A Classification of Cognitive Agents

Mehdi Dastani (mehdi@cs.uu.nl)
Institute of Information and Computer Sciences
P.O.Box 80.089
3508 TB Utrecht, The Netherlands

Leendert van der Torre (torre@cs.vu.nl)
Department of Artificial Intelligence, Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam, The Netherlands

Abstract

In this paper we discuss a generic component of a cognitive agent architecture that merges beliefs, obligations, intentions and desires into goals. The output of belief, obligation, intention and desire components may conflict and the way the conflicts are resolved determines the type of the agent. For component based cognitive agents, we introduce an alternative classification of agent types based on the order of output generation among components. This ordering determines the type of agents. Given four components, there are 24 distinct total orders and 144 distinct partial orders of output generation. These orders of output generation provide the space of possible types for the suggested component based cognitive agents. Some of these agent types correspond to well-known agent types such as realistic, social, and selfish, but most of them are new characterizing specific types of cognitive agents.

Introduction

Imagine an agent who is obliged to settle his debt, desires to go on holiday, and intends to attend a conference. Suppose that he believes he can only afford to finance one of these activities and decides to pay his checks to settle his debt. Unfortunately, our agent does not earn much money and is in the habit of buying expensive books. Therefore, he runs again into debt after a short while. Despite the fact that he still has the same obligation, desire, and intention and believes that he can only afford to finance one of these activities, he decides this time to attend the conference. Directly after this decision, he hears that the conference is cancelled and he receives a telephone call from his mother telling him that she is willing to pay his checks for this time. The agent is now happy and decides to go on holiday. Our agent has a friend who has the same obligation, desire, and intention, and likewise believes that he can only afford to finance one of these activities. In contrast to our agent, this friend decides to go on holiday. However, he is late with arranging his holiday; all travel agencies are sold out. Therefore, he decides to attend the conference. In a different situation where these two agents are obliged to visit their mothers, desire to go to cinema, and believe they cannot do both simultaneously, the first agent decides to visit his mother while his friend goes to cinema. Yet in another situation where these agents intend to clean up their houses, are obliged to help their friends, and believe they cannot do both, they decide to clean up their houses. Although these agents behave differently, each of them seems to follow a certain behavior pattern under different situations. The first agent seems to be more sensitive to his intentions and obligations than to his desires while the second agent seems to prefer his desires more than his intentions and obligations. Moreover, the first agent seems to be indifferent towards his intentions and obligations while the second agent seems to prefer his intentions above his obligations. These characteristics and principles that govern agent's actions and behavior determine the type of cognitive agents and can be used as the basis for a classification of cognitive agents.

We are motivated by the studies of cognitive agents where the behavior of an agent is defined in terms of rational balance between its mental attitudes [1, 9, 5]. A classification of cognitive agent types specifies possible ways to define the rational balance. Beside the scientific need to study possible definitions of rational balance in a systematic way, a classification of cognitive agent types is important for many applications where it is impossible to specify agent behavior in specific and usually unknown situations. In such applications, it is important to specify the behavior of agents in strategic terms and by means of types of behavior.

In [2] we investigate the design and implementation issues of generic component-based cognitive agents. In the present paper, we propose an alternative classification of cognitive agent types. There has been many formal and informal studies proposing agent types [1, 8, 4]. In these studies, there is a trade-off between the space of possible agent types and their precise and formal definitions. In particular, informal studies provide a rich space of possible types of cognitive agents and ignore their precise definitions, while formal studies provide precise definition of agent types but ignore the richness of the space of possible types. The proposed classification of cognitive agent types in this paper is formal and in terms of a generic component based architecture.

This classification is systematic and provides a large space of possible types for cognitive agents. Some of these agent types such as realistic, social, and selfish are well-known. However, most of these agent types are new and characterize specific types of behavior.

The layout of this paper is as follows. First, we discuss different ways of classifying agent types. Since our classification is based on generic component based agent architecture, we briefly discuss this architecture and explain some of its properties that are relevant for the agent type classification. Possible agent types within this architecture are discussed. An example of a conflict situation is formalized and it is shown how different agent types behave differently in this situation. Finally, we conclude the paper and indicate future research directions.

Classification based on Agent Architecture

Various frameworks with corresponding type classifications for cognitive agents are proposed [9, 5, 3]. Considering different phases in agent oriented software development process such as analysis, design, and implementation phases, most proposed cognitive agent frameworks with corresponding type classifications are provided for the analysis phase. For example, Rao and Georgeff's BDI framework with realism and commitment strategies as agents types [9] have been developed as formal specification tools for the analysis phase. In this framework, the single minded agent type is thought to be the one which maintains its commitments until either it believes it has fulfilled its commitments or it does not believe it can ever fulfill its commitments.

Although these formal tools and concepts are very useful to specify various types of cognitive agents, they are specifically developed for the analysis phase which makes them too abstract for other phases. In fact, to design and to implement various types of cognitive agents, we need to define agent types in terms of tools and concepts available at the design and the implementation phases such that they can be translated into agent architectures and agent implementations. A closer look at the specification formalisms such as Rao and Georgeff's BDICTL formalism shows that the space of theoretically possible cognitive agent types is determined by the expressive power of that formalism. Obviously, other phases of agent development process restrict and narrow down the space of possible agent types since available concepts and tools at those phases should satisfy conditions such as realizability and computability. This implies that each agent architecture allows only a subset of possible agent types that can be specified at the analysis phase. Therefore, it is essential for each agent architecture to indicate which types of agents can be designed in that architecture. The classification of cognitive agent types in this paper is proposed for the design phase and it is thus in terms of agent architecture.

Agent Architecture

In general, agent architectures are defined in terms of knowledge representation (i.e. data) and reasoning mechanism (i.e. control). The agent type classification, which we introduce in the next section, is defined in terms of properties of generic component based architecture called BOID (BOID stands for Beliefs, Obligations, Intention, and Desire). Therefore, we first briefly explain this architecture, which can be seen as a black box with observations as input and intended actions as output. The architecture and the logic of BOID are discussed in more detail elsewhere [2].

A BOID agent observes the environment and reacts to it by means of detectors and effectors, respectively. Each component in the BOID architecture is a process having an input and output behavior. For this reason and to model the input/output behavior of each component, the components are abstracted as a rule-based systems that contains a set of defeasible rules. As these components output mental attitude only for certain inputs, they represent *conditional* mental attitudes. In the BOID architecture two modules are distinguished: the goal generation module and the plan generation module. The goal generation module generates goals based on beliefs, desires, intentions and obligations, and the plan generation module generates sequences of actions based on these goals. In the rest of this paper, we focus only on the goal generation module since the presented classification of the agent types is defined in terms of rational balance between agent's mental attitudes. Possible classification of agent types that can be defined in terms of the plan generation module or in terms of the interaction between the goal or the plan generation modules are out of the scope of this paper.

The BOID architecture differs from the Procedural Reasoning System (PRS) [7], which is developed within the BDI (Beliefs, Desires, and Intention) framework, in several aspects. The first difference is that BOID extends PRS with obligations as an additional component. One reason for this extension is to incorporate elements of the social level, i.e. social commitments, to formalize for example social agents and social rationality. The second difference is related to the conditional nature of mental attitudes in BOID. In fact, each mental attitude is abstracted as a rule-based system containing defeasible rules. This is in contrast with the representation of mental attitudes in PRS which are sets or lists of formula. The third difference is that the BOID components, which represent mental attitudes, are processes having their own control mechanism. Thus, in contrast to the central control mechanism in PRS, in BOID there are two levels of controls. A central control mechanism at the agent level coordinates activities among components. The control mechanism at the component level determines how and which output is generated by each component when it receives input. Finally, the goals in BOID are generated by the interactions between agent's mental attitudes in contrast to the PRS where goals are given beforehand and become selected by the central control mechanism.

As noticed, each component can be abstracted as a rule-based system specified by propositional logical formulas, in the form of defeasible rules represented as $a \hookrightarrow b$. The reading of a rule depends on the component in which it occurs. For example, a rule in the obligation component, represented as $a \overset{O}{\hookrightarrow} b$, should be read as follows: if a is derived as a goal and it is not inconsistent to derive b, then b is obliged to be a goal. The input and the output of components are represented by sets of logical formulas, closed under logical consequence. Following Thomason [10] these are called *extensions*. The logic that specifies extensions is based on prioritized default logic that takes an ordering function ρ as parameter. This function constraints the order of derivation steps for different components and characterizes the type of the agent. We first briefly discuss the BOID conflict resolution mechanism and then explain how the ordering function can be used to define various agent types.

Conflict Resolution Mechanism

In the BOID architecture, goals are generated by a calculation mechanism. The calculation starts with a set of observations Obs, which cannot be overridden, and initial sets of default rules for the other components: B, O, I, D. Moreover, it assumes an ordering function ρ on the rules of the different components. The procedure then determines a sequence of sets of extensions $S_0, S_1, \ldots$. The first element in the sequence is the set of observations: $S_0 = \{Obs\}$. A set of extensions S_{i+1} is calculated from a set of extensions S_i by checking for each extension E in S_i whether there are rules that can extend the extension. There can be none, in which case nothing happens. Otherwise each of the consequents of the applicable rules with highest ρ-value are added to the extension separately, to form distinct extensions in S_{i+1}. The operator $Th(S)$ refers to the logical closure of S, and the syntactic operation $Lit(b)$ extracts the set of literals from a conjunction of literals b. In practice not the whole set of extensions is calculated, but only those that are calculated before the agent runs out of resources.

Definition 1 *A tuple $\Delta = \langle Obs, B, O, I, D, \rho\rangle$ is called a BOID theory. Let L be a propositional logic, and an extension E be a set of L literals (an atom or the negation of an atom). We say that:*

- *a rule $(a \hookrightarrow b)$ is strictly applicable to an extension E, iff $a \in Th(E)$, $b \notin Th(E)$ and $\neg b \notin Th(E)$;*
- $\max(E, \Delta) \subseteq B \cup O \cup I \cup D$ *is the set of rules $(a \hookrightarrow b) \in \max(E, \Delta)$ strictly applicable to E such that there does not exists a $(c \hookrightarrow d) \in B \cup O \cup I \cup D$ strictly applicable to E with $\rho(c \hookrightarrow d) > \rho(a \hookrightarrow b)$;*
- *$E \subseteq L$ is an extension for Δ iff $E \in S_n$ and $S_n = S_{n+1}$ for the procedure in Figure 1.*

$i := 0$; $S_i := \{Obs\}$;
repeat
  $S_{i+1} := \emptyset$;
  for all $E \in S_i$ **do**
    if exists $(a \hookrightarrow b) \in B \cup O \cup I \cup D$ strictly applicable to E **then**
      for all $(a \hookrightarrow b) \in \max(E, \Delta)$ **do**
        $S_{i+1} := S_{i+1} \cup \{\ E \cup Lit(w)\}$;
      end for
    else
      $S_{i+1} := S_{i+1} \cup \{E\}$;
    end if
  end for
  i:=i+1;
until $S_i = S_{i-1}$;

Figure 1: Procedure to calculate extensions

In our model, ρ can assign values to the rules, such that all rules from one component receive either larger or smaller values than the rules from another component. This implies that the rules from one components are applied before the rules from another component can be applied. This is the basis of our idea to define agent types. Of course, in many practical applications ρ must be specified further. For example, an agent may prefer some of his O rules to some of his D rules while conversely preferring some other D rules to some other O rules. However, this does not mean that our basic idea has to be dropped. It just means that the number of components has to be further specified and the ρ function has to be defined accordingly. Each component can thus be subdivided in a number of subcomponents such that the ρ can describe the preference of the rules accordingly. Here we do not further describe this division since it is not important for the general idea of agent type classification that we present in this paper.

The parameter ρ may assign unique values to the rules of all components. In such a case, the BOID calculation scheme can apply in each iteration loop only one rule, which implies that the BOID calculation scheme generates only one extension. However, ρ may also assign identical integers to different rules. In this case, ρ imposes a partial ordering among the rules. For such a ρ, the above BOID calculation scheme can apply more than one rule in each iteration loop, which implies that the BOID calculation scheme may generate a set of extensions. For example, consider a scenario in which an agent believes that he is in a non-smoking area (i.e. $\top \overset{B}{\hookrightarrow} nsa$). He intends to smoke (i.e. $\top \overset{I}{\hookrightarrow} s$), but he intends not to smoke when he is in a non-smoking area (i.e. $nsa \overset{I}{\hookrightarrow} \neg s$). Define ρ as follows:

$$\rho(\top \overset{B}{\hookrightarrow} nsa) > \rho(nsa \overset{I}{\hookrightarrow} \neg s) > \rho(\top \overset{I}{\hookrightarrow} s)$$

For this ρ, the BOID calculation scheme as defined in Definition 1 generates one single extension which is: $\{nsa, \neg s\}$.
Now, suppose ρ is defined as follows:

$$\rho(\top \overset{B}{\hookrightarrow} nsa) > \rho(nsa \overset{I}{\hookrightarrow} \neg s) = \rho(\top \overset{I}{\hookrightarrow} s)$$

This ρ does assign identical integers to the intention rules and the BOID calculation scheme generates the following two extensions: $\{nsa, \neg s\}$ and $\{nsa, s\}$.

Agent Types

Given the presentation of mental attitudes and the BOID calculation scheme, we investigate which type of interactions between mental attitudes can arise within the BOID architecture and how these interactions can be classified. In principle, there are fifteen types of conflicts that can occur between the mentioned four mental attitudes [2]. These conflicts can be solved in different ways. We explain how different ways of resolving conflicts can be modelled by restricting the order of rule application in the BOID calculation scheme. We argue that these restrictions specify different types of the BOID agent and introduce a classification of the types for the BOID agents. Finally, some examples of BOID types and their solutions to one and the same conflict situation is presented.

Conflict resolution and agent types

One of the main tasks of deliberative agents is to solve possible conflicts among their mental attitudes. In principle, there are fifteen different types of conflicts that may arise either within each class or between classes. Dependent on the exact interpretation of these classes, some of the conflict types may be more interesting or important than others. We distinguish two general types of conflicts: internal and external conflicts. *Internal conflicts* are caused within each component while *external conflicts* are caused between them. Internal conflicts can be distinguished into four unary subtypes (B ; O ; I ; D). External conflicts can be distinguished into six binary conflict subtypes (BO ; BI ; BD ; OI; OD ; ID), and four ternary conflict types (BOI; BOD; BID; OID) and one quadruplicate conflict type (BOID). An example of the BOID external conflict type is the following situation: *The agent intends to go to a conference. It is obligatory for the agent not to spend too much money for the conference. In particular, either the agent should pay for a cheap flight ticket and stay in a better hotel, or the agent should pay for an expensive flight ticket and stay in a budget hotel. The agent desires to stay in a better hotel. But, he believes that the secretary has booked an expensive flight ticket for him.* More examples of these conflicts are presented in [2].

A conflict resolution type, which characterizes an agent type, is considered here as an order of overruling. Given four components in the goal generation module of the BOID architecture, there are 24 possible orders of overruling. In this paper, we only consider those orders according to which the belief component overrules any other component. This reduces the number of possible overruling orders to 6. Some examples of conflict resolution with beliefs are as follows. A conflict between a belief and an intention means that an intended action can no longer be executed due to the changing environment. Beliefs therefore overrule the intention, which is retracted. Any derived consequences of this intention are retracted too. Of course, one may allow intentions to overrule beliefs, but this results in unrealistic behavior. Conflicts between beliefs and obligations or desires need to be resolved as well. As observed by Thomason [10], the beliefs must override the desires or otherwise there is wishful thinking. Moreover, a conflict between an intention and an obligation or desire means that you now should or want to do something else than you intended before. Here intentions override the latter because it is exactly this property for which intentions have been introduced: to bring stability [1]. Only in a call for intention reconsideration such conflicts may be resolved otherwise. For example, if I intend to go to cinema but I am obliged to visit my mother, then I go to cinema unless I reconsider my intentions.

Using the order of string letters as the overruling order and thus as representing the agent type, a realistic agent can have any of the following six specific agent types, i.e. BOID, BODI, BDIO, BDOI, BIOD, and BIDO. These specific agent types are not known in the literature and we do not have any name for them. Note that we overloaded the name BOID in this way, because it becomes a specific type of agent as well as the general name for the agent architecture. These six specific agent types, in which beliefs override all other components, can be represented as a constraint on the ρ function resulting in the well-known agent type, called *realistic*.

Definition 2 *Realistic agent type is a constraint on the ρ function formulated as follows:*
$\forall r_b \in B, r_o \in O, r_i \in I, r_d \in D$
$(\rho(r_b) > \rho(r_o) \;\wedge\; \rho(r_b) > \rho(r_i) \;\wedge\; \rho(r_b) > \rho(r_d))$
or simply

$$B \succ O \;\wedge\; B \succ I \;\wedge\; B \succ D$$

Now that we have a specific ρ function that characterizes realistic BOID types, we indicate how the extension is calculated. Following definition 1, a realistic BOID agent starts with the observations and calculates belief extensions by iteratively applying belief rules. When no belief rule is applicable anymore, then either the O, the I, or the D component is chosen from which one applicable rule is selected and applied. When a rule from a chosen component is applied successfully, the belief component is attended again and belief rules are applied. If there is no rule from the chosen component applicable, then another component is chosen again. If there is no rule from any of the components applicable, then the process terminates – a fixed point is reached – and extensions are calculated.

Other agent types can be specified as constraints on the ρ function as well. Since we consider in this paper only realistic agent types, we limit ourselves to those agent types that are subtypes of realistic agent types. Some of well-known agent types can now be represented as follows.

BIDO, BOID, and BIOD are called *stable*, because intentions overrule desires, i.e.

$$B \succ O \ \wedge \ B \succ I \ \wedge \ B \succ D \ \wedge \ I \succ D$$

BDIO, BIDO, and BDOI are called *selfish*, because desires overrule intentions, i.e.

$$B \succ O \ \wedge \ B \succ I \ \wedge \ B \succ D \ \wedge \ D \succ O$$

BOID, BIOD, and BODI are called *social*, because obligations overrule desires, i.e.

$$B \succ O \ \wedge \ B \succ I \ \wedge \ B \succ D \wedge \ O \succ D$$

The six specific realistic agent types mentioned earlier are subtypes of these three well-known more general realistic agent types. Other agent types, for which we do not have any name, are still possible. The relation between these and other realistic agent types forms a lattice illustrated in Figure 2. The level in this hierarchy indicates the generality of agent types. The bottom of this lattice is the realistic agent type that is characterized by the least number of constraints on the ρ function. Each higher layer adds additional constraints resulting in more specific agent types. At the second level, the stable, social, and selfish agent types result, and at the fourth level the mentioned six specific and unknown agent types (BIDO, BIOD, BDIO, BDOI, BOID, and BODI) result. The top of this lattice is the falsum, which indicates that adding any additional constraint to the ρ function results in an inconsistent ordering.

Example

In this section, we illustrate how conflicts between mental attitudes can be solved within the BOID architecture by giving an example that describes the following mental attitudes: *If I go to Washington DC (Go2DC), then I believe that there are no cheap rooms (ChRm) close to the conference site (Close2ConfSite). If I go to Washington DC, then I am obliged to take a cheap room. If I go to Washington DC, then I desire to stay close to the conference site. I intend to go to Washington DC.* This example can be represented by the following rules:

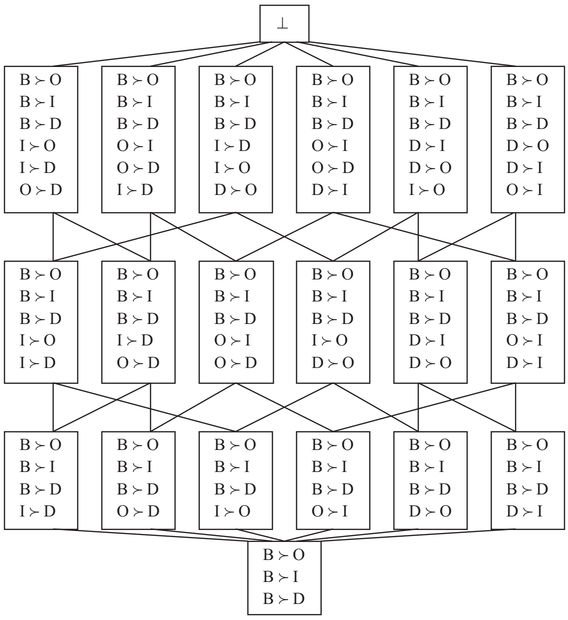

Figure 2: The lattice structure of agent types.

$$\rho = 5 \quad (Go2DC \wedge ChRm) \overset{\mathrm{B}}{\hookrightarrow} \neg Close2ConfSite$$
$$\rho = 4 \quad (Go2DC \wedge Close2ConfSite) \overset{\mathrm{B}}{\hookrightarrow} \neg ChRm$$
$$\rho = 3 \quad Go2DC \overset{\mathrm{D}}{\hookrightarrow} Close2ConfSite$$
$$\rho = 2 \quad Go2DC \overset{\mathrm{O}}{\hookrightarrow} ChRm$$
$$\rho = 1 \quad \top \overset{\mathrm{I}}{\hookrightarrow} Go2DC$$

Lets examine a specific type of social agent, i.e. BIOD. Let the input of the agent be empty. Then, following the extension calculation mechanism, we first derive all beliefs and intentions, resulting in the following extension:

{Go2DC}

Because it is a social agent (i.e. the fourth rule has a higher priority than the fifth rule), the obligation rule is applied first. This results in the following intermediate extension:

{Go2DC, ChRm}

This extension is fed back into the B component where it triggers the first belief rule (i.e. the first rule), because the second belief rules is not applicable as we already have ChRm. This produces the following final extension:

{Go2DC,ChRm,¬Close2ConfSite}

This extension denotes the situation in which the agent has decided to go to Washington DC and takes a cheap room not close to the conference site, which is indeed social behavior.

However, if we exchange the priority of the fourth and the fifth rules the agent becomes a selfish agent 'BIDO'. Then, the *D*-rule would be applied before any obligation rule is applied, resulting in the following final extension:

{Go2DC,¬ChRm, Close2ConfSite}

Sending the results back to the belief component does not make any difference here. This extension denotes the situation in which an agent has decided to go to Washington DC and takes an expensive room close to the conference site, which is indeed selfish behavior.

Concluding Remarks

We have briefly discussed the generic component based BOID architecture that is developed for cognitive agents. Each component in the BOID architecture represents a mental attitudes of the agent. The output of components may conflict. Some of the conflicts that may arise among BOID's components are presented. In the BOID architecture the conflicts are resolved by the order of output generation from different components. We have shown that the order of output generation determines the type of an agent. In general, the order of output generation can be used to identify different types of agents. We have shown that these conflict resolution mechanisms provide some well-known agent types and an interesting set of unknown agent types. In particular, we have shown that for a realistic agent beliefs are generated before obligations, intentions or desires; for a stable agent intentions are generated before desires; and for selfish agents desires are generated before intentions.

We believe that the way the BOID components are updated depends also on the type of the agent. The integration of updating various components have the highest priority in our research agenda. Another issue which in on our future research agenda is the incorporation of agent types derived from plan generation module and its interaction with goal generation modules. In the BOID architecture, the plan generation module influences the computation of extensions and therefore may play an important role in agent type classification. For example, when a generated extension cannot be transformed into a sequence of actions, another extension should be selected. The exact choice for a new extension should depends on the type of agent as well.

References

[1] M. E. Bratman. *Intention, plans, and practical reason.* Harvard University Press, Cambridge Mass, 1987.

[2] J. Broersen, M. Dastani, Z. Huang, J. Hulstijn and L. van der Torre. The BOID Architecture: Conflicts between beliefs, obligations, intentions, and desires. Proceedings of Fifth International Conference on Autonomous Agents (AA'01), "9–16",ACM Press (2001)

[3] R. A. Brooks. A robust layered control system for a mobile robot. IEEE J. Robotics Auromat., RA-2(7):14–23, Apr. 1986.

[4] C. Castelfranchi. Prescribed Mental Attitudes in Goal-Adoption and Norm-Adoption. In AI and Law, Special Issue on Agents and Norms, 7, 1999, 37-50.

[5] P. Cohen and H. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.

[6] M. Gelfond and T. Cao Son. *Reasoning with Prioritized Defaults.* Proceedings of Logic Programming and Knowledge Representation 1997, 164-223, Port Jeerson, New York, October 1997.

[7] M. P. Georgeff and A. L. Lansky. *Reactive reasoning and planning.* In Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), pages 677–682, 1987.

[8] A. Rao and M. Georgeff. *An abstract architecture for rational agents.* In Proceedings of the KR92, 1992.

[9] A. Rao and M. Georgeff. BDI agents: From theory to practice. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS'95)*, 1995.

[10] R. Thomason. Desires and defaults. In *Proceedings of the KR'2000.* Morgan Kaufmann, 2000.