} \frac{1}{|\mathcal{I}_{t_2} \rangle \{\delta\}|} \\
&= \frac{1}{|\mathcal{I}_{t2} \rangle \{\text{'guide guest'}\}|} + \frac{1}{|\mathcal{I}_{t_2} \rangle \{\text{'deliver document'}\}|} \\
&= \frac{1}{|\{\text{'deliver coffee'}\}|} + \frac{1}{|\{\text{'deliver coffee'}\}|} \\
&= \frac{1}{1} + \frac{1}{1} \\
&= 2
\end{aligned}$$

It leads also to the following subsequent to the commitment to outsource the document delivery at time t_3:

$$\begin{aligned}
&\mathcal{I}_{t_3} = \{\text{'deliver coffee'}, \text{'outsource document'}\} \\
&\mathcal{D}_{t_3} = \{\text{'guide guest'}, \text{'vacuum'}, \text{'deliver coffee'}, \text{'deliver document'}\} \\
&\mathcal{D}_{t_3} \rangle \text{'deliver coffee'} = \{\text{'guide guest'}, \text{'deliver document'}\} \\
&\mathcal{I}_{t_3} \rangle \text{'guide guest'} = \{\text{'deliver coffee'}\} \\
&\mathcal{I}_{t_3} \rangle \text{'deliver document'} = \{\text{'deliver coffee'}, \text{'outsource document'}\}
\end{aligned}$$

and thus:

$$\begin{aligned}
\Omega(\text{'deliver coffee'}) &= \sum_{\delta \in \mathcal{D}_{t_3} \rangle \{\text{'deliver coffee'}\}} \frac{1}{|\mathcal{I}_{t_3} \rangle \{\delta\}|} \\
&= \sum_{\delta \in \{\text{'guide guest'}, \text{'deliver document'}\}} \frac{1}{|\mathcal{I}_{t_3} \rangle \{\delta\}|} \\
&= \frac{1}{|\mathcal{I}_{t3} \rangle \{\text{'guide guest'}\}|} + \frac{1}{|\mathcal{I}_{t_2} \rangle \{\text{'deliver document'}\}|} \\
&= \frac{1}{|\{\text{'deliver coffee'}\}|} + \frac{1}{\left|\left\{ \begin{array}{c} \text{'deliver coffee'}, \\ \text{'outsource document'} \end{array} \right\}\right|} \\
&= \frac{1}{1} + \frac{1}{2} \\
&= 1\,1/2
\end{aligned}$$

and:

$$\mathcal{D}_{t_3}\wr\text{`outsource document'} = \{\text{`deliver document'}\}$$
$$\mathcal{I}_{t_3}\wr\text{`deliver document'} = \{\text{`deliver coffee', `outsource document'}\}$$

and thus:

$$\begin{aligned}
\Omega(\text{`outsource document'}) &= \sum_{\delta \in \mathcal{D}_{t_3}\wr\{\text{`outsource document'}\}} \frac{1}{|\mathcal{I}_{t_3}\wr\{\delta\}|} \\
&= \sum_{\delta \in \{\text{`deliver document'}\}} \frac{1}{|\mathcal{I}_{t_3}\wr\{\delta\}|} \\
&= \frac{1}{|\mathcal{I}_{t_3}\wr\{\text{`deliver document'}\}|} \\
&= \frac{1}{|\{\text{`deliver coffee', `outsource document'}\}|} \\
&= 1/2
\end{aligned}$$

6.2.2 Relative Commitment

Intentions are related by their use in decision making. When an agent makes a decision towards the adoption of a new intention, the outcome of such a decision is dependent on the agent's existing intentions. This dependency arises due to the consistency required of intentions. An agent with inconsistent intentions is likely to act in a way that prevents the achievement of its ends. The dependencies between intentions are not all equivalent. Some intentions may play a stronger role in invalidating the options available in a given decision than others. This strength is reflected in the weights associated with each dependency. The commitment an agent places in a given intention is heavily reliant on the weight of its dependencies and the commitment the agent places in the dependent intentions. This is formalized as follows:

6.2.4 Definition (Commitment):
The commitment an agent holds towards a given intention is defined according to:

$$\Theta(\iota) = \left(\sum_{\iota' \in \mathcal{I}} \Theta(\iota') \times \Psi(\iota, \iota') \right) + \Lambda(\iota) + \Pi(\iota)$$

where:

ι *: is the intention in question.*

ι' *: is an intention dependent on* ι*.*

$\Psi(\iota, \iota')$ *: is a function that returns the weight of the dependency between intentions* ι *and* ι' *(see definition 6.1.4).*

$\Lambda(\iota)$ *: is any commitment the agent has accumulated and attributed directly to intention* ι *(see definition 6.1.5).*

$\Pi(\iota)$ *: is the initial cost associated in adopting the intention (see definition 6.1.3).*

Notice the utilization of the definition of cost in the calculation of total commitment. That is because as an agent behaves this cost changes also. Should a dependency be broken for any reason the commitment the agent has towards the involved intention is automatically adjusted accordingly.

Returning again to the office robot example, the state of the agent's intentions after

the decision to source the coffee from the cafeteria is:

$$\mathcal{I}_{t_5} = \{\text{'outsource document'}, \text{'deliver coffee'}, \text{'coffee from cafeteria'}\}$$
$$\Pi(\text{'coffee from cafeteria'}) = \frac{\Omega(\text{'coffee from cafeteria'})}{2}$$
$$\Pi(\text{'outsource document'}) = \frac{\Omega(\text{'outsource document'})}{2}$$
$$\Pi(\text{'deliver coffee'}) = \Omega(\text{'deliver coffee'})$$
$$\Psi(\text{'deliver coffee'}, \text{'outsource document'}) = 1/2$$
$$\Psi(\text{'deliver coffee'}, \text{'coffee from cafeteria'}) = 0$$
$$\Psi(\text{'deliver coffee'}, \text{'coffee from machine'}) = 0$$
$$\Psi(\text{'outsource document'}, \text{'coffee from cafeteria'}) = 1/2$$
$$\Psi(\text{'outsource document'}, \text{'deliver coffee'}) = 0$$
$$\Psi(\text{'outsource document'}, \text{'coffee from machine'}) = 0$$
$$\Psi(\text{'coffee from cafeteria'}, \text{'deliver coffee'}) = 0$$
$$\Psi(\text{'coffee from cafeteria'}, \text{'outsource document'}) = 0$$
$$\Psi(\text{'coffee from cafeteria'}, \text{'coffee from machine'}) = 0$$
$$\Lambda(\text{'deliver coffee'}) = 0$$
$$\Lambda(\text{'outsource document'}) = 0$$
$$\Lambda(\text{'coffee from cafeteria'}) = \frac{\Omega(\text{'coffee from machine'})}{2}$$

This leads to the agent placing the following commitment in its intention towards sourcing coffee from the cafeteria:

$$\Theta(\text{'coffee from cafeteria'}) = \left(\sum_{\iota' \in \mathcal{I}_{t_5}} \Theta(\iota') \times \Psi(\text{'coffee from cafeteria'}, \iota')\right) + \Lambda(\text{'coffee from cafeteria'}) + \Pi(\text{'coffee from cafeteria'})$$

$$= \Theta(\text{'deliver coffee'}) \times \Psi(\text{'coffee from cafeteria'}, \text{'deliver coffee'}) + \Theta(\text{'outsource document'}) \times \Psi(\text{'coffee from cafeteria'}, \text{'outsource document'}) + \Theta(\text{'coffee from machine'}) \times \Psi(\text{'coffee from cafeteria'}, \text{'coffee from machine'}) + \Lambda(\text{'coffee from cafeteria'}) + \Pi(\text{'coffee from cafeteria'})$$

$$
\begin{aligned}
&= \Theta(\text{`deliver coffee'}) \times 0 + \Theta(\text{`outsource document'}) \times 0 \\
&+ \Theta(\text{`coffee from machine'}) \times 0 + \Lambda(\text{`coffee from cafeteria'}) \\
&\qquad + \Pi(\text{`coffee from cafeteria'})
\end{aligned}
$$

$$
\begin{aligned}
&= \Lambda(\text{`coffee from cafeteria'}) + \Pi(\text{`coffee from cafeteria'}) \\
&= \frac{\Omega(\text{`coffee from machine'})}{2} + \Pi(\text{`coffee from cafeteria'}) \\
&= \frac{\Omega(\text{`coffee from machine'})}{2} + \frac{\Omega(\text{`coffee from cafeteria'})}{2} \\
&= (\Omega(\text{`coffee from machine'}) + \Omega(\text{`coffee from cafeteria'})) \times 1/2
\end{aligned}
$$

It also leads to the agent placing the following commitment in its intention to outsource the document delivery:

$$
\begin{aligned}
\Theta(\text{`outsource document'}) = &\left( \sum_{\iota' \in \mathcal{I}_{t_5}} \Theta(\iota') \times \Psi(\text{`outsource document'}, \iota') \right) \\
&+ \Lambda(\text{`outsource document'}) + \Pi(\text{`outsource document'})
\end{aligned}
$$

$$
\begin{aligned}
&= \Theta(\text{`deliver coffee'}) \times \Psi(\text{`outsource document'}, \text{`deliver coffee'}) \\
&+ \Theta(\text{`coffee from cafeteria'}) \times \Psi(\text{`outsource document'}, \text{`coffee from cafeteria'}) \\
&+ \Theta(\text{`coffee from machine'}) \times \Psi(\text{`outsource document'}, \text{`coffee from machine'}) \\
&\qquad + \Lambda(\text{`outsource document'}) + \Pi(\text{`outsource document'})
\end{aligned}
$$

$$
\begin{aligned}
&= \Theta(\text{`deliver coffee'}) \times 0 + \Theta(\text{`coffee from cafeteria'}) \times 1/2 \\
&+ \Theta(\text{`coffee from machine'}) \times 0 + \Lambda(\text{`outsource document'}) \\
&\qquad + \Pi(\text{`outsource document'})
\end{aligned}
$$

$$
\begin{aligned}
&= \Theta(\text{`coffee from cafeteria'}) \times 1/2 + \Lambda(\text{`outsource document'}) \\
&\qquad + \Pi(\text{`outsource document'})
\end{aligned}
$$

$$
\begin{aligned}
&= \Theta(\text{`coffee from cafeteria'}) \times 1/2 + \Pi(\text{`outsource document'}) \\
&= \Theta(\text{`coffee from cafeteria'}) \times 1/2 + \frac{\Omega(\text{`outsource document'})}{2}
\end{aligned}
$$

$= (\Theta(\text{`coffee from cafeteria'}) + \Omega(\text{`outsource document'})) \times 1/2$

Finally, the commitment the agent places in delivering the coffee is :

$$\Theta(\text{`deliver coffee'}) = \left( \sum_{\iota' \in \mathcal{I}_{\iota_5}} \Theta(\iota') \times \Psi(\text{`deliver coffee'}, \iota') \right) + \Lambda(\text{`deliver coffee'}) + \Pi(\text{`deliver coffee'})$$

$$\begin{aligned} &= \Theta(\text{`outsource document'}) \times \Psi(\text{`deliver coffee'}, \text{`outsource document'}) \\ &+ \Theta(\text{`coffee from machine'}) \times \Psi(\text{`deliver coffee'}, \text{`coffee from machine'}) \\ &+ \Theta(\text{`coffee from cafeteria'}) \times \Psi(\text{`deliver coffee'}, \text{`coffee from cafeteria'}) \\ &+ \Lambda(\text{`deliver coffee'}) + \Pi(\text{`deliver coffee'}) \end{aligned}$$

$$\begin{aligned} &= \Theta(\text{`outsource document'}) \times 1/2 + \Theta(\text{`coffee from machine'}) \times 0 \\ &+ \Theta(\text{`coffee from cafeteria'}) \times 0 + \Lambda(\text{`deliver coffee'}) + \Pi(\text{`deliver coffee'}) \\ &= \Theta(\text{`outsource document'}) \times 1/2 + \Lambda(\text{`deliver coffee'}) + \Pi(\text{`deliver coffee'}) \\ &= \Theta(\text{`outsource document'}) \times 1/2 + \Pi(\text{`deliver coffee'}) \\ &= \Theta(\text{`outsource document'}) \times 1/2 + \Omega(\text{`deliver coffee'}) \end{aligned}$$

6.3 Propagating Revision

In section 6.1 a methodology for capturing the commitment an agent has towards its evolving intentions was proposed. One of the primary uses for such a methodology is in the specification of when it is rational for an agent to reconsider these intentions in light of changing commitments. This takes the form of an intention revision predicate that holds of an intention and a set of intentions whenever the intention should be reconsidered given the set of intentions provided. Initially, a number of revision predicates are defined that revise an intention dependent on changes of the given intentions dependencies. The revision predicates are presented in order of sophistication with subsequent predicates overcoming a limitation with the predicates presented prior. This is followed by revision predicates where the effect on dependent intentions is the primary consideration in deciding whether to revise a given intention. Again, this sequence of predicates is presented in such a fashion that later predicates overcome the limitations of those presented earlier.

6.3.1 Changed Dependencies Reconsideration

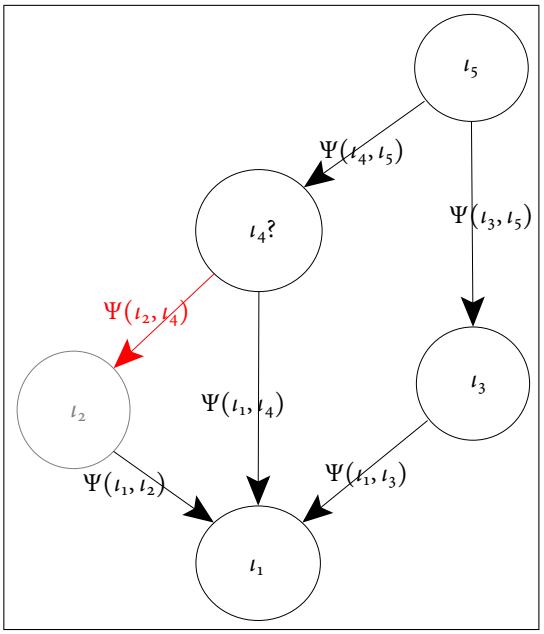


Figure 6.4: Changed Dependencies

The simplest policy is to reconsider an intention when any of its dependencies change. In the case of figure 6.4, upon the removal of ι_2, the reconsideration of ι_4 would be necessary.

6.3.1 Definition:

$$\text{revise}(\iota, \mathcal{I}_t) = \begin{cases} \top & \text{if } \exists \iota' \,.\, \iota' \in \mathcal{I}_{t'} \wr \mathcal{O}_{t'} \wedge \iota' \notin \mathcal{I}_t \\ \bot & \text{otherwise} \end{cases}$$

where:

$\mathcal{I}_t$ *: is the agent's current set of intentions.*

ι *: is the intention that is the topic of the reconsideration process.*

ι' *: is an intention that was used in the decision to adopt* ι *but is no longer held.*

$\mathcal{I}_{t'} \wr \mathcal{O}_{t'}$ *: is the set of intentions that were in conflict with any of the options under consideration during the selection of* ι *at time* t'*.*

t *: is the current time.*

t' *: the time at which the decision to adopt* ι *was made.*

If this is applied to the office robot scenario at the point at which the agent has committed to outsource the document delivery request (t_2):

$$\mathcal{O}_{t_2} = \{\text{'deliver document'}, \text{'outsource document'}\}$$
$$\mathcal{I}_{t_2} = \{\text{'deliver coffee'}\}$$
$$\mathcal{O}_{t_2} \wr \mathcal{I}_{t_2} = \{\text{'deliver document'}\}$$
$$\mathcal{I}_{t_2} \wr \mathcal{O}_{t_2} = \{\text{'deliver coffee'}\}$$

the effect of the agent dropping the commitment to deliver coffee on the commitment towards outsourcing the delivery of the document is:

$$\text{revise}(\text{'outsource document'}, \{\text{'outsource document'}\}) = \top$$

that the agent will reconsider its commitment towards outsourcing the document. This is because:

$$\text{'deliver coffee'} \in \mathcal{I}_{t_2} \wr \mathcal{O}_{t_2} \wedge \text{'deliver coffee'} \notin \{\text{'outsource document'}\}$$

Conversely, the effect on dropping the intention towards outsourcing the document delivery:

$$\text{revise}(\text{'deliver coffee'}, \{\text{'deliver coffee'}\}) = \bot$$

is to not reconsider its intention towards the delivery of coffee because at the time (t_1) the agent adopted the intention to 'deliver coffee' the agent had no existing intentions to influence this decision:

$$\mathcal{O}_{t_1} = \{\text{'deliver coffee'}, \text{'outsource coffee'}\} \qquad \mathcal{I}_{t_1} = \mathcal{O}_{t_1} \wr \mathcal{I}_{t_1} = \mathcal{I}_{t_1} \wr \mathcal{O}_{t_1} = \varnothing$$

While the above formalisation[3] utilises the agent's intentions and the options it considers in adopting it's intentions over time, an implementation need not retain these sets to achieve equivalent decisions. By associating the weights of the dependencies with both depending and dependant intentions and updating these weights to sentinel values when an intention is revised that breaks such a dependency, an implementation can reconstruct the required sets as necessary.

[3] This holds true for all the revision predicates in this section.

6.3.2 Proportional Changed Dependencies Reconsideration

Alternatively, an agent could wait until a fixed proportion (p) of the intentions used in the adoption of ι have been dropped before reconsidering:

(6.3.2) Definition:

$$\text{revise}(\iota, \mathcal{I}_t) = \begin{cases} \bot & \text{if } \mathcal{I}_{t'} \wr \mathcal{O}_{t'} = \varnothing \\ \top & \text{if } |\mathcal{I}_{t'} \wr \mathcal{O}_{t'} \cap \mathcal{I}_t| / |\mathcal{I}_{t'} \wr \mathcal{O}_{t'}| < p \\ \bot & \text{otherwise} \end{cases}$$

where:

(t) *: is the current time.*

(t') *: was the time at which ι was adopted.*

($\mathcal{I}_{t'} \wr \mathcal{O}_{t'}$) *: is the set of intentions that were in conflict with any of the options under consideration during the selection of ι at time t'.*

(p) *: is the reconsideration threshold (a real number in the range $[0, 1]$).*

The first case handles scenarios in which the intention under consideration was adopted in isolation or the agent's prior intentions had no impact on the decision process. Given there were no relevant intentions in effect when the decision to adopt ι was made, changes to the subsequent intentions are irrelevant in deciding whether to reconsider ι. Thus, irrespective of these changes, ι should be retained. The second case captures scenarios in which the agent held a number of intentions and these intentions were a consideration throughout decision making. The decision is then based upon the proportion of commonality between the current intentions and those applicable to the time of decision making. The intersection of the intentions that were utilized during decision making ($\mathcal{I}_{t'} \wr \mathcal{O}_{t'}$) and those in the agent's current intentions ($\mathcal{I}_t$) represents the intentions that have not changed. The larger the proportion of retained intentions given the decision relevant intentions for the intention in question the less that has changed. Thus, it is only once this proportion falls below the specified threshold that sufficient change has occurred to warrant reconsideration. The final case captures scenarios in which none of the others are applicable.

Again, when applied to the office robot scenario at the point at which the agent has committed to outsource the document delivery request (t_2):

$\mathcal{O}_{t_2} = \{$'deliver document', 'outsource document'$\}$
$\mathcal{I}_{t_2} = \{$'deliver coffee'$\}$
$\mathcal{O}_{t_2} \wr \mathcal{I}_{t_2} = \{$'deliver document'$\}$
$\mathcal{I}_{t_2} \wr \mathcal{O}_{t_2} = \{$'deliver coffee'$\}$

the effect of the agent dropping the commitment to deliver coffee on the commitment towards outsourcing the delivery of the document is:

$$\text{revise}(\text{'outsource document'}, \{\text{'outsource document'}\}) = \top$$

that the agent will reconsider its commitment towards outsourcing the document. This is due to $\mathcal{I} \wr_{t'}$'outsource document' = {'deliver coffee'} which results in the second clause of the definition being applicable.

Conversely, the effect on dropping the intention towards outsourcing the document delivery:

$$\text{revise}(\text{'deliver coffee'}, \{\text{'deliver coffee'}\}) = \bot$$

is to not reconsider its intention towards the delivery of coffee. This follows from the first clause of the definition which applies when there were no intentions utilized in the decision making process.

6.3.3 Weighted Proportional Reconsideration

However, utilizing such policies fails to take into account how much of the cost of intending a given intention was based on the existing intentions and how much relates to its merits. This cost is distributed between the new intention and those pre-existing proportionally to the role they played in the adoption process. These weights are stored in the dependencies between intentions.

(6.3.3) Definition:

$$\text{revise}(\iota, \mathcal{I}_t) = \begin{cases} \top & \textit{if } \left(\sum_{\iota' \in \mathcal{I}_{t'} \wr \mathcal{O}_{t'} \cap \mathcal{I}_t} \Omega(\iota) \times \Psi(\iota', \iota)\right) + \Pi(\iota) < p \times \Omega(\iota) \\ \bot & \textit{otherwise} \end{cases}$$

where:

$\boxed{\Omega(\iota)}$ *: is a function that can calculate the total cost of intending* ι *(see section 6.2.1 for an example).*

$\boxed{\Psi(\iota', \iota)}$ *: is a function that returns the weight of the dependency between intentions* ι *and* ι' *(see definition 6.1.4).*

$\boxed{\Pi(\iota)}$ *: is the initial cost associated in adopting the intention (see definition 6.1.3).*

$\boxed{p}$ *: is the reconsideration threshold (a real number in the range* $[0, 1]$*).*

Intuitively, if the commitment the agent has towards an intention falls below a proportion of the cost of intending it then the agent should reconsider. Given that the commitment towards an intention contains a base commitment such a condition may never be met. It may be the case that for all time the base commitment exceeds the reconsideration threshold. In such cases, the agent does not consider the intentions dependencies when deciding whether to reconsider. However, when the base commitment falls below the reconsideration threshold then the agent allows for some changes to the intentions dependencies before reconsidering. As time passes, the intentions relevant to the decision to adopt a given intention will either complete or fail. As these intentions are dropped, the left side of the inequality above approaches the value of the base commitment (the commitment accumulates in the intention in question). Provided the base commitment is less than the reconsideration threshold, once sufficient dependent intentions have been dropped, the depending intention will be reconsidered.

The application of this propagation strategy to the office robot domain yields, initially:

$$\mathcal{I}_{t_1} = \varnothing \qquad \mathcal{I}_{t_1} \wr \mathcal{O}_{t_1} = \varnothing$$
$$\Pi(\text{`deliver coffee'}) = \Omega(\text{`deliver coffee'})$$
$$\Omega(\text{`deliver coffee'}) \not< p \times \Omega(\text{`deliver coffee'}) \text{ for any } p \in [0, 1]$$
$$\therefore \quad \text{revise}\,(\text{`deliver coffee'}, \{\text{`deliver coffee'}, \text{`outsource document'}\}) = \bot$$

a decision not to revise the intention towards delivering coffee. It also follows that

the intention towards outsourcing the document won't be reconsidered either:

$$\mathcal{O}_{t_2} = \{\text{'deliver document'}, \text{'outsource document'}\}$$
$$\mathcal{I}_{t_2} = \{\text{'deliver coffee'}\}$$
$$\mathcal{O}_{t_2} \wr \mathcal{I}_{t_2} = \{\text{'deliver document'}\}$$
$$\mathcal{I}_{t_2} \wr \mathcal{O}_{t_2} = \{\text{'deliver coffee'}\}$$
$$\Pi(\text{'outsource document'}) = \Omega(\text{'outsource document'}) \times 1/2$$
$$\Psi(\text{'deliver coffee'}, \text{'outsource document'}) = 1/2$$
$$\mathcal{I}_{t_2} \wr \mathcal{O}_{t_2} \cap \{\text{'deliver coffee'}, \text{'outsource document'}\} = \{\text{'deliver coffee'}\}$$

$$\Omega(\text{'outsource document'}) \times \Psi(\text{'deliver coffee'}, \text{'outsource document'}) = \Omega(\text{'outsource document'}) \times 1/2$$

$$\Omega(\text{'outsource document'}) \times 1/2 + \Pi(\text{'outsource document'}) = \Omega(\text{'outsource document'}) \times 1/2 + \Omega(\text{'outsource document'}) \times 1/2 = \Omega(\text{'outsource document'})$$

$$\Omega(\text{'outsource document'}) \not< p \times \Omega(\text{'outsource document'}) \text{ for any } p \in [0, 1]$$

$$\therefore \quad \text{revise}(\text{'outsource document'}, \{\text{'deliver coffee'}, \text{'outsource document'}\}) = \bot$$

In the scenario in which the intention towards 'outsource document' fails or completes first, the following is entailed:

$$\Pi(\text{'deliver coffee'}) = \Omega(\text{'deliver coffee'})$$

and thus, for any value of p, due to the empty set of intentions at the time the intention to 'deliver coffee' was adopted:

$$\text{revise}(\text{'deliver coffee'}, \{\text{'deliver coffee'}\}) = \bot$$

In the scenario in which the intention towards 'deliver coffee' fails or completes first

and $p > 1/2$:

$\mathcal{O}_{t_2}$ = {'deliver document', 'outsource document'}
$\mathcal{I}_{t_2}$ = {'deliver coffee'}
$\mathcal{O}_{t_2} \wr \mathcal{I}_{t_2}$ = {'deliver document'}
$\mathcal{I}_{t_2} \wr \mathcal{O}_{t_2}$ = {'deliver coffee'}
Π('outsource document') = Ω('outsource document') × 1/2
Ψ('deliver coffee', 'outsource document') = 1/2
$\mathcal{I}_{t_2} \wr \mathcal{O}_{t_2} \cap$ {'outsource document'} = $\varnothing$

$$0 + \Pi(\text{'outsource document'}) = 0 + \Omega(\text{'outsource document'}) \times 1/2 = \Omega(\text{'outsource document'}) \times 1/2$$

Ω('outsource document') × 1/2 < p × Ω('outsource document') for any $p \in (1/2, 1]$
∴ revise ('outsource document', {'deliver coffee', 'outsource document'}) = $\top$

In the scenario in which $p < 1/2$ and the completion or failure of the intention towards 'deliver coffee' precedes that of 'outsource document':

revise('outsource document', {'outsource document'}) = $\bot$

as

Π('outsource document') = Ω('outsource document') × 1/2

and

Ω('outsource document') × 1/2 $\not<$ p × Ω('outsource document') for any $p \in [0, 1/2)$

6.3.4 Dependent Non-Reconsideration

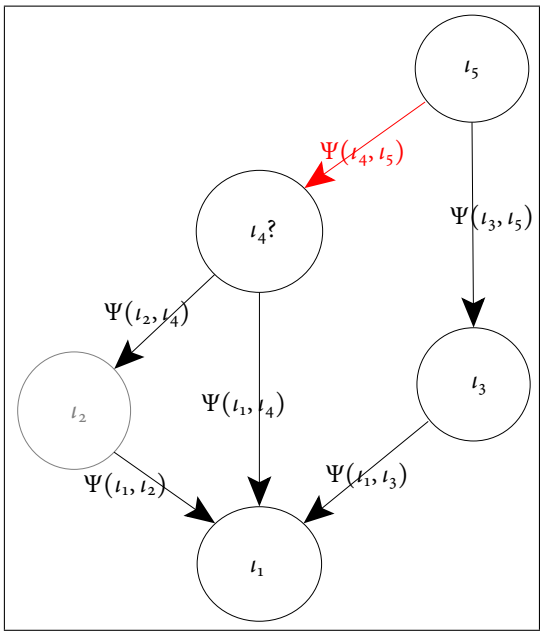


Figure 6.5: Intention Dependants

Although changes in the dependencies of an intention motivate reconsideration, the intentions dependent on it justify its non-reconsideration. This is because the dependants of a given intention require its stability. If the agent reconsiders, this further motivates the agent to reconsider all dependent intentions. Thus, an agent should take the potential cost of such a reconsideration into account when deciding to reconsider. One simple policy is to not reconsider any intention which has dependants. Given such a policy, ι_4 of figure 6.5 would be retained regardless of the changes to either ι_2 or ι_1 due to the commitment towards ι_5:

6.3.4 Definition:

$$\text{revise}(\iota, \mathcal{I}_t) = \begin{cases} \top & \textit{if } \{\iota' \in \mathcal{I}_t \mid \Psi(\iota, \iota') > 0\} = \varnothing \\ \bot & \textit{otherwise} \end{cases}$$

where:

$\mathcal{I}_t$ *: is the agent's current set of intentions.*

$\boxed{\iota}$ *: is the intention that is the topic of reconsideration.*

$\boxed{\Psi(\iota, \iota') > 0}$ *: indicates that there is a dependency between* ι *and* ι'*.*

Applying this to the office robot scenario at the point after the agent adopted an intention towards outsourcing the document (t_3), the effect of the agent dropping the commitment to deliver coffee on the commitment towards outsourcing the delivery of the document is:

$$\text{revise}(\text{`outsource document'}, \{\text{`outsource document'}\}) = \top$$

as

$$\{\iota \in \{\text{`outsource document'}\} \mid \Psi(\text{`outsource document'}, \iota) > 0\} = \varnothing$$

Thus, the agent will reconsider its commitment towards outsourcing the document. Similarly, the effect on dropping the intention towards outsourcing the document delivery:

$$\text{revise}(\text{`deliver coffee'}, \{\text{`deliver coffee'}\}) = \top$$

is for the agent to reconsider its intention towards delivering coffee. This is because neither commitment has any dependencies at this point.
However, when applied to the above prior to dropping either mentioned commitment:

$$\left\{ \iota' \in \left\{ \begin{array}{c} \text{`deliver coffee'}, \\ \text{`outsource document'} \end{array} \right\} \middle| \Psi(\text{`outsource document'}, \iota') > 0 \right\} = \varnothing$$

$$\therefore \quad \text{revise}\,(\text{`outsource document'}, \{\text{`deliver coffee'}, \text{`outsource document'}\}) = \top$$

and

$$\Psi(\text{`deliver coffee'}, \text{`outsource document'}) = 1/2$$

$$\{\iota' \in \{\text{`deliver coffee'}, \text{`outsource document'}\} \mid \Psi(\text{`deliver coffee'}, \iota') > 0\} = \{\text{`outsource document'}\}$$

$$\therefore \quad \text{revise}(\text{'deliver coffee'}, \{\text{'deliver coffee'}, \text{'outsource document'}\}) = \bot$$

the agent will reconsider its intentions towards outsourcing the document but not towards the delivery of coffee because its intention towards delivering coffee is a dependency of outsourcing the document, but outsourcing the document is not a dependent of any other intentions.

The requirement expressed by the above reconsideration rule is extreme. It is presented only as a simple starting point for reconsideration predicates that utilize the dependants of intentions. A more plausible reconsideration predicate is presented below.

6.3.5 Proportional Dependent Non-Reconsideration

Alternatively, an agent could only reconsider an intention provided that doing so will not result in more dependent intentions than a predefined limit being reconsidered:

(6.3.5) Definition:

$$\text{revise}(\iota, \mathcal{I}) = \begin{cases} \top & \textit{if } \frac{|\{\iota' \in \mathcal{I} |\ \Psi(\iota,\iota') > 0 \wedge \text{revise}(\iota', \mathcal{I} \setminus \{\iota\}) = \top\}|}{|\{\iota'' \in \mathcal{I} |\ \Psi(\iota,\iota'') > 0\}|} < p \\ \bot & \textit{otherwise} \end{cases}$$

where:

$(\mathcal{I})$ *: is the agent's current set of intentions.*

(ι) *: is the intention that is the topic of reconsideration.*

(p) *: the proportion of intentions that the agent is willing to potentially reconsider should it reconsider and drop the intention over which it is contemplating.*

This is illustrated through the use of the office robot example at time t_3:

$$\mathcal{I}_{t_3} = \{\text{'deliver coffee'}, \text{'outsource document'}\}$$

$$\iota = \text{'deliver coffee'}$$

$$\Psi(\text{'deliver coffee'}, \text{'outsource document'}) = 1/2$$

$$\{\iota' \in \mathcal{I}_{t_3} \mid \Psi(\text{'deliver coffee'}, \iota') > 0\} = \{\text{'outsource document'}\}$$

$$\{\iota' \in \mathcal{I}_{t_3} \mid \Psi(\text{'outsource document'}, \iota') > 0\} = \varnothing$$

$$\text{revise}(\iota, \mathcal{I}_{t_3}) = \frac{\left|\left\{\iota' \in \mathcal{I}_{t_3} \,\middle|\, \Psi(\iota, \iota') > 0 \wedge \text{revise}(\iota', \mathcal{I}_{t_3} \setminus \{\iota\}) = \top\right\}\right|}{\left|\left\{\iota'' \in \mathcal{I}_{t_3} \,\middle|\, \Psi(\iota, \iota'') > 0\right\}\right|}$$

$$= \frac{\left|\left\{\iota' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \text{revise}(\iota', \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \setminus \{\iota\}) = \top \wedge \Psi(\iota, \iota') > 0\right\}\right|}{\left|\left\{\iota'' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \Psi(\iota, \iota'') > 0\right\}\right|}$$

$$= \frac{\left|\left\{\iota' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \text{revise}(\iota', \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \setminus \{\text{'deliver coffee'}\}) = \top \wedge \Psi(\text{'deliver coffee'}, \iota') > 0\right\}\right|}{\left|\left\{\iota'' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \Psi(\text{'deliver coffee'}, \iota'') > 0\right\}\right|}$$

$$= \frac{\left|\left\{\iota' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \text{revise}(\iota', \{\text{'outsource document'}\}) = \top \wedge \Psi(\text{'deliver coffee'}, \iota') > 0\right\}\right|}{\left|\left\{\iota'' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \Psi(\text{'deliver coffee'}, \iota'') > 0\right\}\right|}$$

$$= \frac{\left|\left\{\iota' \in \left\{\begin{array}{c}\text{'deliver coffee'},\\ \text{'outsource document'}\end{array}\right\} \,\middle|\, \text{revise}(\iota', \{\text{'outsource document'}\}) = \top \wedge \Psi(\text{'deliver coffee'}, \iota') > 0\right\}\right|}{|\{\text{'outsource document'}\}|}$$

$$= \frac{|\{\iota' \in \{\text{'outsource document'}\} \mid \text{revise}(\iota', \{\text{'outsource document'}\}) = \top\}|}{|\{\text{'outsource document'}\}|}$$

$$= \frac{|\{\iota' \in \{\text{'outsource document'}\} \mid \text{revise}(\iota', \{\text{'outsource document'}\}) = \top\}|}{|\{\text{'outsource document'}\}|}$$

$$= \frac{|\{\text{'outsource document'}\}|}{|\{\text{'outsource document'}\}|}$$

$$= \frac{1}{1}$$

$$1 \not< p$$

$$\therefore \quad \text{revise}(\text{'deliver coffee'}, \{\text{'deliver coffee'}, \text{'outsource document'}\}) = \bot$$

and[4] subsequent ($t_{3'}$)[5] to the agent dropping its intention towards delivering coffee (assuming $p > 0$):

$$\mathcal{I}_{t_{3'}} = \{\text{'outsource document'}\}$$

[4]The derivation of revise('outsource document', {'outsource document'}) = ⊤ is shown below.

[5]The time is designated $t_{3'}$ instead of t_4 so as to avoid confusion with the alternative evolution of the agent's intentions given previously.

ι = 'outsource document'

$\left\{\iota' \in \mathcal{I}_{t_{3'}} \mid \Psi(\text{'outsource document'}, \iota') > 0\right\} = \varnothing$

$$\text{revise}(\iota, \mathcal{I}_{t_{3'}}) = \frac{\left|\left\{\iota' \in \mathcal{I}_{t_{3'}} \mid \Psi(\iota, \iota') > 0 \wedge \text{revise}(\iota', \mathcal{I}_{t_{3'}} \setminus \{\iota\}) = \top\right\}\right|}{\left|\left\{\iota'' \in \mathcal{I}_{t_{3'}} \mid \Psi(\iota, \iota'') > 0\right\}\right|}$$

$$= \frac{\left|\left\{\iota' \in \{\text{'outsource document'}\} \;\middle|\; \begin{array}{c}\Psi(\iota, \iota') > 0 \wedge \\ \text{revise}(\iota', \{\text{'outsource document'}\} \setminus \{\iota\}) = \top\end{array}\right\}\right|}{|\{\iota'' \in \{\text{'outsource document'}\} \mid \Psi(\iota, \iota'') > 0\}|}$$

$$= \frac{\left|\left\{\iota' \in \{\text{'outsource document'}\} \;\middle|\; \begin{array}{c}\Psi(\text{'outsource document'}, \iota') > 0 \wedge \\ \text{revise}(\iota', \{\text{'outsource document'}\} \setminus \{\text{'outsource document'}\}) = \top\end{array}\right\}\right|}{|\{\iota'' \in \{\text{'outsource document'}\} \mid \Psi(\text{'outsource document'}, \iota'') > 0\}|}$$

$$= \frac{|\varnothing|}{|\varnothing|} \qquad = \frac{0}{0}$$

$0 < p$

$\therefore \quad \text{revise}(\text{'outsource document'}, \{\text{'outsource document'}\}) = \top$

Clearly an agent should balance the required stability of its intentions against the opportunities arising due to its changing intentions. While the above reconsideration predicates utilize both the dependants and dependencies of intentions in the decision to reconsider, they do not utilize the commitment an agent places in each intention. This is because to do so requires an evaluation of the benefits of such a reconsideration to compare with the existing cost/commitment measurement. The measure of the benefit of a particular revision is an open question. Thus, utilizing the cost/commitment measurement developed above in reconsideration predicates and balancing opportunities against the stability of the agent's intentions remains an avenue of future work.

6.4 Intention Structure

When additional structure, beyond that generated through consistency maintenance, is present in an agent's intentions, such structure must be respected and modelled within any theory of commitment. The interpretation of intentions in AgentSpeak (section 2.1.3) provides such additional structure (see figure 6.6a). It models the interleaving execution of goals and the plans adopted for their achievement. In figure 6.6a $\eth_{x,y,z}$ represents the zth goal of the yth plan in intention x

and $\rho_{x,y}$ represents the yth plan of intention x. Plans are adopted for the achievement of goals. Goals are embedded as steps within plans. Dependencies may exist between the goals within plans (see figure 6.6b), in addition to those arising from consistency maintenance (see figure 6.6c). These dependencies between goals are represented as maintenance conditions where $\mu_{x,y}^{i,j}$ is a maintenance condition (as per definition 2.1.3) instantiated after goal i and holding until goal j of the yth plan on intention x. Goals are entirely dependent on the plans in which they lie. Plans are entirely dependent on the goals they are adopted to achieve. If a plan is revised or dropped then any embedded goals, too, will be dropped. In order to capture this relationship, the weight of the dependencies between plans and the goals adopted as part of their execution is the maximum possible (1)[6]. This ensures, when using any of the revision predicates (definitions 6.3.1 to 6.3.3) based on changing dependencies, that whenever a goal or plan is dropped the goals and plans adopted as part of their execution are dropped also.

(6.4.1) Definition (Inter-Plan Dependencies):

$$\Psi(\rho, \rho') = 1$$

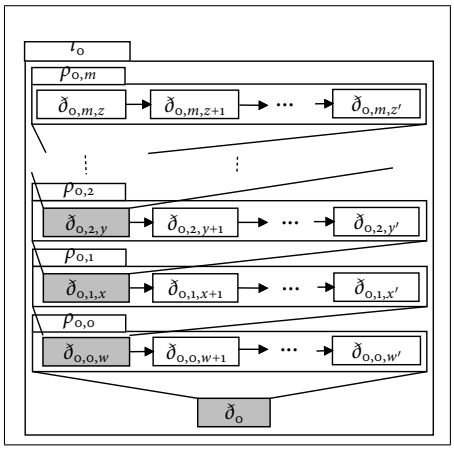
where:

(ρ) *: is an intended plan.*

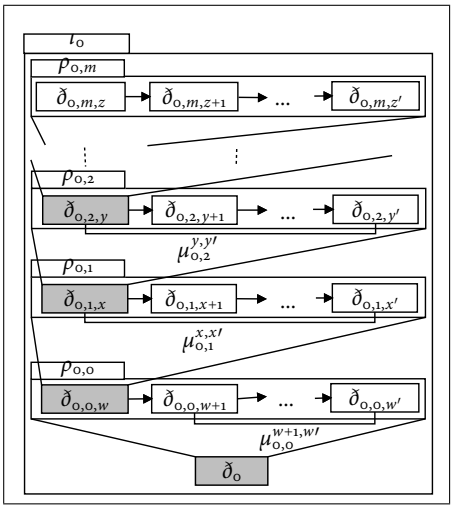
(ρ') *: is a plan intended to achieve a goal that is embedded in the intended plan ρ.*

A plan completes execution when there are no more goals in its body to execute or one of its success conditions is met. A plan fails when one of the goals in its body fails, its maintenance conditions are not satisfied or one of its listed failure conditions follows from the agent's beliefs. Both completion and failure provide motivation for the agent to drop the plan. In dropping the plan an agent finalizes its commitment towards it and propagates this to the plan in which the goal for which it was adopted is embedded.

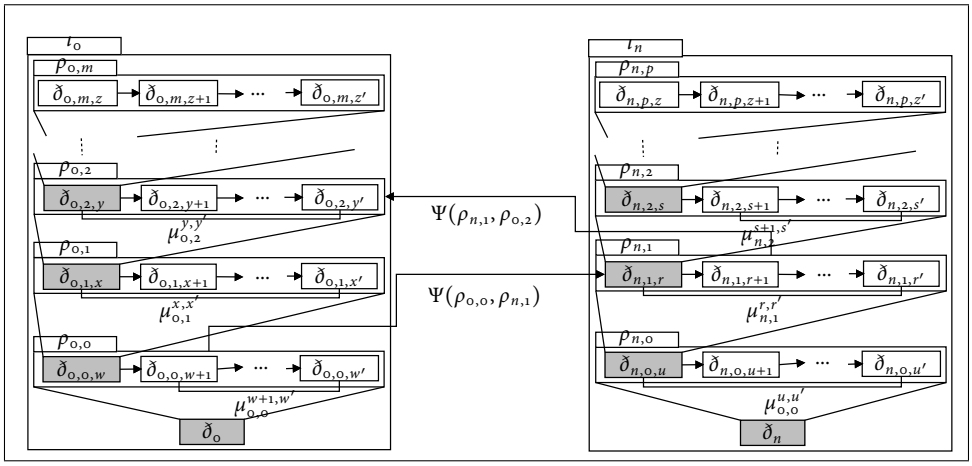
[6]In AgentSpeak, there are no *intentions* as such, only *intention stacks*, *plans*, *goals* and *maintenance conditions*. Thus, the abstract concept of intention utilized thus far is extended to include the intended plans of an AgentSpeak agent.



(a) Single Intention



(b) Intention with Maintenance Conditions



(c) Multiple Interdependent Intentions

Figure 6.6: AgentSpeak Intention Structures

6.4.2 Definition (Plan Completion):

$$\Lambda(\rho) ::= \Lambda(\rho) + \Lambda(\rho') + \Omega(\rho')$$

where:

ρ *: is the intended plan in which the goal for which the dropped plan was adopted.*

ρ' *: is the plan adopted and subsequently dropped or revised to achieve a goal embedded in* ρ*.*

$\Lambda(\rho)$ *: is the commitment the agent has accumulated towards* ρ*.*

$\Lambda(\rho')$ *: is the accumulated commitment the agent has accrued throughout the lifetime of the plan* ρ'*.*

$\Omega(\rho')$ *: is the cost incurred to adopt the plan* ρ' *(see section 6.2.1).*

Intuitively, this models the fact that adopting and executing a plan ρ' with the intention of achieving goal $\eth$ embedded in plan ρ furthers the agent's commitment towards ρ upon ρ's completion. Only the commitment the plan ρ' has accumulated directly, including the initial cost of adoption, is propagated. Once dropped, or revised, any dependencies existent on ρ' become invalid. Thus, propagating commitment from these sources would over commit the agent to ρ.

If the achievement and finalization of goal $\eth$ causes the completion of plan ρ, then the procedure defined in definition 6.4.2 repeats until an incomplete intended plans is found or the independent originating intention is reached.

6.5 Related Work

Similar intuitions pertaining to the relationship between cost and consistency were developed in [53]. A theory by which options are evaluated in the context of an agent's existing plans was presented. The plans considered were non-hierarchical and completely specified with actions with deterministic and instantaneous outcomes. Actions within plans were related by both temporal and causal constraints. Thus, the order of execution of the actions within plans were only partially specified. A number of degrees of consistency between such plans were defined: perfectly, strongly and weakly consistent. Two plans were deemed *perfectly* consistent

if there was a linearization of the actions within both plans without the need to modify the properties of the plans in any way. *Strongly* consistent plans are such that the actions within each plan can by linearly scheduled through the addition of temporal constraints between the execution of actions. Two plans were defined as *weakly* consistent provided the actions contained within were linearly executable given freedom to add temporal constraints, causal constraints or additional time points.

The cost of a plan was defined as the least expensive way in which it could be linearly scheduled given real valued costs for the constituent actions. The contextual cost of a plan that is strongly consistent with an agent's existing plans was defined as the cost of the linearization of the plan within the agent's existing plans less the cost of the agent's existing plans alone. Because the context may be such that a given option can be merged and the execution of actions shared, the contextual cost of a plan may be less than the sum of the independent cost of the option and the cost of the agent's context. Conversely, because the integration of an option into the agent's plans may require the addition of constraints or actions, the contextual cost can be greater than the sum of the option's independent cost and the cost of the agent's existing plans. Given a measure of the benefit of a particular option, a means of determining the value of adopting a particular option was presented. This mechanism was shown to be robust against errors in the estimation of cost estimates. Any-time algorithms were developed to calculate both independent and contextual costs for options.

There are a number of factors that differentiate the work described within this chapter from that of [53]. This work considers the dynamic and ongoing cost and commitment an agent holds towards a given adoption. The plans are considered as unmodifiable for the purposes of adoption and thus do not facilitate degrees of consistency. The plans are also both abstract and hierarchical and therefore some of the considerations with regard to scheduling individual actions lie outside the adoption process and cost calculation.

6.6 Summary

This chapter presents mechanisms for deciding whether to reconsider intentions, in light of their relation to other intentions, and the extent to which they affected earlier decisions. These relations between intentions arise as a consequence of the requirement that an agent's intentions be consistent. Using these relations as a basis, a technique to calculate the commitment an agent should hold toward its intentions

was presented. A number of intention changing operations were identified and for each operation of intention change, a mechanism to update the agent's commitments appropriately was developed. Through the application of both the reconsideration policies and commitment calculations to a running example based on an office robot, the plausibility of the approach was argued. These techniques were then specialized to capture the model of intentions utilized in AgentSpeak agents.

This approach to quantifying commitment and intention revision propagation differs significantly from the intention reconsideration policies outlined in section 2.1.3. The agents under consideration there are planning agents that generate, adapt and adopt a single complete plan for the achievement of their intentions. The agents considered here are those that adopt multiple independent, though consistent, intentions. Consequently the intention structure considered is richer and facilitates additional relationships between intentions. The dynamics of these relationships were used to identify new intention reconsideration policies.

Another aspect unique to the approach above is the explicit modelling of the commitments an agent makes towards its intentions. Typically, commitment is an emergent property of the intention revision policy of the agent. Without explicit representation, reasoning about the commitment an agent places in an intention is impossible. Although the theory presented has this commitment explicitly represented, an interpretation of these commitment values that is compoundable is lacking. Consequently, it is not possible to reason about the commitment towards compound intentions by way of the commitments towards the constituent intentions.

A hermit is a deserter from the army of humanity.

Unknown.

7 Implementation

In the course of developing and testing the ideas presented in chapters 3, 4 and 6 a new implementation of the AgentSpeak language was developed. This was necessary due to the perceived deficiencies of the implementations of the time. Many of the existing AgentSpeak-like implementations were concerned primarily with the development of a single "authoritative" language. New syntax and semantics were integrated into these preferred implementations, making the previous approaches obsolete. However, the need for evaluating these differing alternatives and facilitating additional experimentation, with both syntax and semantics, required a much more modular approach and philosophy. As such, the implementation developed possesses a number of unique characteristics that are worth noting. These include the use of island grammars to dynamically build syntactic analysers for a variety of AgentSpeak-like dialects, extensive use of the proxy pattern to construct classes dynamically, dependency injection to handle configuration and customization, a multi-language and thus layered development approach and unique network-aware control and debugging facilities. This chapter will discuss these techniques in some detail before detailing how these unique features facilitated implementation of the core concepts of chapters 3, 4 and 6 and the degree to which these are complete.

7.1 Techniques

This section will detail the unique aspects of the new implementation. Some of these techniques, such as immutable objects and dependency injection, are well-known and widely utilized for the implementation of modular systems. Others, such as island grammars and dynamic object construction, are less common[1]. How-

[1]This is particularly so given Java as the chosen implementation language.

ever, it is the particular combination that differentiates the new implementation from the current state of the art in AgentSpeak interpreters. In some respects, the implementation, although new, shares characteristics with existing systems. It is written in the Java programming language and as such the software design is primarily object-oriented in nature. It provides support for the original AgentSpeak(L) language in addition to multiple new dialects. However, it is the novel aspects of the implementation that bears emphasis.

7.1.1 Island Grammars

Island grammars are a mechanism by which the state of lexical analysers (lexers) and syntactic analysers (parsers) may be shared across language definitions to facilitate the embedding of one language in another. A lexical analyser is a function that maps a stream of input characters into a stream of semantic tokens. A syntactic analyser is an implementation of a grammar that accepts only token streams that match the rules of the grammar. The output of a syntactic analyser match is an abstract syntax tree (AST) that provides an unambiguous structural description of the input. Such a tree can then be traversed to construct the semantic objects of the language parsed or execute the language directly. By utilizing island grammars in the development of syntactic analysers for the AgentSpeak and AgentSpeak-like languages it is possible to decompose the syntactic analyser problem into sub-languages that can be individually modified or replaced. The AgentSpeak language is made up of 6 such sub-languages:

1. The language of stored beliefs. This language describes the objects that can be stored directly in the belief base of the agent[2].

2. The language to query an agent's beliefs. This language describes the structure of valid queries that can be posed to a given belief base. In many cases this will be equivalent to the language of stored beliefs, but in the case of AgentSpeak proper the two differ. The query language of an AgentSpeak agent's beliefs additionally supports variables, conjunction and default negation.

3. The language of actions. This defines the syntax used to describe the actions that are directly executable within the environment.

[2] Technically, the language of stored beliefs supports variables, but will issue a run-time error when a variable isn't ground.

4. The language of goals. This language details the syntax used to express the various goals types that constitute the body of plans.

5. The language of triggers. This describes the language used to match against the events generated throughout the execution of an AgentSpeak agent.

6. The language of plans. This language provides the syntax that separates the trigger, context and body elements of an AgentSpeak plan. It passes control of the syntactic analyser state to the syntactic analysers of the appropriate sub-languages as necessary.

Figure 7.1 demonstrates the embedding of the multiple sub-languages to construct the resulting AgentSpeak grammar. The syntactic analyser used is indicated by the numbering of the bracketing. The numbers are as per the list above.

This feature was utilized to build syntactic analysers that extended the syntax of plans to include the specification of effects without need to modify the syntactic analysers for goals, beliefs or events. In so doing, an additional syntactic analyser was developed for the language of effects and a new plan syntax analyser developed to call into this sub-language as necessary.

Extending the modularity provided by the use of island grammars is the complete separation of the syntactic form of a semantic object from the representation of the semantic object itself. This was achieved through the use of separate abstract syntax tree walkers. These traverse the syntax tree constructing the semantic objects separately from the construction of the syntax tree by the syntactic analyser. This facilitates the modular development of the syntactic analysers and the semantic objects. Thus, the same semantic objects can be constructed from differing syntax and the same syntactic analyser can be brought to bare to construct differing semantic objects.

There are some issues that the use of island grammars entail. Because of the additional layers of abstraction throughout the parsing process and the maintenance of the state of multiple analysers, there is a computational cost involved. However, because AgentSpeak agents are intended as long lived processes the additional parsing cost is negligible, particularly considering the advantages such an approach provides in a language research platform. Another problem lies in the definition of the grammars themselves. All sub-languages must be self-contained. It is not possible to have rules that require an arbitrary number of tokens. Each rule must contain a token by which the end of an arbitrary sequence can be detected. This can often require the addition of artificial tokens, that in the context of the resulting grammar seem redundant. Furthermore, the current implementation does

Rubbish Robot Example

```
1  ⌈[1]adjacent(a,b).
2  adjacent(b,c).
3  adjacent(c,d).
4  adjacent(d,a).
5  location(robot,a).
6  location(bin,d).[1]⌋

7
8  ⌈[6]⌈[5]+!⌈[1]location(robot,X)[1]⌋[5]⌋ : ⌈[2]location(robot,X)[2]⌋
9    <-
10       ⌈[4]TRUE[4]⌋.[6]⌋
11
12 ⌈[6]⌈[5]+!⌈[1]location(robot,X)[1]⌋[5]⌋ : ⌈[2]location(robot,Y)
13       & adjacent(Y,Z)
14       & not(location(robot,X))[2]⌋
15   <-
16       ⌈[4]⌈[3]move(Y,Z)[3]⌋[4]⌋;
17       ⌈[4]!⌈[1]location(robot,X)[1]⌋[4]⌋.[6]⌋
18
19 ⌈[6]⌈[5]+⌈[1]location(waste,X)[1]⌋[5]⌋ : ⌈[2]not(location(robot,X))
20       & location(bin,Y)[2]⌋
21   <-
22       ⌈[4]!⌈[1]location(robot, X)[1]⌋[4]⌋;
23       ⌈[4]⌈[3]pick(waste)[3]⌋[4]⌋;
24       ⌈[4]!⌈[1]location(robot, Y)[1]⌋[4]⌋;
25       ⌈[4]⌈[3]drop(waste)[3]⌋[4]⌋.[6]⌋
```

Figure 7.1: Island Grammars applied to AgentSpeak

not support look-ahead and backtracking of tokens across grammar boundaries. Thus, whenever a grammar must make a choice towards which island-grammar to apply, there must be an element of syntax by which the grammar can detect the correct grammar. This can require the insertion of artificial tokens. This issue is manifest when dealing with parsing goal types. Each unique goal type requires a unique prefix to ensure the correct goal type is instantiated. Because of this a custom goal parser is required for every combination of compatible goal types that a user wishes to utilize.

7.1.2 Dynamic Class Construction

Multiplying the utility of constructing syntactic analysers at run-time, is the extensive use of dynamically constructed classes. This enables the semantic objects constructed in the application of dynamically constructed syntactic analysers to be decorated with the semantic processes supporting the syntactic additions. An example from the AgentSpeak interpreter, as previously discussed (section 2.1.3), is the way the plan semantic objects are built. The body of AgentSpeak plans are sequences of goals[3]. Thus, there is a corresponding goal sequence object. Such goal sequences are only applicable given a condition on the agent's beliefs holds. There is, therefore, a guarded plan decorator object that encapsulates the process of testing this condition. This decorator can be applied to the goal sequence object to produce an object that satisfies the interfaces of both goal sequences and guarded plans. Furthering this, is the need for AgentSpeak plans to declare the events to which they respond. To capture the process of doing so is the event handler decorator. This decorator is applied to the object resulting from the application of the guarded plan decorator to the goal sequence object to produce the final AgentSpeak plan object. Such decorators constitute an interface, an implementation object, a wrapped object and a dynamically created proxy object. The interface defines the methods to decorate the wrapped object with. The implementation object provides the implementation of these new methods. The dynamically created proxy object is an intermediary object that has an interface that is the union of the interfaces implemented by the wrapper object and the decorator interface (see figure 7.2 for a UML depiction of these relationships). It dispatches method calls on the decorator interface to the implementation object, and the remaining method invocations to the wrapped object. Thus, care must be taken to ensure the correct method is called during manual dispatch.

[3]To simplify the implementation and the discussion actions are wrapped in *"do"* goals for which there is no syntax.

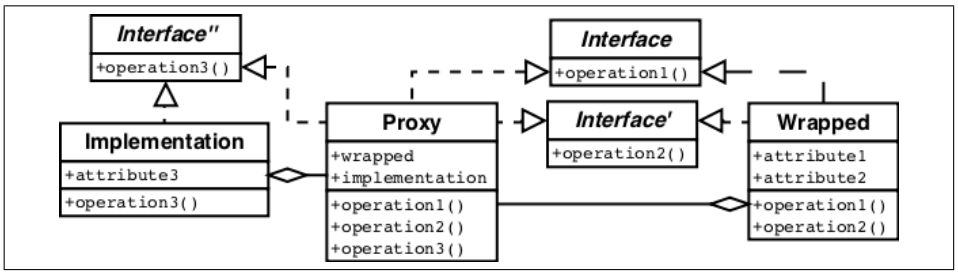


Figure 7.2: Decorator UML

The application of the decorators are independent of each other. Thus it is possible to construct goal sequences that declare the event to which they respond without declaring the context in which they are applicable. Similarly, it is possible to use goal trees, goal graphs, or any other body structure in place of goal sequences, and construct plans with additional capabilities analogously. In the same way using island grammars removes the need to explicitly write syntactic analysers for each combination of syntactic features of a given language, the use of dynamic proxy objects obviates the requirement to explicitly write the corresponding semantic objects.

Because dynamic proxy objects bypass much of the Java compiler's control on object construction, some difficulties arise from their use. Custom method dispatch code on the generated classes must be written. This code is heavily reliant on reflection and type introspection and as a result will be slower than the inbuilt equivalents. Care must be taken in the declaration of the interfaces from which the dynamically generated class will be built to ensure there are no naming conflicts. When problems arise, the lack of support in the Java infrastructure for the debugging of dynamically constructed classes becomes a hindrance to their resolution.

7.1.3 Dependency Injection

Dependency injection is a program design technique that favours the injection of dependencies into an object at the time of construction in preference to the use of "setter" methods. This has a number of advantages. For example, once constructed, an object will be known to be fully instantiated and ready for use. It also eases testing as stub objects can be utilized in place of the more complicated implementations utilized in the final product. It also facilitates the automatic deduction of the order of object construction and there are tools available that do just this. Provided with a set of classes, such tools can automatically instantiate a requested

object type with its dependencies fulfilled. This is utilized in the implementation of AgentSpeak as the means for constructing customized architectures. Custom selection functions for events, options and intentions can be automatically injected based on configuration files. Custom syntactic analysers for each of the components discussed in section 7.1.1 can be injected to customize the AgentSpeak grammar. Custom belief, intention, or plan bases can be injected to take advantage of the extra features made available by the aforementioned custom syntactic analysers. As a consequence of the adherence to the philosophy of dependency injection, immutable objects are used throughout the new implementation of AgentSpeak. This is advantageous in that it makes the use of AgentSpeak semantic objects in the standard collection classes safe and reduces the difficulty of their use in multi-threaded architectures[4].

7.1.4 Layered Implementation

Each agent corresponds to a process in the new AgentSpeak implementation. Communication and control of an agent is facilitated by network protocols as discussed section 7.1.5. This facilitates the construction of multi-agent systems via scripting languages that issue commands and requests using the aforementioned protocols. This clearly separates the logic of multi-agent systems from that of a single agent. This is advantageous over threaded solutions as standard operating system facilities can be utilized to manage agents and faults in various components of a multi-agent system are isolated. The cost of taking such an approach is that each agent utilizes more memory and requires its own instance of the Java Virtual Machine. Subsequently, the total number of agents that can be utilized in a given multi-agent system is more restricted compared to a threaded architecture.

7.1.5 Network Controls

By default, communication with individual agents is conducted via simple network protocols, although the network implementation can be replaced with other means. Each agent accepts control messages; can establish debug connections from which it can receive debug information requests; responds to these debug requests; and, communicates bi-directionally with the environment to which it is configured to connect.

[4] "Immutable objects are always thread-safe."
(Goetz, Peierls, Bloch, Bowbeer, Holmes, and Lea [49, Page 46])

Control messages allow the agent to be: (1.) started, (2.) stopped, (3.) stepped, (4.) cycled, and (5.) paused. Starting an agent directs it to continuously execute its reasoning cycle, waiting only to synchronize communications with the environment. Stopping an agent causes the agent to break out of its reasoning cycle and for the agent process to exit. Stepping an agent causes it to execute the next step of its reasoning cycle and return to the paused state waiting for further direction. Cycling an agent causes it to execute an entire reasoning cycle and return to the paused state, again, waiting further instruction. Should the agent be paused mid reasoning cycle, then cycling the agent will cause it to execute the remaining elements in the current reasoning cycle before waiting for user input. Pausing an agent only has meaning in the context of a started agent. Pausing an agent causes it to complete the pending reasoning step and await further instruction. This differs from stopping in that stopping an agent causes it to cease execution and no longer accept further commands.

Debugging sessions are also controlled by simple network messages. Each session requires two communication channels: a control channel that allows the debugging client to request information about the available targets to receive data about and then to subsequently request such data; and a data channel on which the debugging information is returned. Each debugging target is an element of the agent's state that is exposed via the debugging protocol. Such targets include, but are not limited to: the current reasoning step of the agent; its belief set; its intention set; its set of pending events; the selected event, option and intention; and the agent's plans. Once a control connection is established between the agent and the debugging client, the agent establishes the additional data communication channel. In this way, the communication between debugging client and agent is bi-directional. In combination with an appropriate controller, the state of an agent can be monitored as the agent's state changes, per reasoning cycle or per reasoning step.

Communication between an agent and the environment in which it is embedded is, by default, conducted over a simple text-based network protocol much like the control and debug protocols. This allows agents to enter and leave environments arbitrarily. It also provides clear separation of the interaction cycle of the environment and the agents contained therein. This communication is synchronous from the perspective of an agent. An agent requests perception information and blocks waiting for a response. The reasoning cycle continues once the perception data has been received. The raw protocol data is converted into the belief representation through adapter classes. These adapter classes must be developed for each

protocol and belief representation combination. Once the reasoning cycle of the agent completes it must send an action to the environment. Again, the internal action representation is converted into an appropriate protocol message by adapter classes. Given not every reasoning cycle of the agent produces an action to be executed, the no operation action ("no-op") must be sent on such occasions. The agent will block until the execution result of the action is returned from the environment. This interaction cycle between the agents and the environment continues until both the agents and the environment are stopped.

Environments can execute in both a synchronous and asynchronous fashion. When operating synchronously, an environment will provide one set of perceptions and accept a single action from each agent known to the environment at the beginning of that interaction cycle. Alternatively, when operating asynchronously, each perception and action request from an agent will be processed in the order in which they are received. This allows agents with shorter reasoning cycles to interact more quickly with the environment than those with longer cycles. In so doing, such environments facilitate execution of both timing controlled experiments as well as those in which the order of interactions can be arbitrary.

7.2 AgentSpeak(L)

As discussed in section 7.1.1, one of the dialects of AgentSpeak that is supported on the new interpreter is AgentSpeak(L)[77]. In many ways, however, the original presentation was under specified[5]. Subsequent implementations, such as Jason, extend beyond the original presentation to offer the lacking facilities. Some of these underspecified elements are amenable to multiple implementation strategies and depending on the application domain and environmental constraints the appropriate choice of which implementation is most effective differs. Jason handles this situation by offering a number of (user extensible) configuration options. These options control aspects of the interpreter such as whether events should be discarded, returned to the event queue or a request posed to other agents in the multi-agent system whenever the agent has no applicable plans for a given event. The new implementation handles options differently. Rather than offer an extensible configuration file that is backed at run-time by a "settings" object, the configuration file is used to control which Java objects are instantiated by the dependency injection framework. Each such object implements a particular strategy. Thus in the new

[5] As noted previously (see section 2.1.3), this was intentional so as to generalize the systems on which AgentSpeak was based.

implementation, the strategy to handle situations in which the agent has no applicable plans for a given event is implemented via a failure handler interface. Users can then implement this interface and the dependency injection framework will instantiate and inject the implementation into the agent's architecture. This mechanism is also used to instantiate the event selection strategy ($\gamma_{\mathcal{E}}$), option selection strategy ($\gamma_{\mathcal{O}}$), intention selection strategy ($\gamma_{\mathcal{I}}$), the strategy to handle cases where the agent is unable to successfully select an intention to execute and the strategy to handle the failure of an intention to execute.

The new implementation is like Jason in that it offers the ability to annotate beliefs with additional information. Unlike Jason, however, the new implementation does not support belief rules nor internal actions. In place of internal actions, the new implementation offers the ability to utilize custom goal parsers and interfaces to implement arbitrary goals. Currently implemented are goals to: post achieve events, execute actions, assert beliefs, drop events, drop all the pending events, drop intentions, drop all intentions, fail a plan, retract a belief, revise the belief base, test the agent's beliefs and update the agent's beliefs[6]. Notable omissions from this list include inter-agent communication primitives, inter-intention coordination mechanisms and reflection facilities.

This dialect is currently working. To demonstrate this a simple agent program to successfully navigate and satisfy the requirements of the Tileworld test domain has been implemented. However, this agent program itself is relatively simple and the agent remains prone to making poor decisions. Consequently, some work is required to improve the performance of the agent program. This dialect will act as a benchmark against which the extensions proposed in this thesis will be compared. Thus, this agent program must be as robust and performant as is possible to ensure a fair baseline to compare against.

7.3 Monitoring

A dialect that extends the traditional AgentSpeak syntax and semantics with the specification and monitoring of success and failure conditions has also been implemented. The implementation differs from the semantics of chapter 4 in that maintenance conditions and modifiers remain unimplemented. However, the representation of section 3.2 is fully supported by the syntax. This was achieved through

[6] Note that updating a belief base and revising it can be considered distinct logical operations (see [60]).

the embedding of island grammars as per figure 7.3 in which $\lceil^{\xi}$ and $_{\xi}\rfloor$ delimits the effect island grammar and the other delimiters are as before (see figure 7.1).

Rubbish Robot Example

```
1  ⌈6⌈5+!⌈1location(robot,X)1⌋5⌋ : ⌈2location(robot,Y)
2        & adjacent(Y,Z)
3        & not(location(robot,X))2⌋
4    <-
5        ⌈4⌈3move(Y,Z)3⌋4⌋;
6        ⌈4!⌈1location(robot,X)1⌋4⌋
7    <~ ⌈ξ(
8        (⌈1location(robot,X)1⌋):
9          suc(⌈2location(robot,X)2⌋);
10         fail(⌈2false2⌋)
11     )ξ⌋.6⌋
```

Figure 7.3: Island Grammars applied to AgentSpeak Plans with Outcome Conditions

The architecture implemented to support the monitoring of success and failure conditions extends that of the traditional AgentSpeak dialect. Thus, all the ways of customizing the architecture discussed in section 7.2 are applicable to this extended implementation. Beyond these, the new architecture supports user definable monitoring strategies. These strategies define the circumstances in which a particular effect should be monitored (see section 4.2). A number of these strategies have been implemented:

- always monitor every effect (definition 4.2.8);
- monitor the effects of the most recently executed plan (definition 4.2.5);
- monitoring all the effects of all plans on the currently selected intention (definition 4.2.6 and figure 7.4); and,
- never monitoring (definition 4.2.9).

However, a number of strategies are yet to be implemented:

- monitoring on plan completion (definition 4.2.1)

- monitoring post plan completion (definition 4.2.2)
- monitoring all plans on the intention stack of a just completed plan (definition 4.2.3)
- monitoring all plans on the selected intention stack post plan completion (definition 4.2.4)

In addition to the implementation of the listed unrealized monitoring strategies, the implemented strategies are largely untested and the implementation of an agent program for the Tileworld domain using these extensions remains future work.

7.4 Managing Commitment

In order to evaluate the intuitions of chapter 6, a new architecture that further extends the implementation above was developed. It was extended to support the propagation of intention revision based on the dependencies that arise between intentions as a consequence of decision making. A prerequisite for doing so is to ensure that the agent only adopts consistent intentions. To achieve this, an additional reasoning step was added to the agent's reasoning cycle. This step filters the set of applicable plans for handling the selected event by the agent's current intentions. The final selection for handling the selected event is then made from this consistent set.

In order to identify the options that are inconsistent with the agent's existing intentions, a consistency detection mechanism is necessary. Given the representation of goals, success and failure upon which such a consistency checker operates, there are a number of alternative means of testing for consistency and different degrees of consistency that can be guaranteed. This is supported in this architecture in the same manner as multiple monitoring strategies were supported in the architecture upon which this builds.

A number of consistency checking mechanisms have been implemented:

Content Checking : This strategy ensures that the goals within two plans are compatible. Two goals are deemed incompatible whenever: both are achieve goals and one aims to achieve the strong negation of the other; both are test goals and one will only succeed provided it is the string negation of the other; or finally, one is an achieve goal and the other a test goal and the achievement goal achieves the strong negation of the condition of the test goal.

```
package agent.architecture.agentspeak.e;

import agent.component.plans.effective.Plan;

import agent.component.intentions.Intention;

import agent.architecture.agentspeak.SelectedIntention;

/**
 * The selected intention monitoring strategy checks only those plans on the
 * currently selected plan stack.
 *
 * @author Timothy Cleaver
 */
public final class SelectedStrategy implements MonitoringStrategy {
   /**
    * The agents selected intention.
    */
   private final SelectedIntention<?, ?, ?> selectedIntention;

   /**
    * Constructor.
    *
    * @param intention
    *  the agents selected intention.
    */
   public SelectedStrategy(final SelectedIntention<?, ?, ?> intention) {
      selectedIntention = intention;
   }

   /**
    * Test whether to check the effects of the plans in this plan stack.
    *
    * @param <V>
    *  the variable type of the intention
    * @param <D>
    *  the value type of the intention
    * @param intention
    *  the intention we are testing
    * @return
    *  true if the input plan should be checked this cycle, false otherwise.
    */
   public <V, D> boolean monitor(final Intention<?, V, D> intention) {
      return intention.equals(selectedIntention.intention());
   }

   /**
    * Test whether this plans effects should be checked.
    *
    * @param <T>
    *  the concrete type of the effective plan
    * @param <V>
    *  the variable type of the effective plan
    * @param <D>
    *  the value type of the effective plan
    * @param plan
    *  the effective plan we are testing
    * @return
    *  true if the input plan should be checked this cycle, false otherwise.
    */
   public <T extends Plan<T, ?, ?, ?, V, D>, V, D>
      boolean monitor(final T plan) {
      return true;
   }
}
```

Figure 7.4: Example Monitoring Strategy Implementation

Direct Effect Checking : This strategy compares the effects of the involved plans. This strategy ensures that none of the success conditions of one plan are failure conditions of the other. It also ensures that none of the success conditions of one plan are the strong negation of any of the success conditions of the other and conversely that none of the failure conditions of one plan are the strong negation of any of the failure conditions of the other.

Necessary Effect Checking : This strategy poses the same constraints as above but instead considers necessary success and failure conditions. The success and failure conditions of the effects of a given plan are necessary conditions. For each goal in the plan, if all options for its achievement declare a particular success or failure condition then this to is a necessary condition. Thus, this strategy checks not only for current consistency but future consistency and discards options that will necessarily result in an inability to find a consistent option in the future.

Possible Effect Checking : This strategy is much like necessary effect checking except that possible success and failure conditions are checked instead. Any necessary condition is a possible condition. For each goal in a plan, if any options for its achievement declare a particular success or failure condition then this is a possible condition.

Given an appropriate consistency checking mechanism, it is possible to detect the intentions that are involved in a given decision and make these dependencies explicit. Based on these explicit dependencies it is possible to propagate changes in dependent intentions to their dependencies. Section 6.3 discussed a number of strategies for doing so. Of those, dependent non-reconsideration (definition 6.3.4), proportional dependent non-reconsideration (definition 6.3.5) and non-reconsideration have been implemented. The remaining strategies remain future work. Given this architecture is based on that described above for monitoring effects and that architecture is still under active development and testing, the same holds true for the architecture underlying these extensions.

7.5 Summary

This chapter outlined the novel properties of the implementation of AgentSpeak developed as a sanity checking mechanism and experimental platform for the propositions of this thesis. Although these aspects were discussed in relative isolation,

their combination leads to a highly modular and extensible platform. It is hoped it will provide the infrastructure for significant empirical investigations, both syntactically and semantically, towards AgentSpeak like agents in the future. The properties discussed ranged from implementation techniques to design decisions and philosophies that emphasize extensibility over efficiency and simplicity, while attempting to retain all three. Like any implementation, issues and work remains to bring it in line with both the semantics presented in chapters 4 and 6, and that of [10].

Demographic polls show that you have lost credibility across the board. Especially with those 14 year-old Valley girls.

Unknown.

8 Conclusion

THREE primary hypotheses were investigated throughout the course of this thesis. The results subsequent to these investigations will be summarized below. Following this, a number of directions for future work will be outlined. Many of these will constitute direct extensions and completions of the work presented in this thesis. Others will require a greater departure from the methodologies utilized here.

8.1 Summary

Intention has been argued to play a central functional role in the reasoning pertaining to action and behaviour of intelligent agents. Similarly, causal reasoning and the study of the effects of action have long been core to the advancement of intelligent machines. Little work has concentrated on the intersection of these disciplines and the opportunities for reasoning that lie in this space. This thesis begins an exploration into this area by providing an initial investigation into three related hypotheses.

Hypothesis:
The successful monitoring of the effects and validity of intentional behaviour requires mechanisms to: (1.) resolve the conflict for the success or failure of a behaviour given evidence for both; and (2.) balance the time spent monitoring against other mental tasks.

Hypothesis:
It is rational for an agent to revise its beliefs in accordance with the intended outcomes and constraints of its behaviours.

Hypothesis:
The explicit modelling of the dependencies between intentions that arise as a consequence of consistency maintenance during decision making facilitates both the quantification of commitment and mechanisms for intention reconsideration propagation.

The underlying concept that connects these hypotheses is the reasoning opportunities presented when the intended effects and side-effects entailed by intentions are known. In the first instance, the requirement of monitoring for these intentional outcomes is investigated. This investigation was conducted through the extension of the syntax and semantics of an existing and widely studied agent architecture and language called AgentSpeak. A number of issues relating to the resolution of conflicting evidence for both the success and failure of given intentions were uncovered and preliminary solutions proposed. In the second case, the opportunity to utilize the intended effects of intentions on resolving problems in the revision of beliefs was examined. Classical belief revision was utilized in the determination of the rationality of the belief revision operators proposed. It was demonstrated that, although not ideal, the operators produce revisions that satisfied the primary requirements of rationality. The third hypothesis lead to inquiry into the inter-dependencies generated through the maintenance of consistency of intentions in light of their intended effects. These dependencies facilitated the development of a theory to quantify the commitment an agent places in a given intention and rational mechanisms to propagate reconsideration from revised intentions to their dependencies.

The contributions subsequent to the investigations into the first hypothesis was previously presented in a series of papers [21, 23, 22]. These papers introduced the representation, as utilized in the extension to the syntax of the AgentSpeak programming language, to capture the success and failure conditions of behaviours. Included also was the identification of the difficulties of resolving the conflicting evidence that may arise subsequent to the use of the advocated representation. With the development of several strategies to overcome this issue, the challenge of balancing the effort expended in monitoring against other reasoning processes was addressed. Multiple approaches that trade the accuracy of the monitoring process against its complexity to different degrees were outlined. With these difficulties overcome, the semantics of the traditional AgentSpeak architecture was generalized and extended to facilitate the integration of these monitoring strategies. These semantics were then implemented, in a preliminary way, in the manner outlined in chapter 7. This implementation, pending additional work, will be tested using the Tileworld domain as discussed in section 2.1.3. An analysis of the strategies for

both conflict resolution and monitoring efficiency will result.

The contributions central to chapter 5 were previously made public in [19]. Included therein was the formal description of the relationship between intention and belief. Crucial to this formalization was the role changes in belief imposes on the validity of intention. Also included, were the formal definitions of a number of belief revision operators that respect the role of beliefs on the validity of intentions. The proofs of the rationality of the operators defined, according to the traditional interpretation of belief change rationality, were established. Belief revision operators that aim towards the minimization of changes to the intentions, maximization of the success of held intentions and minimization of failed intentions were defined. These respected both the maintenance conditions and effect specifications developed as extensions to AgentSpeak central to the previous contributions.

Two major contributions underlie the work of chapter 6. These contributions are central to [20]. Firstly, techniques to calculate the commitment an agent should hold towards its intentions were proposed. These techniques were based on two observations: that intentions are related through their role in decision making; and that the commitment an agent places in a given intention is fundamentally related to the cost of adopting it. Secondly, a number of decision procedures were proposed to define the circumstances under which it is rational to propagate the reconsideration of a particular intention to the other intentions that are related to it. To demonstrate the plausibility of these mechanisms, they were applied to a running example involving an office robot.

As a means of evaluating the proposed solutions, particularly those pertaining to the first and third hypotheses, a new, and in some ways novel, implementation of the AgentSpeak language was developed. The unique properties of this implementation were discussed.

8.2 Future Directions

Although progress was made and contributions produced, these initial findings lead to significant opportunities for future work. These issues will be recounted in order of the hypothesis to which they pertain.

There are known theoretical issues implied by the representation and semantics developed in the investigation of the first hypothesis. Partial solutions have been developed; however, none are sufficiently advanced as to warrant inclusion in this document. Strategies for the balance of the accuracy of monitoring against its cost need development. While a number of generic approaches were discussed, the

structure supplied by the implementation of intention in the AgentSpeak architecture affords avenues of further research. Furthermore, empirical demonstrations of the advantages of the monitoring advocated remains absent. Thus, one obvious avenue of future work is to improve the reliability and robustness of the prototype implementation, which has shown promise, to a level befitting the object of repeatable experimentation.

There remains a disconnect between how intention and belief relate in the investigation of the second hypothesis and their relationship in BDI logic (see section 2.1.2). Subsequently, the generalization of the approach advocated to a logic of multiple modalities, as per section 2.3.2, seems a fertile source of opportunity.

As discussed in section 5.4 the operators of belief revision presented in this thesis sacrifice the future utility of beliefs in favour of maintaining the agent's current intentions. Empirical evaluation of the rationality of such a prioritization is necessary. Such an evaluation would need to be relative to the axis on which environments may vary (as discussed in chapter 1). Such a study would aim to determine the environmental conditions under which such a prioritization is rational. However, in some domains, such as those that require significant hypothetical reasoning to enable the correct selection and monitoring of behaviour, such a prioritization may be found entirely inappropriate. Under such circumstances, disregarding the informational content of belief in favour of the maintenance of intention when revising beliefs may be sub-optimal. Given that an environment may evolve from situations in which existing behaviour is of primary concern to those in which hypothetical reasoning is paramount, techniques that balance the informational content of beliefs and their use in controlling behaviour are necessary. Further, dynamic policies that can adapt the agents belief revision priorities between maintaining intentions and hypothetical reasoning dependent on environmental factors would represent a significant improvement.

Much of the intuition of the theory of commitment developed with respect to the third hypothesis is hidden through the attribution of numerical quantities. These intuitions would, in all likelihood, be better presented using a qualitative approach. This is exacerbated by the need to provide intuitive numerical interpretations of the quantities defined that are compoundable such that the commitment an agent should hold towards compound intentions can be straightforwardly calculated based on the commitment held in the constituent intentions. A qualitative approach requires no such interpretations. Given that a qualitative approach would reduce to providing a priority ordering over intentions according to the commitment they entail, such an ordering could be utilized in providing additional structure to the revisions of belief as discussed in the examination of the second hypoth-

esis. The integration of the solutions proposed to the second and third hypotheses offers an exciting direction for further exploration.

While a number of reconsideration predicates were defined in the exploration of the third hypothesis, these predicates were simple and in many ways extreme. More sophisticated reconsideration strategies are necessary that take into account the commitment values an agent holds towards the involved intentions. One of the primary factors preventing the development of such strategies is the lack of a technique for estimating the benefit of holding or adopting a particular intention.

Again, an initial prototype of the reasoning encompassed in the discussion of intention revision propagation has been developed. Like that developed for monitoring, indications are positive but the implementation is insufficiently reliable for the elimination of outliers in any data collected utilizing it. This speaks purely to the lack of maturity of the implementation and does not indicate an inherent difficulty in the ability to implement the concepts proposed. One difficulty of the integration of intention revision propagation is the definition/implementation of an appropriate initial intention revision operator to instigate such propagation. Given intention revision itself remains an open question and the sensitivity of the effectiveness of propagation to the revision operator utilized, progress in this direction is difficult. Work continues on stabilizing the implementation, however, and optimism surrounds the resolution of these problems.

The implementation developed as a sanity checking mechanism for the solutions proposed in the investigation of the first and third hypotheses is yet to be applied to a benchmark domain to evaluate the intuitions contained therein. Beyond its direct application to the hypotheses of this thesis, the implementation developed offers a number of independent research trajectories. The embodiment of AgentSpeak in autonomous robotic applications remains a little explored topic. The exploitation of the modularity of the new architecture to test the assumption that the efficiency gained by the use of limited belief languages adequately compensates for the additional burdens of their use compared to more expressive approaches warrants study. Given the philosophy of experimentation enabled by the chosen design of the architecture and syntactic analysers, there are a number of uses to which it may be placed in future.

In conclusion, while notable contributions have been made as a consequence of these investigations, like any endeavour of discovery, more questions have been raised than answers provided.

Bibliography

[1] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. "On the logic of theory change: partial meet contraction and revision functions". In: *Journal of Symbolic Logic* 50 (1985), pp. 510–530.

[2] Carlos E. Alchourrón and David Makinson. "Hierarchies of regulation and their logic". Ed. by Hilpinen. In: *New Studies in Deontic Logic* (1981), pp. 125–148.

[3] Natasha Alechina, Rafael H. Bordini, Jomi F. Hübner, and Brian Logan. "Automating Belief Revision for AgentSpeak". In: *Proceedings of the fourth International Workshop on Declarative Agent Languages and Technologies* (2006).

[4] Davide Ancona, Jomi Fred Hübner, Viviana Mascardi, and Rafael H. Bordini. "Coo-AgentSpeak: Cooperation in AgentSpeak through Plan Exchange". In: *Proceedings of the third International Joint Conference on Autonomous Agents and Multi-Agent Systems*. 2004, pp. 696–705.

[5] G. E. M. Anscombe. *Intention*. Harvard University Press, 1957.

[6] Marc S. Atkin and Paul R. Cohen. "Monitoring Strategies for Embedded Agents: Experiments and Analysis". In: *Journal of Adaptive Behaviour* 4.2 (2 1996), pp. 125–172.

[7] Rafael H. Bordini, Ana L. C. Bazzan, Rafael de Oliviera Jannone, Daniel M. Basso, Rosa Maria Vicari, and Victor R. Lesser. "Agentspeak(XL): Efficient Intention Selection in BDI Agents via Decision-Theoretic Task Scheduling". In: *Proceedings of the first International Joint Conference on Autonomous Agents and Multi-Agent Systems*. 2002, pp. 1294–1302.

[8] Rafael H. Bordini, Michael Fisher, Carmen Pardavila, and Michael Wooldridge. "Model checking AgentSpeak". In: *Proceedings of the second International Joint Conference on Autonomous Agents and Multi-agent Systems*. 2003, pp. 409–416.

[9] Rafael H. Bordini, Jomi Fred Hübner, and Renata Vieira. "Jason and the Golden Fleece of Agent-Oriented Programming". In: *Multi-Agent Programming: Languages, Platforms and Applications*. Ed. by Rafael H. Bordini, Mehdi Dastani, Jürgen Dix, and Amal El Fallah Seghrouchni. Springer, 2005, pp. 3–37.

[10] Rafael H. Bordini, Jomi Fred Hübner, and Michael Wooldridge. *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley, 2007.

[11] Rafael H. Bordini and Álvaro F. Moreira. "Proving BDI properties of agent-oriented programming languages: The Asymmetry thesis principles in AgentSpeak(L)". In: *Annals of Mathematics and Artificial Intelligence* 42 (2004), pp. 197–226.

[12] Rafael H. Bordini and Álvaro F. Moreira. "Proving the Asymmetry Thesis Principles for a BDI Agent-Oriented Programming Language". In: *Electronic Notes in Theoretical Computer Science* 70.5 (2002).

[13] Michael E. Bratman. *Faces of Intention: Selected Essays on Intention and Agency*. Cambridge: Cambridge University Press, 1999.

[14] Michael E. Bratman. *Intention, Plans and Practical Reason*. Harvard University Press, 1987.

[15] Michael E. Bratman. *Structures of Agency: Essays*. Oxford University Press, 2007.

[16] Jan Broersen, Mehdi Dastani, Joris Hulstjin, Zisheng Huang, and Leendert van der Torre. "The BOID Architecture". In: *Proceedings of the fifth International Conference on Autonomous Agents*. 2001, pp. 9–16.

[17] Lawrence Cavedon, Lin Padgham, Anand S. Rao, and Elizabeth Sonenberg. "Revisiting rationality for agents with intentions". In: *Proceedings of the eighth Australian Joint Conference on Artificial intellligence*. Ed. by X. Yao. 1995, pp. 131–138.

[18] Brian Chellas. *Modal Logic, An Introduction*. Cambridge: Cambridge University Press, 1980.

[19] Timothy William Cleaver and Abdul Sattar. "Intention Guided Belief Revision". In: *Proceedings of the twenty second Conference on Artificial Intelligence (AAAI 2007)*. 2007, pp. 36–41.

[20] Timothy William Cleaver and Abdul Sattar. "Quantifying Commitment". In: *PRICAI 2008: Trends in Artificial Intelligence: 10th Pacific Rim International Conference on Artificial Intelligence*. Vol. 5351. Lecture Notes in Computer Science. Springer, 2008, pp. 57–69.

[21] Timothy William Cleaver and Abdul Sattar. "Reasoning about Success and Failure in Intentional Agents". In: *Proceedings of the 2005 Pacific Rim International Workshop on Multi-Agents (PRIMA 2005)*. 2005.

[22] Timothy William Cleaver, Abdul Sattar, and Raihana Ferdous. "User defined monitoring strategies for BDI agent programs". In: *Proceedings of the fifth International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2006)*. ACM Press, 2006, pp. 1055–1057.

[23] Timothy William Cleaver, Abdul Sattar, and Kewen Wang. "Reasoning with the Outcomes of Plan Execution in Intentional Agents". In: *Australian Conference on Artificial Intelligence (Aust-AI 2005)*. 2005, pp. 60–69.

[24] Philip R. Cohen and Hector J. Levesque. "Intention = Choice + Commitment". In: *Proceedings of the sixth National Conference on Artificial Intelligence*. 1987, pp. 410–415.

[25] Philip R. Cohen and Hector J. Levesque. "Intention is choice with commitment". In: *Artificial Intelligence* 42 (1990), pp. 213–261.

[26] Aniruddha Dasgupta and Aditya K. Ghose. "Dealing with Objectives in a Constraint-based Extension to AgentSpeak(L)". In: *Proceedings of the eighth Pacific-Rim International Workshop in Multi-Agents*. 2005.

[27] Giuseppe De Giacomo, Yves Lespérance, and Hector J. Levesque. "ConGolog, A Concurrent Programming Language Based on the Situation Calculus". In: *Artificial Intelligence* 121.1-2 (2000), pp. 109–169.

[28] Giuseppe De Giacomo, Yves Lespérance, Hector J. Levesque, and Sebastian Sardina. "On the semantics of IndiGolog - From theory to implementation". In: *Annals of Mathematics and Artificial Intelligence* 41.2-4 (2004), pp. 259–299.

[29] Giuseppe De Giacomo, Ray Reiter, and Mikhail Soutchanski. "Execution Monitoring of High-Level Robot Programs". In: *Proceedings of the sixth International Conference on Principles of Knowledge Representation and Reasoning*. 1998, pp. 453–465.

[30] Guiseppe De Giacomo, Yves Lespérance, and Hector J. Levesque. "On the Semantics of Deliberations in IndiGolog". In: *Annals of Mathematics and Artificial Intelligence* 41.2-4 (2004), pp. 249–299.

[31] Guiseppe De Giacomo, Yves Lespérance, and Hector J. Levesque. "Reasoning about concurrent execution, prioritized interrupts and exogenous actions in the Situation Calculus". In: *Proceedings of the fifteenth International Joint Conference on Artificial Intelligence*. 1997, pp. 1221–1226.

[32] Frank Dignum, John-Jules Meyer, Roel Wieringa, and R. Kuipr. "A modal approach to intentions, commitments and obligations: intention plus commitment yields obligation". In: *Proceedings of the third International Workshop on Deontic Logic in Computer Science*. Ed. by Mark A. Brown and Jose Carno. 1996, pp. 80–97.

[33] Mark d'Inverno, David Kinny, Michael Luck, and Michael Wooldridge. "A Formal Specification of dMARS". In: *Proceedings of the fourth International Workshop on Intelligent Agents IV, Agent Theories, Architectures and Languages*. Lecture Notes in Computer Science 1365 (1997), pp. 115–176.

[34] Mark d'Inverno, Michael Luck, Michael Georgeff, David Kinny, and Michael Wooldridge. "The dMARS Architecture: A Specification of the Distributed Multi-Agent Reasoning System". In: *Autonomous Agents and Multi-Agent Systems* 9 (2004), pp. 5–53.

[35] Jon Doyle. "A truth maintenance system". In: *Artificial Intelligence* 12.2 (1979), pp. 231–272.

[36] Jon Doyle. "Reason Maintenance and Belief Revision: Foundations versus Coherence Theories". In: *Belief Revision*. Ed. by Peter Gärdenfors. Cambridge University Press, 1992, pp. 29–51.

[37] E. Allen Emerson. "Temporal and Modal Logic". In: *Handbook of Theoretical Computer Science*. Elsevier, 1990, pp. 997–1072.

[38] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about Knowledge*. Cambridge: MIT Press, 1995.

[39] Richard E. Fikes and Nils J. Nilsson. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving". In: *Artificial Intelligence* 2 (1971), pp. 189–208.

[40] Peter Gärdenfors. *Belief Revision*. Cambridge University Press, 1992.

[41] Peter Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. Bradford Books, The MIT Press, 1988.

[42] Peter Gärdenfors and David Makinson. "Revisions of knowledge systems using epistemic entrenchment". In: *Proceedings of the second Conference on Theoretical Aspects of Reasoning about Knowledge*. Ed. by M. Vardi. 1988, pp. 83–95.

[43] Michael Gelfond and Vladimir Lifschitz. "Action Languages". In: *Electronic Transactions on AI* 3.16 (1998).

[44] Michael Georgeff and Amy L. Lansky. "Procedural Knowledge". In: *Proceedings of the Institute of Electrical and Electronic Engineers - Special Issue on Knowledge Representation*. 1986, pp. 1383–1398.

[45] Michael Georgeff and Amy L. Lansky. "Reactive Reasoning and Planning". In: *Proceedings of sixth National Conference on Artificial Intelligence*. 1987, pp. 677–682.

[46] Michael P. Georgeff and Francois F. Ingrand. "Decision-making in an Embedded Reasoning System". In: *Proceedings of the eleventh International Joint Conference on Artificial Intelligence*. 1989, pp. 972–978.

[47] Michael Peter Georgeff and Anand S. Rao. "The Semantics of Intention Maintenance for Rational Agents". In: *Proceedings of the fourteenth International Joint Conference on Artificial Intelligence*. Ed. by C. Mellish. 1995, pp. 704–710.

[48] Malik Ghallab, Dana S. Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, 2004.

[49] Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, and Doug Lea. *Java: Concurrency in Practice*. Addison-Wesley, 2006.

[50] Eric A. Hansen. "Cost-Effective Sensing During Plan Execution". In: *Proceedings of the twelfth National Conference on Artificial Intelligence* (1994), pp. 1029–1035.

[51] Sven Ove Hansson. *A Textbook of Belief Dynamics: Theory Change and Database Updating*. First. Kluwer Academic Publishers, 1999.

[52] Sven Ove Hansson. "Reversing the Levi Indentity". In: *Journal of Philosophical Logic* 22 (1993), pp. 637–669.

[53] John F. Horty and Martha E. Pollack. "Evaluating new options in the context of existing plans". In: *Artificial Intelligence* 127 (2001), pp. 199–220.

[54] Nick Howden, Ralph Rönnquist, Andrew Hodgson, and Andrew Lucas. "Jack Intelligent Agents - Summary of an Agent Infrastructure". In: *Proceedings of the fifth International Conference on Autonomous Agents*. 2001.

[55] Marcus J. Huber. "JAM: A BDI-Theoretic Mobile Agent Architecture". In: *Proceedings of the third International Conference on Autonomous Agents*. 1999, pp. 236–243.

[56] George Edward Hughes and M. J. Cresswell. *A New Introduction to Modal Logic*. New York: Routledge, 1996.

[57] Francois Felix Ingrand, Raja Chatila, Rachid Alami, and Frederick Robert. "PRS: A High Level Supervision and Control Language for Autonomous Mobile Robots". In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 1996.

[58] Mark d'Inverno and Michael Luck. "Engineering AgentSpeak(L): A Formal Computational Model". In: *Journal of Logic and Computation* 8.3 (1998), pp. 1–27.

[59] Nicholas R. Jennings and Michael Wooldridge. "Applications of Intelligent Agents". In: *Agent Technology: Foundations, Applications and Markets*. Ed. by Nicholas R. Jennings and Michael Wooldridge. Springer-Verlag, 1998, pp. 2–38.

[60] Hirofumi Katsuno and Alberto O. Mendelzon. "On the Difference between Updating a Knowledge Base and Revising It". In: *In the Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*. 1991.

[61] David Kinny. *Measuring the Effectiveness of Situated Agents*. Tech. rep. 11. Australian Artificial Intelligence Institute, 1990.

[62] David Kinny and Michael Peter Georgeff. "Commitment and effectiveness of situated agents". In: *Proceedings of the twelfth International Joint Conference on Artificial Intelligence*. 1991, pp. 82–88.

[63] David Kinny and Michael Peter Georgeff. "Experiments on optimal sensing for situated agents". In: *Proceedings of the second Pacific Rim International Conference on Artificial Intelligence*. 1992.

[64] Hector J. Levesque, Raymond Reiter, Yves Lespérance, Fangzhen Lin, and Richard B. Scherl. "GoloG: A Logic Programming Language for Dynamic Domains". In: *Journal of Logic Programming* 19.20 (1994), pp. 1–25.

[65] Daniel Lewis. *Counterfactuals*. Harvard University Press, 1973.

[66] Rodrigo Machado and Rafael H. Bordini. "Running Agentspeak(L) agents on SIM_AGENT". In: *Intelligent Agents VIII - Proceedings of the eighth International Workshop on Agent Theories, Architectures, and Languages (ATAL-2001)*. Ed. by John-Jules C. Meyer and Milind Tambe. 2002, pp. 158,174.

[67] John-Jules Meyer, Wiebe van der Hoek, and B. van Linder. "A Logical Approach to the Dynamics of Commitment". In: *Artificial Intelligence* 113 (1999), pp. 1–40.

[68] Álvaro F. Moreira, Renata Vieira, and Rafael H. Bordini. "Extending the Operational Semantics of a BDI Agent-Oriented Programming Language for Introducing Speech-Act based Communication". In: *Proceedings of the first International Workshop on Declarative Agent Languages and Technologies*. Ed. by J. Leite, A. Omicini, L. Sterling, and P. Torroni. Vol. 2990. Lecture Notes in Computer Science. Springer, 2003, pp. 135–154.

[69] Álvaro F. Moreira, Renata Vieira, Rafael H. Bordini, and Jomi Fred Hübner. "Agent-oriented programming with underlying ontological reasoning". In: *Proceedings of the third International Workshop on Declarative Agent Languages and Technologies*. 2005.

[70] Steve Munroe, Tim Miller, Roxana Belecheanu, Michael Pechoucek, Peter McBurney, and Michael Luck. "Crossing the agent technology chasm: Experiences and challenges in commercial applications of agents". In: 21.3 (2006), pp. 345–392.

[71] Karen L. Myers. "Towards a Framework for Continuous Planning and Execution". In: *Proceedings of the AAAI 1998 Fall Symposium on Distributed, Continual Planning*. AAAI Press, 1998.

[72] Allen Newell. "The knowledge level". In: *Artificial Intelligence* 18 (1982), pp. 87–127.

[73] Simon Parsons, Ola Pettersson, Alessandro Saffiotti, and Michael Wooldridge. "Intention Reconsideration in Theory and Practice". In: *Proceedings of the fourteenth European Conference on Artificial Intelligence*. Ed. by Werner Horn. John Wiley & Sons, 2000.

[74] Gordon D. Plotkin. "The Origins of Structural operational Semantics". In: *Journal of Logic and Algebraic Programming* 60-61 (2004).

[75] Martha E. Pollack. "Overloading Intentions for Efficient Practical Reasoning". In: *Noûs* 25.4 (1991), pp. 513–536.

[76] Martha E. Pollack and Marc Ringuette. "Introducing the Tileworld: experimentally evaluating agent architectures". In: *Proceedings of the eighth National Conference on Artificial Intelligence*. Ed. by Thomas G. Dietterich and William R. Swartout. AAAI Press, 1990, pp. 183–189.

[77] Anand Rao. "Agentspeak(L): BDI Agents Speak out in a Logical Computable Language". In: *Proceedings of the seventh European Workshop on Modeling Autonomous Agents in a Multi-Agent World*. Ed. by Walter Van de Velde and John W. Perrame. Vol. 1038. Lecture Notes in Artificial Intelligence. Springer-Verlag, 1996, pp. 42–55.

[78] Anand S. Rao. *Decision Procedures for Propositional Belief-Desire-Intention Logic*. Tech. rep. 44. Australian Artificial Intelligence Institute, 1993.

[79] Anand S. Rao and Michael P. Georgeff. "An Abstract Architecture for Rational Agents". In: *Proceedings of Knowledge Representation and Reasoning*. 1992, pp. 439–449.

[80] Anand S. Rao and Michael P Georgeff. *Asymmetry Thesis and Side-Effect Problems in Linear-Time and Branching-Time Intention Logic*. Tech. rep. 13. Melbourne, Australia: Australian Artificial Intelligence Institute, 1991.

[81] Anand S. Rao and Michael P. Georgeff. "BDI Agents: From Theory to Practice". In: *Proceedings of the first International Conference on Multi-Agent Systems*. Ed. by Victor Lesser. MIT Press, 1995, pp. 312–319.

[82] Anand S. Rao and Michael P. Georgeff. "Decision Procedures for BDI Logic". In: *Journal of Logic and Computation* 8.3 (1998).

[83] Anand S. Rao and Michael P. Georgeff. *Formal Models and Decision Procedures for Multi-Agent Systems*. Tech. rep. 61. Australian Artificial Intelligence Institute, 1995.

[84] Anand S. Rao and Michael P. Georgeff. "Modeling Rational Agents within a BDI-Architecture". In: *Proceedings of the second International Conference on Principles of Knowledge Representation and Reasoning* (1991), pp. 439–449.

[85] Anand S. Rao and Michael P. Georgeff. "Verification of Agent-Oriented Situated Systems: A Model-Theoretic Approach". In: *AAAI Workshop on Formal Models of Rational Action* (1993), pp. 115–124.

[86] Anand S. Rao and Michael Peter Georgeff. "Deliberation and its role in the formation of Intentions". In: *Proceedings of the seventh Conference on Uncertainty in Artificial Intelligence*. 1991.

[87] Anand S. Rao and Michael Peter Georgeff. *Intentions and Rational Commitment*. Tech. rep. Technical Note 8. Australian Artificial Intelligence Institute, 1993.

[88] Ray Reiter. *Knowledge in Action*. MIT Press, 2001.

[89] Hans Rott. "Two methods of constructing contractions and revisions of knowledge systems". In: 2 (1991), pp. 149–173.

[90] Martijn Schut and Michael Wooldridge. "Intention reconsideration in Complex Environments". In: *Proceedings of the fourth International Conference on Autonomous Agents*. 2000, pp. 209–216.

[91] Martijn Schut and Michael Wooldridge. "Principles of Intention Reconsideration". In: *Proceedings of the fifth International Conference on Autonomous Agents*. ACM Press, 2001, pp. 340–347.

[92] Martijn Schut, Michael Wooldridge, and Simon Parsons. "Reasoning about Intentions in Uncertain Domains". In: *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*. Lecture Notes in Computer Science 2143 (2001), pp. 84–95.

[93] Martijn Schut, Michael Wooldridge, and Simon Parsons. "The theory and practice of intention reconsideration". In: *Journal of Experimental and Theoretical Artificial Intelligence* 16.4 (2004), pp. 261–293.

[94] Karl Krister Segerberg. "The basic dynamic doxastic logic of AGM". In: *The Goldblatt Variations* 1 (1991), pp. 76–107.

[95] Murray Shanahan. *Solving the Frame Problem*. MIT Press, 1997.

[96] Murray Shanahan. "The event calculus explained". In: *Artificial Intelligence Today* (1999), pp. 409–430.

[97] Munindar P. Singh and Nicholas M. Asher. "Towards a formal theory of Intentions". In: *Logic in Artificial Intelligence* 478 (1990), pp. 472–486.

[98] John Thangarajah, Lin Padgham, and Michael Winikoff. "Detecting & Exploiting Positive Goal Interaction in Intelligent Agents". In: *Proceedings of the second International Joint Conference on Autonomous Agents and Multi-Agent Systems*. 2003, pp. 401–408.

[99] John Thangarajah, Lin Padgham, and Michael Winikoff. "Detecting and Avoiding Interference Between Goals in Intelligent Agents". In: *Proceedings of the eighteenth International Joint Conference on Artificial Intelligence*. 2003.

[100] Michael Thielscher. "FLUX: A Logic Programming Method for Reasoning Agents". In: *Theory and Practice of Logic programming* (2004).

[101] Michael Thielscher. "Introduction to the Fluent Calculus". In: *Electrical Transactions on Artificial Intelligence* 2 (1998), pp. 179–192.

[102] Michael Thielscher. "Programming of Reasoning and Planning Agents with FLUX". In: *Proceedings of the eighth International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann, 2002.

[103] Michael Thielscher. "The qualification problem: A solution to the problem of anomalous models". In: *Artificial Intelligence* 131(1–2) (2001), pp. 1–37.

[104] Daniele Turi and Gordon D. Plotkin. "Towards a Mathematical Operational Semantics". In: *Proceedings of the twelfth Annual IEEE Symposium on Logic in Computer Science* (1997), pp. 280–292.

[105] Johan van Benthem. "Correspondence theory". In: *Handbook of Philosophical logic, Volume II: Extensions of classical logic*. D. Reidel publishing Co., 1984.

[106] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*. Vol. 337. Synthese Library. Springer, 2008.

[107] Manuela M. Veloso, Martha E. Pollack, and Michael T. Cox. "Rationale-based monitoring for planning in dynamic environments". In: *Proceedings of the fourth International Conference on Artificial Intelligence Planning Systems*. Ed. by R. Simmons, M. Veloso, and S. Smith. 1998.

[108] Michael Winikoff, Lin Padgham, James Harland, and John Thangarajah. "Declarative and Procedural Goals in Intelligent Agent Systems". In: *Proceedings of the eighth International Conference on Principles of Knowledge Representation and Reasoning*. 2002.

[109] Wayne Wobcke. "On the correctness of PRS Agent Programs". In: *Intelligent Agents VI*. Ed. by Nick Jennings and Yves Lesperance. Vol. 1757. Lecture Notes in Artificial Intelligence. Springer-Verlag, 2000, pp. 42–56.

[110] Michael Wooldridge. *Reasoning About Rational Agents*. MIT Press, 2000.

Index

Glossary

Notation	Description
Λ	A function that defines how the commitment an agent has in a given intention accumulates over time . 142, 143, 148, 150, 151, 157–160, 175
!	Syntax for an Achieve goal/event . 30, 35, 43, 47, 85, 92, 101, 105, 106
AchvGl	The achieve goal rule of an AgentSpeak agents semantics . 43
α	An action . 30, 35, 36, 43, 50, 51, 85, 92
$\mathcal{A}$	The agents scheduled actions . 28, 29, 32, 33, 35, 36, 38, 39, 43, 50, 51
Action	The action execution rule of an AgentSpeak agents semantics . 43
AddBel	The add belief rule of an AgentSpeak agents semantics . 44
AddIM	The intention adoption state of an AgentSpeak agents reasoning cycle . 39, 41, 96
agent	An agent . 30, 38–45, 85, 92, 96–105
A	Modality of all paths. 19–22, 25
($\Box$)	Temporal modality of always. 19, 22
ApplPlans	A function to generate the set of plans applicable to satisfying the agents selected event . 34, 40, 41, 46
ApplPl	The option generation rule of an AgentSpeak agents semantics . 40, 41
ApplPl	The option generation state of an AgentSpeak agents reasoning cycle . 39–41, 96
::=	Assignment . 30, 85, 92, 175
Π	A function that defines the innate cost of an intention . 141–143, 145, 146, 148–151, 157–160, 164–167
BDI	The multi-modal logic of beliefs, desires and intentions . iv, 16–18, 21, 22, 25, 26, 28, 33, 36, 42, 47, 75, 78, 198, 203, 206, 207, 210, 212
β	A belief held by the agent . 30, 35, 44, 63–72, 75–77, 83, 85, 90, 92, 112, 116–119, 121–124, 126, 129, 130, 133
$\mathcal{L}_{\mathcal{B}}$	The language of the logic of belief . 28, 29, 63, 70, 83, 90, 116, 119, 123, 131

Notation	Description
(Bel)	Modality of belief. 16, 18–23, 25, 27, 47, 48, 77
B	The belief relation of an agent that maps propositions to truth values in the context of a belief modality . 17–21, 24, 25, 27, 206
$\perp\!\!\!\perp$	The belief remainder operator that generates the set of beliefs remaining after removing sufficient beliefs to ensure that a given proposition no longer follows . 64, 65, 68, 70, 118–120, 126, 127
BRF	A belief revision function . 31, 33, 36, 107
$\overset{\omega}{\underset{t}{*_{\mathrm{B}}}}$	BDI belief revision operator. 24, 25, 206
$\mathbb{B}$	The set of belief sets . 118, 126
$\mathcal{B}$	The agents beliefs . 27–31, 33–36, 38–41, 43, 44, 46, 47, 51, 63–77, 81–83, 85, 88–90, 92, 100, 101, 104, 105, 107, 112–114, 116–133, 205, 208, 215
$\overset{\mathrm{B}}{\underset{\mathrm{D}}{\mapsto}}$	Function to map belief accessible world to desire accessible worlds. 25, 206
$\overset{\mathrm{B}}{\underset{\mathrm{I}}{\mapsto}}$	Function to map belief accessible worlds to intention accessible worlds. 25, 206
$\leftrightarrow$	Logical Equivalence . 66
π	The body of a plan . 29, 30, 43–47, 84, 85, 91, 92, 97–99, 101, 102, 104–106, 109, 110, 207, 210, 212
body	A function to extract the body of a plan from a plan structure . 34, 35
C	The circumstance of an agent. These are the elements of its mental state that are retained through multiple reasoning cycles . 38–45, 47, 96–105, 109–112
ClrInt	The clear completed intentions rule of an AgentSpeak agents semantics . 45
ClrInt	The clear completed intentions state of an AgentSpeak agents reasoning cycle . 39, 43–45
Θ	A function that defines the commitment an agent places in an intention . 148, 157–160
χ	A conflict . 96, 104, 105, 107, 112–114, 207
Ξ	A function that defines when two behaviors are in conflict . 140, 141, 145, 153, 154

Notation	Description
$\wr$	Conflict Operator . 140–143, 145, 146, 149, 151, 153–156, 161–167, 207, 209, 212
Φ_χ	The conflict resolution strategy . 96, 104, 105, 107, 112–114
Cn	Logical consequence operator . 28, 29, 43, 44, 46, 47, 63, 64, 66, 67, 71, 72, 76, 77, 100, 101, 112–114, 116, 117, 119–124, 129–131
ϖ	The context of a plan. This must follow from the agent's beliefs for the plan to be applicable . 29, 30, 32, 43–47, 84, 85, 91, 92, 97–99, 101, 102, 104–106, 109, 110, 207, 210, 212
$-$	Belief contraction operator . 63, 65–67, 70, 74, 75, 126
Ω	A function that defines the cost of an intention . 141, 142, 145, 146, 149–151, 153, 155, 156, 158–160, 164–167, 175
CTL	Computation Tree Logics . 19, 203
$\mathcal{D}_t \wr \mathcal{I}_t$	The subset of desires that are in conflict with a set of intentions at time t. 153
$\mathcal{D}_t \wr \iota$	The subset of desires that are in conflict with a given intention at time t. 153
DelBel	The remove belief rule of an AgentSpeak agents semantics . 44
δ	A state of affairs the agent deems desirable . 153–156
(Des)	Modality of desire. 16, 18–22, 25, 47, 48
D	The desire relation of an agent that maps propositions to truth values in the context of a desire modality . 17–21, 24, 25, 206, 207
$\overset{\omega}{\underset{t}{*_{\mathtt{D}}}}$	BDI desire revision operator. 24, 25, 207
$\mathcal{D}$	The agents desires . 36, 153–156, 207, 209
dif	Difference . 117, 118, 120–123, 125–128, 130, 131, 211, 214
ξ	An effect . 91, 92, 97, 98, 104, 105, 109–112, 189
$\tau : \varpi \leftarrow \pi.\epsilon$	The structure of a plan with effects . 91, 92
ϵ	The effects of a plan . 91, 92, 96–99, 101–107, 109–112, 207, 208, 210, 212
$\lambda : \langle \mathtt{suc}, \mathtt{fail} \rangle$	The structure of an effect . 112–114
$\bigcap$	The intersection of sets of sets . 65, 70, 118, 120, 126, 127
$\doteq^{\mathcal{I}}_{\trianglelefteq}$	Shorthand for $\mathcal{I} \trianglelefteq^{\mathcal{I}}_{\trianglelefteq} \mathcal{I}' \wedge \mathcal{I}' \trianglelefteq^{\mathcal{I}}_{\trianglelefteq} \mathcal{I}$. 132

Notation	Description
$=_{\mathcal{I}}^{\mathtt{suc}}$	Relation that defines when one belief base is equally successful with respect to the set of intentions $\mathcal{I}$ as another. This is simply a shorthand for $\mathcal{B} \sqsubseteq_{\mathcal{I}}^{\mathtt{suc}} \mathcal{B}' \wedge \mathcal{B}' \sqsubseteq_{\mathcal{I}}^{\mathtt{suc}} \mathcal{B}$ for arbitrary belief bases $\mathcal{B}$ and $\mathcal{B}'$. . 127
ε	An event . 31, 34, 38–41, 96, 107, 213
$\mathcal{E}$	The agents pending events . 28, 29, 31–35, 38–40, 43, 44, 47, 96, 101, 105, 107, 188, 214
$\langle \tau, \iota \rangle$	The structure of an event . 29, 34, 39–41
ExecInt	The intention execution state of an AgentSpeak agents reasoning cycle . 39, 42–44, 96, 100, 101
E	Modality of path existence. 19–21, 25
$+$	Belief expansion operator . 63, 64, 70–72, 77, 118, 126, 211
$\pm$	External belief revision operator where the contraction is executed after the expansion . 118, 120, 122, 126, 127, 131, 134, 211
ExtEv	The AgentSpeak semantic rule to process external events . 41
$\overset{\mathtt{fail}}{\perp}$	The intention remainder operator that generates the set of intentions whose failure conditions follow from the input beliefs . 124, 132
$\frac{\perp}{\mathtt{fail}}$	The intention remainder operator that generates the set of intentions remaining after removing those that have failed as a result of applying the input belief base . 124–130, 132
$\mathcal{B}^{\mathcal{I}}_{\mathtt{fail}}$	The beliefs that are relevant to the failure conditions of the intentions in $\mathcal{I}$. 124, 130
`fail`	Failure . 87–92, 112–114, 123–134, 207, 208, 211
$\perp$	False . 16, 29, 34, 41, 44, 47, 48, 51, 62, 77, 105, 109–114, 119, 125, 126, 132, 133, 141, 145, 153, 154, 161–168, 170, 171
F	An arbitrary filter function . 96–98, 100, 103, 208, 209
first	A function that returns the first element of a sequence . 34–36, 51
F_ϵ	The effect filter function that filters to the effects to monitor in the agents current reasoning cycle . 96, 97, 103
GenEff	The AgentSpeak semantic rule to generate the set of effects to monitor in the current reasoning cycle . 103

Notation	Description
GenEff	The generate the set of outcome conditions to monitor this reasoning cycle state of an AgentSpeak agents reasoning cycle . 96, 102, 103
$F_{\mathcal{I}}$	The intention filter function that filters out intentions whose bodies are empty . 97, 98
$F_{\varkappa}$	The maintenance condition filter function that filters to the effects to monitor in the agents current reasoning cycle . 96, 97, 100
GenMnt	The AgentSpeak semantic rule to generate the set of maintenance conditions to monitor in the current reasoning cycle . 100, 102
GenMnt	The generate the set of maintenance conditions to monitor this reasoning cycle state of an AgentSpeak agents reasoning cycle . 96, 98–100, 102
$\eth$	A goal . 30, 35, 45, 84, 85, 92, 99, 102, 104, 172, 174, 175
$\vec{g}$	A sequence of ground terms. Ground terms contain no variables . 30, 35, 85, 92
head	A function to extract the head of a plan from a plan structure . 34, 35
$\mathcal{I}_t \wr \mathcal{D}_t$	The subset of intentions that are in conflict with a set of desires at time t. 153
$\mathcal{I}_t \wr \mathcal{D}_t$	The subset of intentions that are in conflict with a given desire at time t. 153–156
$\mathcal{I}_t \wr \mathcal{O}_t$	The subset of intentions that are in conflict with the set of options at time t. 141, 143, 151, 161–167
$\mathcal{I}_t \wr \{\omega\}$	The subset of intentions that are in conflict with a given option at time t. 143
$\rightarrow$	Material Implication . 16, 18–23, 25, 27, 48, 77, 81, 82, 118–120
μ	A maintenance condition within a plan . 84, 96, 97, 99–102, 107, 110–112, 173, 174, 212
$\varkappa$	A set of maintenance conditions . 84, 96–102, 104, 105, 107, 109, 110, 209, 210
Init	The initialization rule of an AgentSpeak agents semantics . 39
Init	The initialization state of an AgentSpeak agents reasoning cycle . 39, 96

Notation	Description
ι	An intention adopted by the agent . 29, 31, 34, 38–45, 47, 83, 90, 96–102, 104–107, 109–112, 116, 117, 121–124, 129, 130, 133, 140–143, 148, 149, 153, 157–165, 168–172, 174, 207, 208, 212, 213
$\mathcal{L}_{\mathcal{I}}$	The language of the logic of intention . 28, 29, 83, 90, 116, 123, 131
(Int)	Modality of intention. 16, 18–23, 25, 27, 47, 48
$\trianglelefteq$	A preference relation over intentions . 131–133, 207, 210, 211
$\mathtt{I}$	The intention relation of an agent that maps propositions to truth values in the context of an intention modality . 17–21, 24, 25, 27, 206, 210
$\doteq$	The intention remainder operator that generates the set of intentions remaining after removing those whose maintenance condition does not follow from the input beliefs . 117, 120–122, 124–130, 132, 133, 208, 215
$\overset{\omega}{\underset{t}{*_{\mathtt{I}}}}$	BDI intention revision operator. 24, 25, 210
$\mathcal{I}$	The agents intentions . 27–29, 31–36, 38, 39, 41–45, 47, 51, 82, 83, 87–90, 96–105, 107, 111, 116–118, 120–134, 140–143, 145, 146, 149–151, 153–168, 170–172, 188, 207–212, 214, 215
$\trianglelefteq^{\mathcal{I}}_{\trianglelefteq}$	A preference relation generator function for sets of intentions given a preference relation over individual intentions . 131–133, 207
$\cap$	Set Intersection . 18–21, 25, 27, 64, 66, 67, 70–72, 76, 119, 120, 127, 163, 164, 166, 167
IntEv	The AgentSpeak semantic rule to process internal events . 41
$\mathcal{L}$	A logical language . 28, 29, 63, 70, 77, 83, 90, 116, 119, 123, 131, 205, 210
$\wedge$	Logical Conjunction . 20–23, 25, 27, 40–46, 54, 66, 67, 72, 77, 97, 100, 101, 104, 105, 109, 110, 112, 125–127, 132, 133, 140, 153, 161, 162, 170–172, 207, 208
$\vee$	Logical Disjunction . 23, 27, 36, 47, 54, 69, 72
`while`	Maintenance . 82, 83, 116–118, 120–122
$\tau : \varpi \parallel \varkappa \leftarrow \pi.\epsilon$	The structure of a plan with both maintenance conditions and effects . 97, 98, 101, 104, 105, 109, 110
$\tau : \varpi \parallel \varkappa \leftarrow \pi$	The structure of a plan with maintenance conditions . 84

Notation	Description
$\trianglelefteq^{\mathtt{suc}}_{\sqsubseteq_{\mathcal{I}}}$	Relation that defines when one belief base is more successful with respect to the set of intentions $\mathcal{I}$ and the preference relation over the intentions $\trianglelefteq$ than another . 131–133
$\sim_{\gamma^{\mathcal{I}}_{\mathtt{suc}}}$	Success maximizing partial-meet contraction operator . 131, 134
$\gamma^{\mathcal{I}}_{\mathtt{suc}}$	A selection function that targets the maximum number of successful intentions as a result of belief revision . 126, 127, 131, 134, 211
$>_{\mathtt{suc}}$	Success maximizing operator . 123
$\sqsubseteq^{\mathtt{suc}}_{\mathcal{I}}$	Relation that defines when one belief base is more successful with respect to the set of intentions $\mathcal{I}$ than another . 125, 127–129, 208
$\pm_{\gamma^{\mathcal{I}}_{\mathtt{suc}}}$	Success maximizing external belief revision operator . 126, 127, 131, 134
$\sim_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$	Change minimizing partial-meet contraction operator . 118, 122, 131
$+_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$	Change minimizing belief expansion operator . 118
$\gamma^{\mathcal{I}}_{\mathtt{dif}}$	A selection function that targets the minimal change in intentions as a result of belief revision . 117, 118, 120, 122, 126, 131, 211
$<_{\mathtt{dif}}$	Change minimizing operator . 123
$\sqsubseteq^{\mathtt{dif}}_{\mathcal{I}}$	Relation that defines when one belief base produces less changes in $\mathcal{I}$ than another . 117, 118, 120–122, 125–128, 130
$\pm_{\gamma^{\mathcal{I}}_{\mathtt{dif}}}$	Change minimizing external belief revision operator . 118, 120, 122, 126, 131
$<_{\mathtt{fail}}$	Failure minimizing operator . 123
$\sqsubseteq^{\mathtt{fail}}_{\mathcal{I}}$	Relation that defines when on belief base produces less failed intentions from $\mathcal{I}$ than another . 125, 127, 128, 130
$\sim_{\gamma^{\mathcal{I}}_{\mathtt{fail}}}$	Failure minimizing partial-meet contraction operator . 131, 134
$\gamma^{\mathcal{I}}_{\mathtt{fail}}$	A selection function that targets the minimum number of failed intentions as a result of belief revision . 126, 131, 134, 211
$\pm_{\gamma^{\mathcal{I}}_{\mathtt{fail}}}$	Failure minimizing external belief revision operator . 131, 134
$\trianglelefteq^{\mathtt{fail}}_{\sqsubseteq_{\mathcal{I}}}$	Relation that defines when one belief base produces less damaging failed intentions from $\mathcal{I}$ given $\trianglelefteq$ than another . 132, 133
M	A model . 24, 27

Notation	Description
Φ_{ϵ}	The effect monitoring strategy . 96, 97, 107, 109–112
MonEff	The AgentSpeak semantic rule to update the effects of a monitored plan to remove those that have succeeded . 104, 105
MonEff	The monitor outcome conditions state of an AgentSpeak agents reasoning cycle . 96, 103–105
Φ_{μ}	The validity monitoring strategy . 96, 97, 107, 110–112
MonMnt	The AgentSpeak semantic rule to monitor a single maintenance condition . 100–102
MonMnt	The check maintenance conditions state of an AgentSpeak agents reasoning cycle . 96, 100, 101
λ	The name of an effect . 91, 112–114, 207
$(\Box)$	Modality of necessity. 15–17
$(\circ)$	Temporal modality of next. 19, 23, 25
$\mathcal{O}_t \wr \mathcal{I}_t$	The subset of options that are in conflict with the set of intentions at time t. 140–143, 145, 146, 149, 151, 162, 164, 166, 167
$\mathcal{O}_t \wr \{\iota\}$	The subset of options that are in conflict with a given intention at time t. 143
$(\lceil \mathsf{Bel} \rfloor)$	Modality of only-belief. 24, 25
$(\lceil \mathsf{Des} \rfloor)$	Modality of only-desire. 24, 25
$(\lceil \mathsf{Int} \rfloor)$	Modality of only-intend. 24, 25
ω	An option available to an agent to satisfy a need . 38, 41, 96, 139–141, 143, 209, 213
$\mathcal{O}$	The options available to an agent to achieve an intended ends . 28, 29, 31, 32, 34, 38, 40, 41, 96, 107, 140–143, 145, 146, 149, 151, 161–167, 188, 209, 212, 214
$\sim$	Partial-meet belief contraction operator . 65, 70, 118, 122, 131, 134, 211, 213
$\wp$	A function that extracts the set of paths from a BDI logic semantic structure . 25
ρ	A plan . 30, 36, 41, 46, 47, 51, 85, 92, 97, 100, 104, 105, 109–112, 172–175
$\mathcal{P}$	The agents known plans . 28–34, 38–40, 46, 83, 85, 92, 96, 107
$\tau : \omega \leftarrow \pi$	The structure of a plan . 29, 30, 43, 46, 85
pop	A function to pop an element off a stack . 34, 35

Notation	Description
$(\Diamond)$	Modality of possibility. 15, 16
push	A function to push an element onto a stack . 34, 35
$\mathtt{R}$	A relation . 16, 75, 76
$\mathcal{R}$	The plans relevant to the agents selected event . 31, 32, 34, 38, 40, 41, 46, 96, 107
RelPlans	A function to generate the set of plans relevant to the agents selected event . 34, 40, 46
RelPl	The relevant plan generation rule of an AgentSpeak agents semantics . 40
RelPl	The relevant plan generation state of an AgentSpeak agents reasoning cycle . 39, 40, 96
rest	A function that removes the first element of a sequence and returns the remaining elements of the sequence . 34–36
revise	A function that defines the circumstances in which intention revision is rational . 161–172
$*$	Belief revision operator . 24, 25, 70–72, 74, 75, 77, 82, 206, 207, 210
SelAppl	The option selection rule of an AgentSpeak agents semantics . 41
SelAppl	The option selection state of an AgentSpeak agents reasoning cycle . 39–41, 96
σ_ε	The event the agent has selected to process in the current reasoning cycle . 31, 38–41, 96, 107
$\pm_\gamma$	External belief revision operator where the contraction is executed after the expansion and the contraction is defined using selection function γ . 70
σ_ι	The intention the agent has selected to process in the current reasoning cycle . 31, 38, 42–45, 96, 98, 99, 102, 107, 110, 111
$\mp_\gamma$	Internal belief revision operator where the expansion is executed after the contraction and the contraction is defined using selection function γ . 70
σ_ω	The option the agent has selected to have the selected event . 38, 41, 96
$\sim_\gamma$	Partial-meet belief contraction operator generated by selection function γ . 65, 70

Notation	Description
σ	A selection . 31, 38–45, 96, 98, 99, 102, 107, 110, 111, 141–143, 145, 146, 149, 213
γ	An arbitrary selection function . 28–32, 34, 35, 38, 39, 41–43, 45, 51, 65, 68, 70, 96, 98, 104, 107, 109, 117, 118, 120, 122, 126, 127, 131, 134, 188, 211, 213, 214
$\gamma_{\mathcal{E}}$	A selection function for the event to process during the agents current reasoning cycle . 28, 29, 31, 32, 34, 38, 39, 96, 107, 188
SelEv	The event selection rule of an AgentSpeak agents semantics . 39, 40
SelEv	The event selection state of an AgentSpeak agents reasoning cycle . 39, 40, 42, 43, 45, 96, 102, 103, 105
$\gamma_{\mathcal{I}}$	A selection function for the intention to process during the agents current reasoning cycle . 28, 29, 31, 32, 35, 38, 42, 51, 96, 98, 107, 188
SelInt	The intention selection rule of an AgentSpeak agents semantics . 42, 98
SelInt	The intention selection state of an AgentSpeak agents reasoning cycle . 39–42, 96, 98
$\gamma_{\mathcal{O}}$	A selection function for the option to adopt to satisfy the selected event during the agents current reasoning cycle . 28, 29, 31, 32, 34, 38, 41, 96, 107, 188
γ_{θ}	A selection function for unifiers . 30, 38, 43, 45, 104
$(\Diamond)$	Temporal modality of eventually. 19
$\odot$	A sphere defining the proximity of a set of propositions to another . 76
$\circledcirc$	A set of spheres defining the proximity between all elements of a power set of propositions . 76
S	The set of mental states that an agent may be in . 38, 39, 50, 51, 75, 76, 96
Φ	An arbitrary strategy function . 96, 97, 104, 105, 107, 109–114, 207, 212
$\lhd$	A strict preference relation over intentions . 89, 132, 133, 214
$\lessdot^{\mathcal{I}}_{\lhd}$	A strict preference relation generator function set of intentions given a preference relation over individual intentions . 132, 133
$\sqsubset^{\mathtt{dif}}_{\mathcal{I}}$	Relation that defines when one belief base produces strictly less changes in $\mathcal{I}$ than another . 120

Notation	Description
suc	Success . 87–92, 112–114, 123–134, 207, 208, 211, 215
$\frac{\perp}{\text{suc}}$	The intention remainder operator that generates the set of intentions whose success conditions follow from the input beliefs . 124, 132, 133
$\frac{\perp}{\overline{\text{suc}}}$	The intention remainder operator that generates the set of intentions remaining after removing those that have succeeded as a result of applying the input belief base . 124–130, 132
$\mathcal{B}^{\mathcal{I}}_{\text{suc}}$	The beliefs that are relevant to the success conditions of the intentions in $\mathcal{I}$. 124, 130
T	The temporary store of an agent. These are elements of its mental state that are cleared at the end of each reasoning cycle . 38–45, 96–105, 109–112
$\vec{t}$	A sequence of terms . 30, 35, 85, 92
?	Syntax for a Test goal/event . 30, 35, 43, 44, 85, 92
TestGl	The test goal rule of an AgentSpeak agents semantics . 43, 44
t	A time-point . 24, 25, 140–143, 145, 146, 149–151, 153–156, 158–172, 206, 207, 209, 210, 212
top	A function to access the top element of a stack without removing it . 34, 35
τ	The trigger of a plan . 29, 30, 34, 39–41, 43–47, 84, 85, 91, 92, 97–99, 101, 102, 104–106, 109, 110, 207, 208, 210, 212
$\top$	True . 30, 33–36, 39, 45, 48, 51, 62, 85, 92, 97, 102, 104, 106, 109–114, 125, 126, 132, 133, 140, 141, 145, 153, 154, 161–164, 167–172
θ	A unifier that maps variables to the values they are bound to . 30, 34, 35, 38, 41, 43–46, 104, 214
$\cup$	Set Union . 27, 33–35, 41, 43–45, 47, 64, 66, 67, 71, 99, 101, 102, 104, 105, 118–120, 126, 127
$\dashv$	Unknown . 105, 112–114
$(\mathcal{U})$	Temporal modality of until. 19
update	A function that can update the effects of a plan in light of the agents current beliefs . 98, 104
UpdPostMnt	The AgentSpeak semantic rule to update maintenance conditions post goal execution . 102

Notation	Description
UpdPostMnt	The update maintenance conditions post goal execution state of an AgentSpeak agents reasoning cycle . 96, 102
UpdPreMnt	The AgentSpeak semantic rule to update maintenance conditions pre goal execution . 99
UpdPreMnt	The update maintenance conditions prior to goal execution state of an AgentSpeak agents reasoning cycle . 96, 98, 99
U	A function that defines the utility of a given intention . 50, 51
Ψ	A function that defines the weight of a dependency between intentions . 142, 143, 146, 148–151, 157–161, 164–174
ω	A possible world in a possible worlds semantic structure . 24, 25, 27, 75, 76, 206, 207, 210
$\mathcal{W}$	A set of worlds in a possible worlds semantic structure . 24, 26, 27