Motivations and Goal-Directed Autonomy

Felipe Meneguzzi
Robotics Institute
Carnegie Mellon University 5000 Forbes Ave.
Pittsburgh, Pennsylvania 15213
meneguzzi@cmu.edu

Abstract

As we move towards a richer model of computation based on interactions among autonomous agents in lieu of the more traditional numeric batch processing model, it is necessary to devise techniques and abstractions for the development of such autonomous agents. The ability to generate, reason about, and select the means to achieve explicit goals is critical for autonomous agents, the so called goal-driven autonomy. Reasoning about goals demands a suitable abstraction for meta-level reasoning, so that an agent not only reasons about the actions it needs to take, but also about the viability of (and interaction among) such goals. In this paper we discuss the use of motivations as an abstraction of meta-reasoning for goal-driven autonomous agents. We review recent developments in motivated agent architectures and discuss possible directions for future research.

1 Introduction

"The success of the internet has changed way we think about computing." (Luck 2005) It has also changed the way we work and interact using computers, as exemplified by applications such as social networking, online libraries and collaborative work platforms. We have moved away from a model of computing focused on numeric computations and serial data processing to one where interaction among people and machines is a fundamental application of computing power. With the growing use of computers as extensions of human reasoning capabilities, it is necessary to provide the means to tap into the possibilities of a networked world without overwhelming people with all the information and services available. Intelligent autonomous agents have been proposed as a solution these challenges. Here, such agents are generally understood to be agents capable of operating effectively without supervision to achieve one or more design goals, while adapting their behaviour to cope with a complex, dynamic environment (Jennings 2000).

Most widely used agent languages are based on an agent model that reactively selects *procedural* plans from a library of pre-defined plans without an explicit representation of the agent's ultimate goals (d'Inverno et al. 2004) aiming at achieving quick reactions to changes in the world. However, it has been shown that these agent models are inadequate to allow the design of agents that can adapt to changing circumstances at runtime, with researchers arguing that an agent must be ultimately guided by a set of explicit *declarative* goals (Winikoff et al. 2002; Dastani, van Riemsdijk, and Meyer 2005). When using this approach to goal specification, an agent ceases to operate based on reactions to events in the world, but rather performs goal-driven deliberation, whereby an agent is aware of the goals it is trying to achieve before it selects the means through which these goals are to be pursued. An agent defined in terms of declarative goals is able not only to choose different means to achieve these goals, but also to generate new plans at runtime if it has access to planning algorithms, further improving its *flexibility* by being able to overcome situations not covered by its original set of behaviours.

Since agents are often designed to achieve a variety of goals, some of these goals might be mutually incompatible if they are adopted at the same time. This creates the need for a goal-driven agent to perform some kind of reasoning over the existing goals in order to decide, at each point in time, which goals must be adopted and which goals must be dropped that are no longer relevant. This type of reasoning is often called *meta-reasoning*, which is a critical ability of autonomous agents, as it enables them to explicitly consider goals before committing to their achievement, as well as consider courses of action before executing plans, as opposed to simply reacting to events in the environment.

More generally, it appears that meta-level control is more easily applied to domains in which agent behaviour can be evaluated as being effective through some existing criterion. When there is no such criterion, it is necessary to create a way to evaluate an agent's mental state to guide plan selection. One particular reasoning model that has been studied in considerable depth within artificial intelligence for the simulation of societies is based on the philosophical notion of motivated reasoning (Mele 2003). In the context of meta-reasoning, motivations can provide an elegant solution for modelling domain-independent meta-level control in an agent system. Motivations are regarded by many as *an orientation towards certain classes of goals* (Luck and d'Inverno 1998), and are used in various theories of animal behaviour to explain why animals behave in certain patterns (Balkenius 1993).

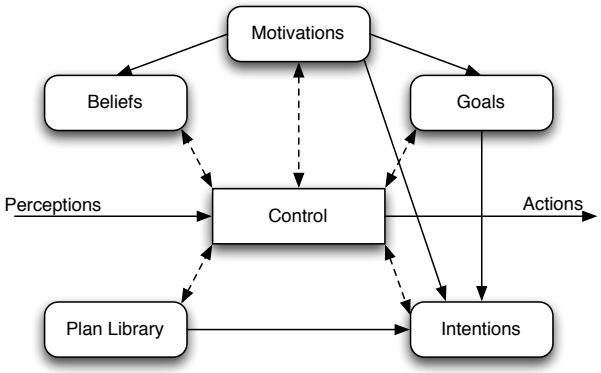

Figure 1: Griffiths and Luck mBDI architecture.

The application of motivations in the control of goal-driven autonomous agent architectures is the subject of this paper. We start by providing a background of motivation-based architectures in Section 2, followed by a discussion of why motivated reasoning is a suitable abstraction for meta-reasoning in Section 3. Further, we exemplify how motivated reasoning can be integrated to a traditional agent architecture in Section 4. Finally, we end the paper with a discussion of possible research directions for motivated reasoning in Section 5.

2 Background

In this section, we review previous efforts in practical motivated agent architectures.

2.1 The Alarms Architecture

The Alarms architecture is one of the first implemented BDI-based motivated architectures. It allows agents to generate goals asynchronously and to focus resources on the accomplishment of important goals (Norman and Long 1995). This process of asynchronous goal generation results in new goals being generated before current ones are accomplished, so that it is possible for an agent to adopt more goals than it can effectively work on at the same time. Alarms uses motivations to allocate processing resources for scheduling and planning towards adopted goals, so as to avoid reaching a specified limit on its processing resources. Thus, the motivated architecture tries to maintain the number of goals the agent pursues simultaneously within an upper bound determined by the limit of an agent processing resources. Without the motivated architecture, when this bound is exceeded, the agent will no longer function effectively.

2.2 Griffiths's mBDI model

(Griffiths and Luck 2003) provide a specific definition of motivation as a tuple $\langle m, i, t, f_i, f_g, f_m \rangle$, where m is the motivation name, i is its current intensity, t is a threshold, f_i is an *intensity update function*, f_g is a *goal generation function*, and f_m is a *mitigation function*. Here, motivations are updated by an agent's beliefs, and in turn, influence the adoption of goals and the selection of intentions, as illustrated in Figure 1, in which solid arrows represent the flow of control and dashed arrows represent the flow of information.

The reasoning cycle for an mBDI agent starts with an agent perceiving the environment, and using this information to update its belief base. In turn, the now updated belief base is used to calculate the intensity of each motivation, according to the intensity update function f_i. After updating motivational intensity values, an agent compares the intensity of each motivation against its threshold for activation, and if this threshold is exceeded, the goal generation function f_g is invoked to generate new goals. Once new goals are generated, all goals are assessed in relation to the motivation that generated them. The goal with the largest intensity value is slated for achievement, resulting in a plan being selected to achieve it, and it is adopted as an intention. Then, the intention that provides the largest motivational reward is selected from among the set of existing intentions, and a step is executed from it. Finally, when an intention finishes executing, the mitigation function f_m is invoked to reduce the intensity of the associated motivation.

2.3 mdBDI: introducing declarative goals

The *mdBDI* model was created in order to extend the mBDI model described in Section 2.2 so that it can apply not only to the notion of goal achievement as plan execution, but also to goal achievement through desired world-states, or goals to be (Meneguzzi and Luck 2007b). The idea here is that motivations must be affected by an agent's perception of world-states, so a particular motivation must only be mitigated when an agent acting to satisfy it perceives that a certain desired world-state holds. This entails that the mitigation function, originally executed as a result of an intention finishing execution must now be associated with the achievement of particular world-states. Thus, the mitigation function in mdBDI must use a mechanism similar to mBDI's intensity update function, so that it takes an agent's beliefs as its inputs, and evaluates whether or not the currently perceived world-state supports the mitigation of a given motivation. Moreover, support for declarative goals leads to the dissociation of mitigation from intention execution. Mitigation can only occur when a certain motivation is driving behaviour, but since the original mBDI model keeps track of active motivations through the intention it adopted to achieve a motivated goal, by dissociating mitigation with plan execution, mdBDI introduces the notion of *active* motivations. An active motivation is a motivation with its intensity level greater than its activation threshold, signalling that a goal to mitigate it must be adopted. When a motivation is active, it can be mitigated, and therefore the mitigation function is used together with the intensity update function to determine motivational intensity.

Unlike mBDI, the mitigation function in mdBDI is no longer associated with plan execution, but rather in the achievement of world-states, so instead of using the original mBDI f_m function, mdBDI uses a *declarative* mitigation function f_{md} containing a mapping between *beliefs* and new motivational intensities. This is illustrated in the diagram of Figure 2.

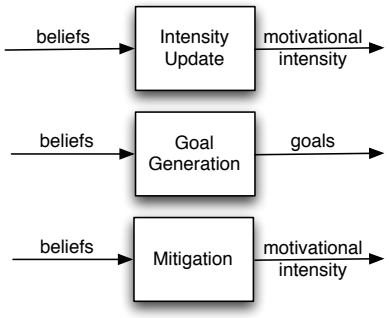

Figure 2: Inputs and outputs of the motivation functions.

Although the modifications to mBDI are small, they have a number of implications to the way in which motivations generate goals and are mitigated. First, due to the new method of mitigating intentions introduced in mdBDI, the last step of selecting an intention implies the capability of an agent to predict the results of a plan after it is executed, otherwise the agent cannot determine how motivationally rewarding a plan is. This prediction of the effects of plans is further developed in Section 4.3. Second, with this new control cycle, plans may execute successfully and still fail to bring about the desired world-state, but a motivation can only be mitigated by achieving a certain desired world-state. For example, if I want to mitigate a motivation to have a video-game console in my home, I may execute my plan to order one from an internet shop successfully, but if the mail service delays the delivery or loses the package, my motivation has not been mitigated by my plan. Moreover, motivations may be mitigated *regardless* of the plan adopted to do so. Using the same example, if a friend gives me the video-game console as a gift after my initial plan failed, my motivation is still mitigated.

This is the motivation model we use later in Section 4 within a concrete agent architecture to underpin the generation of goals.

3 The case for motivations in goal-driven agents

We have seen that most agent languages are defined in terms of a library of plans that contain *action-directed* steps invoked by some kind of trigger-response mechanism, as exemplified by the dMARS architecture (d'Inverno et al. 2004). Thus, if there is a possibility of conflict between any two plans in the plan library, a designer must make sure that these conflicts are handled through extra steps within the plans themselves. As a consequence, the function of meta-reasoning is typically not explicit, but mixed with the action-directed plans in a way that the agent does not handle conflicts not foreseen by the designer. Even in languages that have explicit goal constructs and separate goal-generation rules, such as 3APL (Dastani, van Riemsdijk, and Meyer 2005), goal adoption is a simple process consisting of matching some condition in the agent's perception of the world, adopting a goal and then cease monitoring the reasons for a goal to have been adopted.

While it is certainly possible to develop *ad-hoc* meta-level reasoning using existing traditional agent languages, the absence of distinct processes responsible for meta-level control increases the complexity of an agent's plans while limiting its runtime flexibility, since the function of meta-reasoning must then be accomplished by extra steps within an agent's action-directed plans. Therefore, we believe the development of effective goal-directed autonomous agents can be facilitated by the inclusion of an explicit meta-reasoning capability. Motivations-based meta-reasoning can be used in a variety of control functions, including the selection, prioritisation and abandonment of goals (Cox and Raja 2008); the control of plan selection; and in computationally expensive processes such as planning (Meneguzzi and Luck 2007a; Coddington 2007) (*e.g.* by determining when sufficient effort has been spent in trying to produce new plans). In the specific case of plan-selection, as a plan library grows in size and complexity, the decision to adopt specific courses of action from among multiple possibilities ceases to be as trivial as choosing the first plan that matches a certain condition.

Here, we propose the use of motivations as a suitable generic abstraction for describing a meta-reasoning component that provides an intuitive notion for programmers to describe an agent's subjective notion of valuable world-states to be achieved, or of preferred methods to achieve these world states. Motivation offers a meaningful abstraction that is useful for understanding the meta-reasoning process (as humans), and for modelling the meta-reasoning of other agents. Before we proceed, however, we must clarify the two separate notions of goals in agent systems (Winikoff et al. 2002). In the widely known and used BDI architectures, goals are usually of two types – *procedural* and *declarative* – and in both cases we can apply motivated meta-reasoning. *Procedural* goals require detailed plans and triggers to be described by the designer, hence conflicts between plans must be foreseen, with conflict resolution strategies embedded in each procedural plan as extra *management* steps. For example, if two procedural plans require exclusive access to a particular resource, a designer must make sure that each plan checks whether the other plan is accessing the resource before trying to use it. The function of these extra steps is analogous to meta-reasoning, since they are not action-directed, but rather are evaluating the agent's internal state in querying what other plans are being executed. Thus, in procedural languages specifying meta-reasoning separately from the plans removes the need to replicate such internal *management* steps throughout the plan library, facilitating development. Alternatively, *declarative* goals are defined by desired states to be achieved so that an agent is free to choose which capabilities are employed to achieve goals. Here goal conflict resolution must be done by the underlying interpreter. In declarative-goal architectures, the lack of a goal selection policy means that goals and plans are selected arbitrarily, since in theory the designer does not specify precisely *how* goals are to be achieved.

Thus, motivated reasoning is applicable to the type of BDI architecture we consider in this paper. In particular, the motivational model used to control the architecture must:

- associate motivations with the generation of declarative (*to be*) goals;
- use motivational intensity to select and prioritise intentions adopted to achieve the most rewarding goals; and
- mitigate motivations based on the fulfilment of *declarative* goals.

An agent's motivational intensities change in response to the changes in the state environment and the actions chosen by the agent. Accordingly, motivations-based architectures (Mele 2003) typically provide a function that associates a motivational intensity value with world-states and actions. As far as the model of motivation is concerned, a motivation's intensity serves two purposes: first to determine the relative importance of a motivation compared to others, and second to determine the point at which an agent is sufficiently motivated to generate a goal and actively try to mitigate this intention. In terms of meta-level control, intensity information indicates how important a certain world-state is, which allows an agent to anticipate the motivational effect of certain courses of action before committing to them, allowing it to select plans more effectively.

Translating these functions of motivation to the BDI model requires associating motivational intensity to the belief database, and modifying the reasoning process responsible for committing to intentions so that it uses motivational information. We look at these modifications in Section 4.

4 AgentSpeak(MPL)

We have so far argued for the applicability of motivations for the meta-level control of goal-directed agents. In this section, we take a step further and describe AgentSpeak(MPL): an agent interpreter that employs a meta-level control module that uses motivations as an abstraction for meta-reasoning strategies in order to demonstrate their practical application. This interpreter is an extension of the traditional AgentSpeak(L) language, and implemented and evaluated in our previous work (Meneguzzi and Luck 2007b). AgentSpeak(MPL) was created as a further extension of the AgentSpeak(PL) agent architecture (Meneguzzi and Luck 2007a), which extends traditional AgentSpeak(L) with declarative goals and the ability to generate new plans at runtime. The idea behind adding a meta-reasoning component in AgentSpeak(MPL) is that an agent capable of generating declarative goals must do so pro-actively instead of simply reacting to discrete events in the environment. Generating goals pro-actively requires an agent to have a way of assessing its current situation and anticipating how the environment (or other agents in the environment) will behave, in order to provide a rational justification for the adoption of a goal. Since motivations can be used to associate a measure of importance to world-states (and thus declarative goals), it is possible to use motivational intensity to guide an agent's choice of action when faced with multiple conflicting courses of action.

In traditional AgentSpeak, plans are adopted as a reaction to events in the environment in a direct sense. That is, plans are expressed so that if a certain event `e` happens in a certain world-state, an agent having a plan with a matching triggering event `e` always adopts this plan. Furthermore, since goals in the procedural sense used by AgentSpeak(L) are adopted as part of the execution of plans, an agent does not generate them through deliberation, and they are instead adopted in the process of reacting to some event in the environment. For instance, a plan may be described so that whenever an agent believes that a given block is on a table (*e.g.* `on(block,table)`), a procedure to remove such a block is invoked. This amounts to simple reaction rather than deliberate, future-directed behaviour. This method of *behaviour selection* also fails to properly describe the reasons for goal adoption in a declarative sense. Such shortcomings can be exemplified using the traditional example of a block on a table, in which a declarative goal to remove the block from the table could be described as **`not`** `on(block,table)`. The question here is whether an agent should *always* react to new events and start deliberation immediately even if this agent might be pursuing other, more important, goals. Meta-level control is intended to address this issue by allowing reasoning to take place before action-directed plans are adopted, and our motivation model provides the means with which to compare goals through their relative worth to an agent.

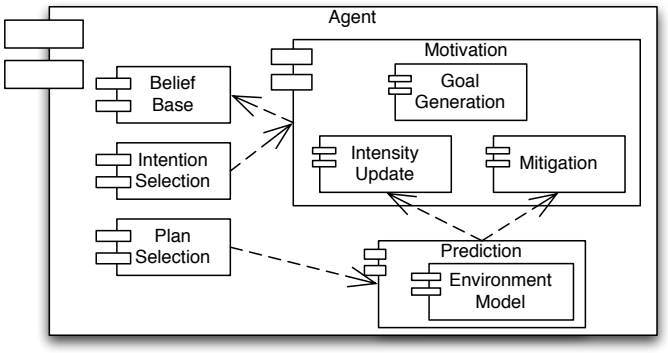

Figure 3: Modules of the motivated AgentSpeak.

AgentSpeak has three key reasoning processes that must be modified in order to accommodate the meta-reasoning model we define in Section 4.1: belief update, plan selection and intention selection. In the following sections, we show how we modify the reasoning processes of AgentSpeak to take the motivations model into account. The components of this architecture are summarised in Figure 3, which shows the motivation module including its three functions and their association with the belief base and intention selection processes, explained in Sections 4.2 and 4.4, as well as the prediction module associated with plan selection, explained in Section 4.3.

4.1 An abstract model of motivation

Following the mdBDI model described in Section 2.3, each motivation is composed of an identifier Id, an intensity value Int, a threshold T, and the name of a concrete function to be used for each of the required abstract functions of our

motivation model, as follows:

$$\langle Id, Int, T, f_i(Beliefs), f_g(Beliefs), f_{md}(Beliefs)\rangle$$

Given the intended meta-level function of our motivational model, we want the computational representation of the motivation to be as simple as possible, limiting the amount of computational resources required in its processing. Thus, we adopt a numeric representation of the intensity value of a motivation in order to simplify motivational processing. Moreover, since the threshold value must be defined in relation to the intensity value, it must also be represented as a numeric value. We have seen in Section 2.3 that each motivation includes functions responsible for controlling the motivational intensity and generating goals when appropriate, namely:

- an intensity update function $f_i(Beliefs)$ responsible for updating inactive motivations based on the currently perceived state of the world;
- a goal generation function $f_g(Beliefs)$ responsible for generating goals when a motivation becomes active as a consequence of its intensity reaching the threshold; and
- a mitigation function $f_{md}(Beliefs)$ responsible for updating active motivations in an analogous way to $f_i(Beliefs)$.

The output of each motivation function must be defined by a mapping process compatible with the purpose of the function, so if we are dealing with the intensity update or mitigation functions, the mapping consists of belief-value correspondences, while if we are dealing with a goal generation function, the mapping is a series of belief-goal associations, since this function aims to generate goals given the perceptions that activated a motivation.

4.2 Motivations and Belief Update

Given that motivation intensity is a function of the perceived world-state, most of the motivation machinery is associated with the agent's belief update function. Each motivation data structure comprises an intensity value, a threshold value, and functions for intensity update, goal generation and mitigation. When an agent receives new perceptions, it updates its belief base which is immediately supplied to the *intensity update function* associated with all motivations affecting this agent. During the update process, if the motivational intensity of any motivation reaches its threshold level, the *goal generation function* is invoked, also inspecting the belief base, and generating a set of goals based on the current world-state. Finally, the belief base is inspected by the mitigation function to determine if any of the motivations triggered previously have been mitigated, in which case motivations are adjusted accordingly.

In practice, the intensity update performed by the mitigation function is equivalent to that of the intensity update function. However, this update can only apply to motivations that had been previously *activated* as a result of their threshold levels being reached, and had generated goals.

4.3 Motivations and Plan Selection

The second modified process in AgentSpeak(L) is plan selection. Since goals are adopted based on a numerically quantified (motivational) importance, it makes sense to use this quantification as a criterion for plan selection. Information regarding the motivation that led an agent to adopt a certain goal can be used in the selection of the best course of action to mitigate this motivation. Since our model provides separate functions for motivation intensity update and mitigation, it is possible to use them along with a world-state prediction model to determine the motivational reward of executing the plans known by the agent. Therefore, we need a motivation-driven plan selection function, which selects the most motivationally rewarding plan from a set of applicable plans.

In order to calculate this quantification, plans are submitted to a prediction module that consists of a *sandbox*[1] copy of the current belief base, as well as a model of the environment (Gong et al. 1997). This prediction module then simulates the execution of the plan using the functions from the motivation module to determine the expected overall decrease in motivational intensity. Once all predictions are collected, the function selects the plan that provides the largest mitigation reward. Alternatively, a designer may specify the declarative outcome of each plan and use this outcome to calculate the expected motivational reward.

4.4 Motivations and Intention Selection

When an agent has committed itself to achieving multiple concurrent goals, their relative priorities may fluctuate as events take place in the environment while the agent carries out plans to achieve these goals. Achieving goals in a timely fashion is crucial in a highly dynamic environment, and an agent has to adapt to changing circumstances with minimum overhead. As we have seen, an AgentSpeak interpreter creates a new intention for every *external* goal adopted by an agent (*i.e.* not subgoals), which contains the plan to achieve this goal, as well as the plans required to achieve any possible subgoal generated in the process of carrying out the initial plan.

Subsequently, the interpreter selects an intention to execute the steps from the plans included in them. In a traditional AgentSpeak interpreter, this selection works on a first-come first-serve basis. In our motivated agent architecture, external goals are generated by the *goal generation function* whenever a threshold intensity is reached in a motivation. Here, using motivation information to prioritise goals can help an agent achieve important goals quickly by postponing the execution of plans to achieve less important goals. Under this assumption, we require a module that evaluates all active intentions and selects the intention associated with the most motivated goal. However, one potential problem with this approach to goal prioritisation is that it can lead to problems in the case of plans that gradually mitigate a motivation as they are executed. In this scenario, this kind of

[1] In computer security and software development, a sandbox is an insulated environment where tests can be performed without affecting the main system.

plan will be constantly preempted by new plans adopted to mitigate higher intensity motivations, possibly leading to it never being finished. This behaviour might be undesirable in certain situations, so it is important to ensure good motivation design to avoid this problem in these situations.

5 Conclusion and Discussion

In this paper we have argued towards the use of motivations as an abstraction for meta-level reasoning in goal-directed agent architectures, enumerating what processes within an agent architecture can be affected by motivations in an intuitive way. We have shown how this abstraction can be easily included in an existing agent architecture based on the AgentSpeak(L) language, demonstrating the practicality of such an approach in AgentSpeak(MPL) (described in Section 4). This agent architecture includes an explicit meta-reasoning module with an accompanying motivation-based specification language, which allows the specification of rules for adopting goals, selecting plans and prioritising currently adopted intentions to satisfy some subjective definition of importance, defined in terms of an agent's motivations. Motivations are specified separately from the agent language, allowing for both the reactive plan adoption mechanism inherent to traditional agent languages and the proactive type of goal adoption enabled by motivated reasoning.

Although the motivation mechanism described in Section 4.1 is a modification of previous work on motivations associated with a PRS-like interpreter, the motivations model is generic enough that it can be used in most current agent interpreters, such as 3APL or JACK. While this mechanism is not as precise as other attempts to quantify rewards for certain behaviours (e.g. decision-theoretic models such as MDPs), it is a simple mechanism that involves minimal overhead in the interpreter reasoning cycle. In addition, specifying meta-level behaviour separately from action-directed behaviour also results in a simpler agent description.

The potential application of motivated reasoning is not limited to the agent processes shown in this paper, and motivation intensities can be used whenever some valuation is needed for an agent to decide between alternatives. Potential directions of future work in motivated reasoning include agents interacting in an environment ruled by norms (Meneguzzi and Luck 2009). Norms provide prescriptive rules of behaviour within an agent society, specifying desirable, undesirable and allowable behaviours in terms of obligations, prohibitions and permissions. In this context, an agent can use motivations to decide on whether to comply with specific norms or not depending on the prospects of mitigating motivations. Where norms have deadlines by which they must be obeyed, motivational intensity can be used to gradually prioritise behaviours needed to comply with a norm as deadlines approach, thus providing an intuitive mechanism for ordering norm-related goals.

Acknowledgement: The author would like to thank Michael Luck and Nir Oren for their invaluable comments and last minute feedback on the paper.

References

[Balkenius 1993] Balkenius, C. 1993. The roots of motivation. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, 513–523.

[Coddington 2007] Coddington, A. 2007. Integrating motivations with planning. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, 1–3. New York, NY, USA: ACM.

[Cox and Raja 2008] Cox, M., and Raja, A. 2008. Metareasoning: A manifesto. In *Proceedings of AAAI 2008 Workshop on Metareasoning: Thinking about Thinking*.

[Dastani, van Riemsdijk, and Meyer 2005] Dastani, M.; van Riemsdijk, M. B.; and Meyer, J.-J. C. 2005. *Multi-Agent Programming: Languages, Platforms and Applications*. Springer-Verlag. chapter 2: Programming Multi-Agent Systems in 3APL, 39–68.

[d'Inverno et al. 2004] d'Inverno, M.; Luck, M.; Georgeff, M.; Kinny, D.; and Wooldridge, M. 2004. The dMARS Architecture: A Specification of the Distributed Multi-Agent Reasoning System. *Autonomous Agents and Multi-Agent Systems* 9(1 - 2):5–53.

[Gong et al. 1997] Gong, L.; Mueller, M.; Prafullchandra, H.; and Schemers, R. 1997. Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java Development Kit 1.2. In *USENIX Symposium on Internet Technologies and Systems*.

[Griffiths and Luck 2003] Griffiths, N., and Luck, M. 2003. Coalition formation through motivation and trust. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 17–24. ACM Press.

[Jennings 2000] Jennings, N. R. 2000. On agent-based software engineering. *Artificial Intelligence* 117(2):277–296.

[Luck and d'Inverno 1998] Luck, M., and d'Inverno, M. 1998. Motivated behavior for goal adoption. In Springer-Verlag., ed., *Selected Papers from the 4th Australian Workshop on Distributed Artificial Intelligence, Multi-Agent Systems*, 58–73.

[Luck 2005] Luck, M. 2005. 50 Facts About Agent-Based Computing, AgentLink III. Available online at `http://bit.ly/bIBYM5`.

[Mele 2003] Mele, A. R. 2003. *Motivation and Agency*. Oxford University Press.

[Meneguzzi and Luck 2007a] Meneguzzi, F., and Luck, M. 2007a. Composing high-level plans for declarative agent programming. In *Proceedings of the Fifth Workshop on Declarative Agent Languages*, 115–130.

[Meneguzzi and Luck 2007b] Meneguzzi, F., and Luck, M. 2007b. Motivations as an abstraction of meta-level reasoning. In Burkhard, H.-D.; Lindemann, G.; Verbrugge, R.; and Varga, L. Z., eds., *Proceedings of the 5th International Central and Eastern European Conference on Multi-Agent Systems*, volume 4696 of *LNAI*. Springer-Verlag. 204–214.

[Meneguzzi and Luck 2009] Meneguzzi, F., and Luck, M. 2009. Norm-based behaviour modification in BDI agents. In *Proceedings of the Eighth International Conference on Autonomous Agents and Multiagent Systems*, 177–184.

[Norman and Long 1995] Norman, T. J., and Long, D. 1995. Alarms: An implementation of motivated agency. In *Intelligent Agents II*, volume 1037 of *LNCS*. Springer-Verlag. 219–234.

[Winikoff et al. 2002] Winikoff, M.; Padgham, L.; Harland, J.; and Thangarajah, J. 2002. Declarative & Procedural Goals in Intelligent Agent Systems. In Fensel, D.; Giunchiglia, F.; McGuinness, D. L.; and Williams, M.-A., eds., *Proceedings of the Eighth International Conference on Principles and Knowledge Representation and Reasoning*, 470–481. Morgan Kaufmann.