Motivations as an abstraction of meta-level reasoning

Felipe Meneguzzi and Michael Luck

Department of Computer Science, King's College London, London WC2R 2LS, UK
{felipe.meneguzzi,michael.luck}@kcl.ac.uk

Abstract. In agent systems, meta-level reasoning is commonly used in enforcing rationality in the choice of goals and actions performed by an agent, ensuring that an agent behaves as effectively and efficiently as possible. Through meta-reasoning an agent is able to explicitly consider goals before committing to them, and consider courses of action before executing plans. In this paper, we argue that although seldom considered, a flexible meta-level reasoning component is a valuable addition to any agent architecture. We describe such a component for use in BDI architectures, underpinned by a model of motivation and a motivation-based description language, and demonstrate its effectiveness empirically.

1 Introduction

In order to act effectively in any complex environment, autonomous agents must have control over their internal state and behaviour [1]. To exercise this control an agent needs some means of reasoning about its internal state, often in a process known as meta-level reasoning (or meta-reasoning). This is higher level reasoning about the reasoning process itself, and in agent systems it is commonly used in enforcing rationality in the choice of goals and actions performed by an agent, ensuring that an agent behaves as effectively and efficiently as possible. Through meta-reasoning an agent is able to explicitly consider goals before committing to them, and consider courses of action before executing plans, in opposition to simply reacting to events in the environment.

Indeed, several efforts towards refining the deliberation process, which can be viewed as meta-level reasoning, have been proposed recently. For example, meta-reasoning can be used to optimise task scheduling using some utility measure [2], and to improve and manage concurrent goal execution by exploiting opportunities and avoiding conflicts [3,4]. Most of these strategies rely on optimising agent behaviour by comparing some intrinsic characteristic of an agent's plans, execution time, or some abstract notion of utility. While improvements brought about by these techniques underline the benefits of meta-reasoning, most existing work does not treat meta-reasoning independently from the more common action-directed reasoning, and we believe that an agent architecture should have a separate such component.

For example, the widely known and used BDI architectures are usually of two types — procedural and declarative — and in both cases, there are advantages of meta-reasoning. Procedural architectures require detailed plans and triggers to be described by the designer, hence conflicts must be foreseen, with conflict resolution strategies embedded in the procedural plans. In procedural languages, specifying meta-reasoning separately from the plans removes the need to replicate internal *management* code

throughout the plan library, facilitating development. Alternatively, declarative architectures are defined by desired states to be achieved and capabilities with which an agent can achieve them, where an interpreter selects capabilities to achieve goals, and conflict resolution must be done by this interpreter. In declarative languages, the lack of some 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.

In this paper we argue that models of *motivated* behaviour [5,6] can provide a valuable abstraction for the specification of meta-reasoning, specifically in the context of the BDI model. In particular, we describe how reasoning rules can be described using a motivational language, allowing the specification of customised meta-level behaviours. Integrating this with the AgentSpeak(L) architecture, we demonstrate empirically that such explicit meta-reasoning can bring significant improvements. The paper is organised as follows: Section 2 gives an overview of recent research on motivations; Section 3 describes an extended agent interpreter augmented with a motivation-based meta-level reasoner; Section 4 reports on the results obtained from experiments performed using our architecture; and, finally, Section 5 compares this work with existing efforts and outlines the main results and future work.

2 Motivations

Understood as the root cause of future-directed behaviour, motivation has been studied by researchers in a variety of areas, such as psychology [7], ethology [8] and philosophy [5]. A psychology-inspired definition of motivation considers it as representing an individual's orientation towards particular classes of goals [7], while a philosophical definition [5] encompasses the concept of varying motivational strength linked to an agent's urgency in relation to its associated goals. Based on multiple sources, Mele [5] posits that motivation is a trait present in animals capable of representing goals and means to goals, both of which may be influenced by motivation. That is, a motivation may influence both the adoption of a goal and the choice of the means to accomplish goals. Motivations vary in strength, dictating the choice of behaviours, so that intentional actions stem from underlying motivations.

From an ethological perspective, motivation is commonly associated with *drives* and *incentives* [8,9]. Simply, drives are internally generated states resulting from the violation of an animal's homeostasis, such as the deprivation of food or the excess of a given hormone. Incentives are externally generated stimuli that increase certain motivations, such as the presence of abundant food causing an animal to feed [8]. Motivations have also been described as giving rise to a continuum of appetitive behaviours (causing agent to need something) leading to consummatory ones (that satisfy this need). Thus some behaviours result in the build up of strength of certain motivations related to appetitive behaviour, and when a motivation has reached a high enough level, consummatory behaviours for the mitigation of this motivation are triggered.

The analysis of the motivational rewards of certain actions can also provide a mechanism to prevent certain undesirable behaviours from occurring simultaneously (also referred to as *lateral inhibition*) [10], as in trying to look at a watch while holding a mug of coffee with the same hand. More generally, if one assumes that the consequences of

any action can be measured as affecting motivations either positively or negatively, then such values can be used to determine which plans an agent can execute simultaneously without incurring detrimental interference among these plans.

The aspect of motivation most commonly sought to be captured by computational architectures is the continuous representation of priorities as a means to determine the *focus of attention* at any given time [11,12,6]. This is important as it allows an agent with limited resources to concentrate its efforts on achieving goals that are relevant to it at specific moments, and to adapt such a concentration of effort to the current reality. Contrasting with the traditional process of goal selection based solely on environmental state, real biological systems often generate different plans of action under the same environment. Here, motivations can be modelled as a mechanism associated with internal *cues* that trigger goal generation in parallel with external factors [9]. Such cues can be seen as trigger conditions that, when activated, cause an agent to consider the adoption of a set of associated goals. They differ from the simple logical preconditions traditionally used in that they result from the dynamics of motivation strength.

3 AgentSpeak-MPL

The BDI model has been the focus of agents research for a significant time [13], and is still ongoing. However, the first instances of complete BDI logics [14] assumed an agent able to foresee all of the future ramifications of its actions as part of the process of deciding which courses of action to take. This assumption was clearly too strong if computationally-bounded BDI architectures were to be constructed. Subsequent agent architectures were designed to avoid unbounded computations by selecting courses of action as a reaction to particular events in the environment under the assumption that they bring about the desired goal. This type of *reactive goal adoption* mechanism [15] is not compatible with a proactive stance, since it forces any future-directed behaviour to be bound to a stream of events for an agent to trigger its goal selection.

This is also true for AgentSpeak(L), in which plans matching events are adopted. For instance, whenever an agent believes that a given block is on a table, a procedure to remove the block may be invoked. This amounts to simple reaction rather than autonomous behaviour, since there is no consideration of any other goals that might have been pursued elsewhere at the time. Furthermore, this method of *behaviour selection* fails to describe the reasons for goal adoption in a declarative sense. The question here is whether the agent should *always* react to new events and start deliberation immediately even if it might be pursuing other more important goals. We believe that in performing meta-level reasoning, an agent can effectively consider its courses of actions without the risk of unbounded computation, and pay heed to any other more important goals. Moreover, by using motivations as a means to evaluate future goals, the assumption of omniscience required for evaluating competing goals is replaced with a preference relation, which is more realistic.

In order to demonstrate the utility of motivations as an abstraction for meta-reasoning, we have extended an AgentSpeak(L) interpreter with a motivation-based meta-level reasoner. This extended interpreter, AgentSpeak-MPL, takes a motivation specification

Algorithm 1 mBDI control cycle.

1: **loop**
2: perceive the environment and update the beliefs;
3: **for all** motivation m **do**
4: apply f_i to m to update i;
5: **end for**
6: **for all** motivation m **do**
7: **if** $i > t$ **then**
8: apply f_g to m to generate new goals;
9: **end if**
10: **end for**
11: select a plan for the most motivated of these new goals and adopt it as an intention;
12: select the most motivationally valuable intention and perform the next step in its plan;
13: on completion of an intention apply f_m to each motivation to reduce its intensity;
14: **end loop**

in addition to a regular AgentSpeak(L) agent specification, and is used by the motivation module to update motivational intensity, generate goals and mitigate motivations.

3.1 A Language of Motivation

An Abstract Motivation Model In order to create the motivation component for our experiments, we adopt the model of Griffiths *et al.* [6] as a base. Here, a motivation is a tuple $< m, i, t, f_i, f_g, f_m >$, 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. The model underpins the mBDI architecture [6], which in turn is based on the PRS/AgentSpeak architecture plus motivations. The reasoning cycle for an mBDI agent is illustrated in Algorithm 1. In brief, it consists of perceiving the environment and using this to update the intensity of each motivation, later generating goals for motivations with intensities that exceed their thresholds. When an agent adopts a new motivated goal, it selects plans to satisfy it, and then the most motivationally valuable intention for execution. When an intention finishes executing, the motivation is mitigated. This model was created for a procedural agent architecture, as is apparent from Steps 11 and 13 of the algorithm, which describes an intention as a plan to be executed, and mitigation of a motivation is equated to the completion of that plan.

So far we have described the abstract machinery that drives motivated control, and it is necessary to associate these abstractions to concrete motivations. Since different individuals can have different sets of motivations, they are affected by their motivations in varying ways, each with its own dynamics to allow evaluation of situations and achievement of goals according to an agent's unique priorities.

In order to allow a designer to describe motivational aspects for each agent, we therefore require a language to describe unique sets of motivations based on the abstract functions and data structures of the mBDI model. In consequence, we have designed a language centred on the three abstract functions of mBDI model: intensity update; goal generation; and mitigation. Concrete versions of these functions are essentially mappings between beliefs and an intensity value in the case of intensity update and

mitigation, or new goals for the goal generation function. These functions are specified for each individual motivation, of which the agent can have several.

At a high level, each motivation is composed of an identifier, an intensity value, a threshold, and the name of a concrete function to be used for each of the required abstract functions of our motivation model, as follows:

$$\langle processBay, I, 10, f_i(Beliefs), f_g(Beliefs), f_m(Beliefs)\rangle$$

Whenever the intensity of a motivation reaches the declared *threshold* as a result of the intensity update function, it is said to be *activated*, and the goal generation function is invoked, after which the mitigation function is invoked to verify if the condition for the motivation to be mitigated is reached. Within the declaration of each concrete function, details of the mapping process are described, so with an intensity update function, the mapping consists of belief-value correspondences, since we are using new percepts to update a motivation intensity, while with a goal generation function, the mapping is a series of belief-goal associations, since this function aims to generate goals given the perceptions which activated a motivation. We consider each of these in detail below.

Intensity Update and Mitigation Functions As we have seen, the functions for updating the intensity of, and mitigating, a motivation need to provide some kind of mapping between perceptual data and an intensity variation. As a result, our *language of motivation* allows the specification of a mapping between belief formulas and an arithmetic expression expressing how the intensity level should be modified as a result of the beliefs being true. Any specific mapping is represented as $log\_expr \rightarrow arithm\_expr$, where log_expr is a logical expression on the beliefs (*e.g.* a(X) & b(Y)), and $arithm\_expr$ is an arithmetic expression (*e.g.* X+2). An example of such a mapping is shown below:

$$f_i(Beliefs) = \begin{cases} over(P, bay1) \wedge batt(10) \rightarrow 2 \\ occupied(agent) \rightarrow -1 \end{cases}$$

Here, the intensity of the motivation to process a packet is increased by 2 whenever the agent believes a new packet has arrived in loading bay 1 (*i.e.* $bay1$) and it has a battery level of 10. It is important to notice that this language deals exclusively with beliefs, both intrinsic ones and those resulting from perception, whereas some motivation models assign values to actions and, by doing so, conform to a procedural view of reasoning. The mitigation function provides a mapping that is syntactically the same as the intensity update function but, according to our model of motivations, is only invoked when an intention is adopted to satisfy its associated motivation.

Goal Generation Aside from mapping beliefs into perceptions, we must also describe the mapping of beliefs into goals. Since goal generation is only invoked when the motivation threshold is exceeded as a result of intensity accumulation, our language allows the specification of additional constraints before a goal is generated, or the unconditional generation of goals through the *true* condition. Similar to intensity update in that mappings start from a logical expression over beliefs, the target of this mapping are new goals to be achieved as a result of the intensity reaching the threshold in the motivation containing this goal generation function. This is illustrated below, where the agent generates an event to sort a packet located over $bay1$ whenever the goal generation function is invoked. Here, the constraint on the left of the mapping exists only to

$\langle processBay1, I, 10, f_i(Beliefs), f_g(Beliefs), f_m(Beliefs)\rangle$, where

$$f_i(Beliefs) = \begin{cases} over(P, bay1) \wedge batt(10) \rightarrow 2 \\ occupied(agent) \rightarrow -1 \end{cases}$$

$$f_g(Beliefs) = \Big\{ over(Packet, bay1) \rightarrow +!sort(Packet)$$

$$f_m(Beliefs) = \Big\{ over(Packet, pigeonHoles) \rightarrow -20$$

Table 1: Example of a set of motivations.

allow the unification of the packet name with the goal to be generated.

$$f_g(Beliefs) = \Big\{ over(Packet, bay1) \rightarrow +!sort(Packet)$$

Example A complete example of a motivation described using our language is shown in Table 1, specifying the motivation to process packets arriving from loading bay 1 from our previous examples. The motivational intensity starts to increase as soon as the agent detects an unsorted packet over $bay1$, until it reaches the the threshold of 10. Once the threshold is reached, the goal generation function adds a goal to sort this packet. Finally, the agent assumes the motivation is mitigated when it perceives the packet to be over the pigeonholes, diminishing the motivational intensity accordingly.

3.2 Integration with AgentSpeak

In our model, motivation intensity is a function of the perceived world state, so 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. These data structures are updated as a result of the agent perception of the world, as illustrated in the activity diagram of Figure 1. When an agent receives new perceptions, it updates its belief base which is immediately inspected by 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 Experiments and Results

In order to evaluate the potential improvements to agent efficiency, we adapted the scenario used by Duff *et al.* [16] to outline the advantage of proactive maintenance goals in agents. This scenario consists of a Mars rover capable of moving about a two-dimensional environment, in which movement consumes energy from the rover's batteries as it moves. Each unit of distance covered by the rover drains one unit of battery

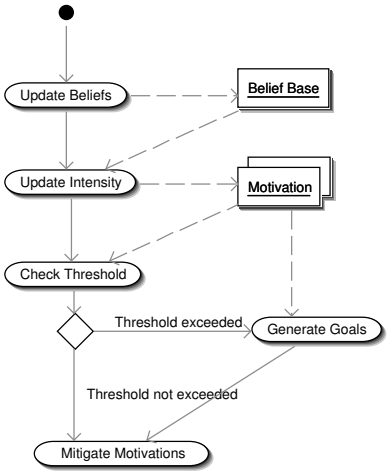


Fig. 1: Activity diagram of a motivated belief update.

energy, and the rover can recharge at the mothership located at the centre of the environment. In the scenario, goals consist of waypoints through which the rover must pass in its exploratory expedition. A varying number of goals was given the agent to assess four test parameters: effective movement, consisting of the distance travelled towards waypoints; supply movement, consisting of the distance travelled towards the mother ship for recharging; wasted movement, consisting of the distance travelled to a waypoint that was wasted due to the need to recharge halfway through getting to a waypoint; and the number of intentions dropped to avoid complete battery discharge.

In this context, a more effective agent travels the least wasted distance, as well as the least supply distance (optimising battery use). Regarding the reasoning process itself, a more rational agent can be seen as one that drops the least amount of goals, since reasoning on adopting and managing ultimately dropped goals is also wasteful.

Our experiments consisted of submitting an increasingly larger number of waypoints, randomly generated for each set varying from 10 to 100 waypoints, to three different agents. The baseline of performance for the experiments was established by an agent with an infinite amount of energy, giving optimal movement to navigate through the entire set of waypoints, since there is no need to move to the mothership. These waypoints were also submitted to a traditional AgentSpeak(L) agent, which must monitor its battery level and recharge it before being too far from the mothership. Its strategy is to navigate to the waypoints in the order they arrive, without prior consideration of battery level. Whenever it is about to move to a position beyond reach of the mothership, it drops the goal to navigate the waypoints and adopts the goal to move to the mothership and recharge. A third agent used AgentSpeak(MPL) , driven by two motivations to navigate and to keep a safe battery level. These motivations behave as a sort of bilateral inhibition mechanism, in which a high intensity for the motivation to have a safe battery level suppresses the intensity of the motivation to navigate. In more detail, whenever it perceives a new waypoint, the motivation to navigate is stimulated, and when it reaches

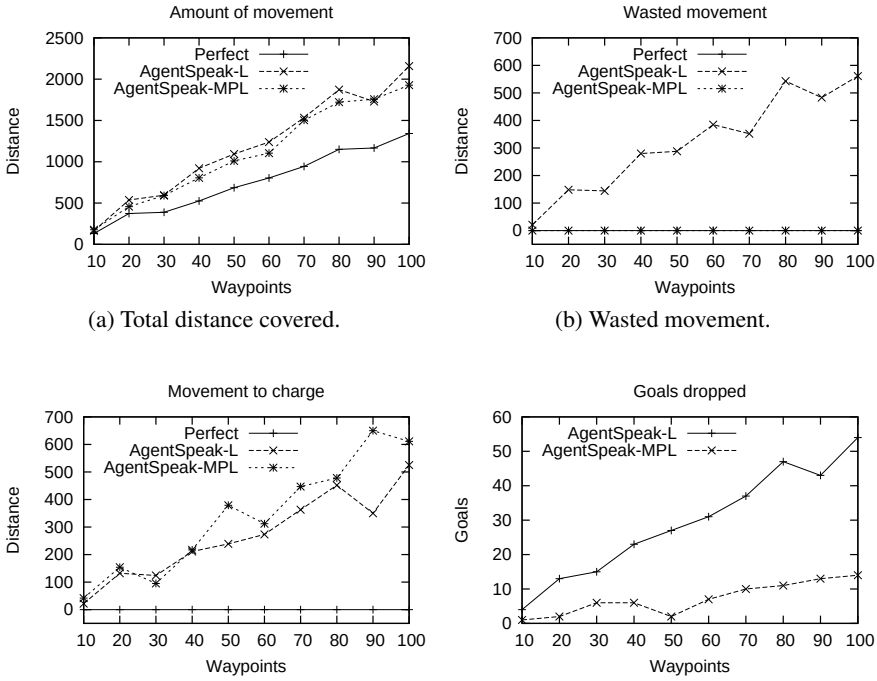
	AgentSpeak(L)	**AgentSpeak-MPL**
# atoms	119	95
# plans	19	10

Table 2: Plan library size comparison.

its threshold, a goal to navigate to this waypoint is generated. The motivation to navigate, however, is suppressed if the battery level is not sufficient to reach that waypoint or if the charge spent doing so will place the agent in a position too far from the mothership. Conversely, the motivation to keep the battery level safe is stimulated by these two battery-related conditions. When the intensity of this motivation reaches its threshold, a goal to move to the mothership and recharge is generated. The result of this is that a goal to navigate to a waypoint should not be generated unless the rover has enough battery to move to it and then to the mothership.

Because the traditional AgentSpeak(L) agent starts executing plans to navigate as it perceives new waypoints, it only detects a critical battery level after having moved some way towards its target position, resulting in a waste of movement actions and a larger total distance covered. On the other hand, the effect of considering the amount of charge before starting to execute a plan to navigate is that no movement is wasted, and thus the total distance covered is smaller, as illustrated in Figure 2a, which compares the total distance covered by each of the agents. In these experiments, the motivated agent had to cover an average 6% less distance than the traditional AgentSpeak(L) agent. The traditional agent had to cover an average of 54% more distance than the baseline agent, compared to 45% for the motivated one, as illustrated in Figure 2b. Figure 2c illustrates the distance covered while moving towards the mothership for charging, where the motivated agent appears to move more than the traditional agent. This is a side-effect of the motivated agent always moving towards the mothership *intentionally*, rather than a side-effect of moving towards a waypoint closer to the mothership, which is corroborated by the smaller amount of total movement shown previously. The last evaluation parameter for this experiment relates to the number of goals dropped as a result of higher-priority goals being pursued. Goals to navigate must be dropped by the rover whenever it has to move back to the mother ship to recharge. Goals that are eventually dropped amount to wasteful deliberation, which rational agents should minimise. Here, the difference between the two approaches is more pronounced, with the motivated agent dropping an average of 75% fewer goals than the traditional agent, as shown in Figure 2d.

Specification size In terms of the size of the agent specification, illustrated in Table 2, traditional AgentSpeak(L) uses a larger plan library to perform the meta-reasoning required to manage concurrent goals, and to allow the prioritisation of the goal to recharge. On the other hand, the motivated agent's plan library, which was derived from the former, requires a significantly smaller number of plans, since the motivation module ensures that goals are generated once per motivation until being mitigated, while also taking care of bilateral coordination.



(a) Total distance covered.

(b) Wasted movement.

(c) Movement towards the mothership.

(d) Number of dropped goals.

Fig. 2: Graphics for the rover experiment

5 Related Work and Conclusions

In this paper we have described an extended agent architecture that includes an explicit meta-reasoning module with an accompanying motivation-based specification language. This follows other recent efforts in adding such reasoning to agent architectures: Raja and Lesser [2] use meta-reasoning to allow the scheduling of existing tasks based on a pre-defined preference relation; Pokahr *et al.* [4] and Thangarajah *et al.* [3], rely on a technique that summarises the effects of plans considered for adoption and analyse positive and negative interactions to avoid conflicts and maximise opportunities; and, finally, Shaw and Bordini [17] use a Petri-Net representation of plans to reason about their utility and potential conflicts. These efforts improve agent efficiency by focusing on specific areas of the reasoning process to optimise. However, such optimisations rely on detailed knowledge of their underlying architectures [4,3], on particular plan representations [17], or on some abstract notion of utility to allow prioritisation [2], and all use a single, static strategy to improve agent efficiency. Our approach differs in that it enables the flexible specification of meta-level behaviour, which, in theory, could be used to specify optimisations similar to all of these efforts.

Given the advantages of having meta-reasoning capabilities in an agent architecture, we have described how motivations can be used as an abstraction mechanism to

define meta-level behaviour. We have created a module of motivations with an associated language based on existing work from the computational motivation literature, and have used it in a prototype based on an extended AgentSpeak interpreter. Our experiments have shown that our model can achieve the same kind of improvement that other reasoning optimisation strategies have achieved, while specifying meta-level behaviour separately from action-directed behaviour also results in a simpler agent description. Even though we used AgentSpeak(L) as our test platform, this approach can be easily employed for other BDI-based agent languages as well. In the future, we intend to perform more experiments using our language and motivation model to determine potential improvements for richer meta-reasoning capabilities.

Acknowledgments: The first author is supported by *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES) of the Brazilian Ministry of Education. Thanks to Andrew Jones for useful discussions regarding the content of this paper.

References

1. Jennings, N.R.: On agent-based software engineering. Artificial Intelligence **117**(2) (2000) 277–296
2. Raja, A., Lesser, V.: Meta-level reasoning in deliberative agents. In: Proc. of the IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology. (2004) 141–147
3. Thangarajah, J., Padgham, L., Winikoff, M.: Detecting & exploiting positive goal interaction in intelligent agents. In: Proc. of the 2nd Int. Joint Conf. on Autonomous Agents and Multiagent Systems. (2003) 401–408
4. Pokahr, A., Braubach, L., Lamersdorf, W.: A goal deliberation strategy for BDI agent systems. In: Proc. of the 3rd German Conf. on Multiagent System Technologies. (2005) 82–93
5. Mele, A.R.: Motivation and Agency. Oxford University Press (2003)
6. Griffiths, N., Luck, M.: Coalition formation through motivation and trust. In: Proc. of the 2nd Int. Joint Conf. on Autonomous Agents and Multiagent Systems. (2003) 17–24
7. Morignot, P., Hayes-Roth, B.: Motivated agents. Tech. report, Stanford University (1996)
8. Balkenius, C.: The roots of motivation. In: Proc. of the 2nd Int. Conf. on Simulation of Adaptive Behavior. (1993) 513–523
9. Munroe, S.J., Luck, M., d'Inverno, M.: Towards motivation-based decisions for worth goals. In: Proc. of the 3rd Int. Central and European Conf. on Multi-Agent Systems. (2003) 17–28
10. Grand, S., Cliff, D.: Creatures: Entertainment software agents with artificial life. Autonomous Agents and Multi-Agent Systems **1**(1) (1998) 39–57
11. Norman, T.J., Long, D.: Alarms: An implementation of motivated agency. In: Intelligent Agents II. Volume 1037 of LNCS. Springer (1995) 219–234
12. Cañamero, D.: Modeling motivations and emotions as a basis for intelligent behavior. In: Proc. of the 1st Int. Conf. on Autonomous agents. (1997) 148–155
13. Georgeff, M., Pell, B., Pollack, M.E., Tambe, M., Wooldridge, M.: The belief-desire-intention model of agency. In Müller, J., Singh, M.P., Rao, A.S., eds.: Intelligent Agents V. Volume 1555 of LNCS. Springer (1999) 1–10
14. Rao, A.S., Georgeff, M.P.: Formal models and decision procedures for multi-agent systems. Tech. Report 61, Australian Artificial Intelligence Institute (1995)
15. Norman, T.J., Long, D.: Goal creation in motivated agents. In: Intelligent Agents. Volume 890 of LNCS. Springer (1994) 277–290
16. Duff, S., Harland, J., Thangarajah, J.: On proactivity and maintenance goals. In: Proc. of the 5th Int. Joint Conf. on Autonomous Agents and Multiagent Systems. (2006) 1033–1040
17. Shaw, P.H., Bordini, R.H.: Towards alternative approaches to reasoning about goals. In: Proc. of the 5th Workshop on Declarative Agent Languages. (2007)