Cognitive Science Quarterly (2000) ***1****, ..-..*

Goal Generation in the BOID Architecture

Jan Broersen[a], Mehdi Dastani[b],

Joris Hulstijn[a], Leendert van der Torre[a]

[a] *Vrije Universiteit, Amsterdam*
[b] *Utrecht University, Utrecht*

In this paper we consider goal generation in cognitive agent architectures. We show how goal generation can be described in terms of interaction between mental attitudes biased by agent types such as realistic, social, selfish and stable. We introduce a generic BOID architecture and agent type specific BOID architectures, in which goals are generated from conditional beliefs, obligations, intentions and desires. We implement the BOID architectures by relating conditional mental attitudes to components, goal generation to extension construction, and agent types to constraints on priority functions.

Keywords: Agent architecture, Goal generation, Reiter's default logic

Introduction

Reasoning about mental attitudes such as beliefs, desires, goals, plans, intentions and obligations has been discussed in practical reasoning in philosophy (von Wright, 1983; Bratman, 1987) and its formalization in qualitative decision making has been discussed in artificial intelligence (Thomason, 2000). Each of these approaches proposes an agent architecture based on a set of primitive concepts in terms of which the behavior of cognitive agents is explained. Classic examples of agent architectures are based on goals and knowledge (Newell, 1982; Laird et al., 1987). More recent agent architectures are based on mental attitudes such as beliefs, desires and intentions (Bratman, Israel & Pollack, 1988; Cohen & Levesque, 1990; Rao & Georgeff, 1991) as well as obligations and norms (Castelfranchi, Dignum, Jonker & Treur, 1999; Dignum, Morley, Sonenberg & Cavedon, 2000; Boella & Lesmo, to appear).

We are interested in the relation between goals on the one hand and desires, intentions and obligations on the other hand. In the classic approaches, desires, intentions and obligations are not incorporated. Goals are primitive

Received 02.05.02

and not derived from more basic attitudes. In the approaches based on beliefs, desires and intentions, either the goals are used interchangeably with desires, or goals and desires are completely separated. Goals are not first class citizens. Thomason (2000) observes that the relation between goals and desires becomes crucial when goal generation is considered. His logical framework has inspired us to propose a cognitive agent architecture, which includes an explicit agent goal generation component. The architecture is called BOID, because goals are generated from the interaction between beliefs, obligations, intentions and desires.

In the BOID architecture we introduce generic and agent type specific goal generation components. The type of the agent such as realistic, social, selfish and stable plays an important role in the interaction between beliefs, obligations, intentions and desires. For example, consider an agent that is obliged to work whereas it desires to swim at the same time. It generates either the goal to go to the office or the goal to go to the beach. Which of these goals will be generated depends on the agent type. A social agent goes to the office and a selfish agent goes to the beach. Agent types can be compared in generality, which leads to an agent type lattice. A computational model for goal generation is proposed based on a prioritized default logic (Reiter,1980). This is inspired by Horty (1994), who showed how motivational attitudes can be formalized in default logic and Thomason (2000), who combined rules for beliefs and desires in one system and who formalized a set of goals generated from mental attitudes as a so-called extension. The BOID architecture adds intentions and obligations, and it implements agent types as constraints on priority functions.

In this paper we focus on goal generation and its relation to agent types. Our analysis remains at the level of interactions between mental attitudes. We do not consider the behavior of individual attitudes; they are considered to be primitive. Moreover, we mainly consider the conceptual model with its implementation, and we leave formal analysis for other papers (Broersen, Dastani & van der Torre, 2001b). Other aspects of practical reasoning, such as scheduling time slots or resources, or coordination of actions with other agents, are topics of further research. In (Broersen, Dastani, Huang, Hulstijn & van der Torre, 2001a) the architecture has been implemented in Prolog, and it has been tested against a list of benchmark examples representing different types of conflicts between mental attitudes.

The layout of this paper is as follows. First we introduce the conceptual decomposition of goal generation in terms of mental attitudes. Thereafter we introduce the corresponding computational BOID architecture and specialized BOID architectures for specific agent types. Finally we consider consequences of our goal generation approach for planning and updating mental attitudes.

Goal generation as interaction of mental attitudes

Our view of the agent's deliberation process is visualized in Figure 1. Observations trigger a goal generation stage. Candidate goal sets are generated based on primitive mental attitudes such as beliefs, obligations, intentions and desires. This results in many different sets of mutually consistent goals. The goal selection stage imposes an ordering on the sets of generated candidate goal sets, such that the best one can be selected for planning. This goal set is then passed on to the planning stage to construct a plan to achieve the goal set. We start with our conceptual view on goal generation, the role of agent types and conditional mental attitudes.

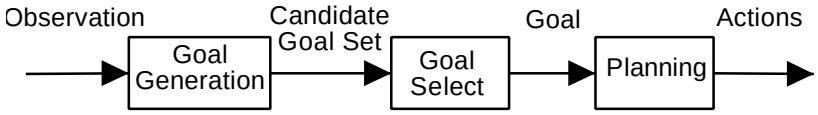

Figure 1: Cycle of the deliberation process

Goal generation

Thomason (2000) proposed that goals are the result of the interaction between beliefs and desires. We extend the interaction with obligations and intentions, as illustrated by the following examples. Each example describes an agent with possibly conflicting motivational attitudes. First, consider the agent that is obliged to work, but desires to swim. If it is at work, then it desires to drink coffee, and if it is swimming at the beach, then it is obliged to wear a bathing suit. Although an agent may have conflicting desires and obligations, it cannot be at work and on the beach at the same time. Now it has to choose one of the two candidate goal sets: being at work with coffee or being at the beach in a bathing suit. In the following example an agent has to find out which actions or plans it can execute to reach the goal, and it has to take the side-effects of its actions into account. The agent desires to be on the beach. If it quits its job and drives to the beach, then it will be on the beach. If it does not have a job, then it will be poor. If it is poor, then it desires to work. The only desire and thus the only goal is to be on the beach. The only way to reach this goal is to quit the job, but the side effect of this action is that it will be poor. In that case it does not want to be on the beach, but it wants to work.

Conflicts within a single attitude have been studied in the literature by for example van Fraassen (1973). However, conflicts between different attitudes have received less attention. A mental attitude conflicts with another mental attitude if both cannot be used to generate a consistent set of goals. The agent thus has to choose which of the two attitudes it should use at the

expense of the other one. For such a choice, we say that the former overrides the latter. The overriding mechanism is the method we use to solve conflicts and generate goal sets. As observed by Thomason (2000), overriding desires by beliefs reflects that the agent does not suffer from wishful thinking. A similar argument applies to obligations and intentions. For example, the overriding of intentions by beliefs reflects that an intended action can no longer be executed due to a change of the environment. Such an intention should not generate a goal. Also overriding among other attitudes can be considered. Overriding obligations or desires by intentions reflects that agents do not reconsider their intentions, which brings stability (Bratman, 1987). Only during a call for intention reconsideration such conflicts may be resolved otherwise. For example, consider an agent that intends to go to the cinema, and that is obliged to visit its mother. It goes to the cinema unless it reconsiders its intentions.

The process of generating goals from different mental attitudes is biased by the type of agents. Agent types are an idea developed in BDI logics to distinguish, classify and compare agent behaviors (Cohen & Levesque, 1990; Rao & Georgeff, 1991). We apply this notion of agent types to goal generation, where agent types correspond to ways to resolve conflicts. Agent types for goal generation are given intuitive names. If the agent's beliefs override its obligations, intentions or desires, then we say that this agent is realistic. If its intentions override its desires and obligations, then we say that the agent is stable. Moreover, if its desires override its obligations, then the agent is called selfish and if its obligations override its desires, then the agent is called social. These agent types can also be combined. For example, in stable social agents intentions override obligations or desires, and obligations override desires. A useful notion here is that some agent types are more specific than other agent types. Of course not all agent types are comparable. For example, the realistic agent type is not more specific than the social agent type, nor vice versa. Goal generation of more specific agents is more predictable than goal generation of general ones.

Conditional mental attitudes

There are two issues discussed in philosophical logic that we believe to be relevant for goal generation. First, goal generation is based on four primitive mental concepts: belief, obligation, intention and desire. We have chosen these four attitudes, because overriding orders between them correspond to intuitive and useful agent types. Philosophical studies of practical reasoning (von Wright, 1983; Bratman, 1987) and formalizations of these attitudes in qualitative decision making show that for decision making these concepts can be further specified. They distinguish between knowledge and defaults, prohibitions and permissions, commitments and plans or wishes and wants, respectively. The four mental attitudes are usually read as follows. Beliefs

are informational attitudes, which model what the world is expected to be like. Intentions are previously generated goals or plans (Bratman et al., 1988). Desires model internal motivations. Obligations model external motivations, such as norms. We do not exclude the possibility that other, more specialized mental attitudes are needed for goal generation. For example, in some cases a distinction may be made between strong and weak intentions, where strong intentions override desires and obligations, whereas desires and obligations override weak intentions. Such additional attitudes must always be motivated by a particular order of overriding, which corresponds to an agent type.

Second, we consider these four mental attitudes to be conditional and context-dependent. Existing models of mental attitudes in cognitive agents such as BDI (Rao & Georgeff, 1991) and 3APL (Hindriks, de Boer, van der Hoek & Meyer, 1999) are not conditional. But in philosophical logic, in particular in deontic logic (Hansson, 1969), it is argued that mental attitudes are conditional by nature. Conditional approaches are often contrasted with modal logics and their possible world semantics. One of the problems of reasoning with conditionals in modal logic is whether a conditional must be represented by $a ? Ox$ or by $O(a ? x)$, where O stands for a modal operator such as obligation. This has traditionally been discussed in the context of so-called contrary-to-duty reasoning, see (van der Torre & Tan, 1999) for a survey.

We interpret the four basic mental attitudes as follows. Beliefs represent the information of an agent about the current state of the world. All observations are turned into beliefs. Beliefs behave as a kind of filter on desires and obligations: only desires and obligations consistent with the beliefs are turned into a goal.

Obligations represent attitudes that reflect the social nature of agents. Obligations can be violated, because agents are autonomous (Dignum, Kinny & Sonenberg, to appear). In a typical example, an agent has a desire to do otherwise and the desire is stronger than the obligation. Even social agents can violate their obligations if they intended earlier to do otherwise and they are not open-minded enough to reconsider this intention. Moreover, norms are often contradictory. In order to deal with conflicts, an agent must be able to drop some obligations in favor of others.

Intentions are considered here as the generated goals that the agent has selected in its previous deliberations. Intentions are incorporated to relate previously generated goals to the goal generation process. In the philosophical literature it has been argued, among others by Bratman (1987), that intentions deserve a special status besides beliefs and desires: they cannot be reduced to them. The incorporation of intentions possibly makes the agent's behavior more stable.

Desires are closely related to goals. In some approaches they are even identified with goals (Cohen & Levesque, 1990; Rao & Georgeff, 1991; Hindriks et al., 1999). The desires of an agent reflect its long term preferences, but also its acute wants and urges. They are often produced by an emotional or affective process. The theoretical notion of desire is used to model both kinds of input to the decision making process. In general an agent can select its goals, but it cannot select its desires. Desires are much more stable (Dastani, Huang & van der Torre, to appear). Desires are long term preferences, and when a desire is triggered by an observation or belief, then a short-term desire is turned into a goal. In this way one can even model the effect of biological survival mechanisms. For example, if an agent is without food for some period, this might trigger an acute desire for food.

BOID architecture

The agent's deliberation loop in Figure 1 is formalized by the control loop for the BOID architecture in Figure 2. It receives input from the environment (*Obs*), it calculates a set of candidate goal sets (*S*) based on a priority function (*?*), selects one goal set, decides which plans should be performed (*P*), updates all components, and starts observing the environment again.

```
select ??;
repeat
    Obs := read_environment();
    S := generate_candidate_goal_sets(Obs, B, O, I, D, ?);
    P := select_goal_set_and_generate_plans(S);
    update(B, O, I, D, ?, P)
until forever
```

Figure 2: Control loop for the BOID architecture

The goal generation of this control loop is explained in detail below. We map the issues discussed in the previous section onto computational concepts: conditional mental attitudes become input-output components, goal generation becomes extension generation, and overriding becomes a priority function.

BOID conditional mental attitudes

Conditional mental attitudes are formalized by components. The behavior of each component is modeled by a set of pairs of propositional logic formulas (*a*, *b*) that take the form of rules, written as *a* ??*X*? *b* with *X* ? {*B*, *O*, *I*, *D*}. These rules instruct the agent what inference steps to make. The rule *a* ??*B*? *b* implements that if *a* is derived as a goal, then the agent believes that as a

consequence *b* is a goal. The rule *a* ?O? *b* implements that if *a* is derived as a goal, then the agent is obliged to adopt *b* as a goal. The rule *a* ?I? *b* implements that if *a* is derived as a goal, then the agent intends *b* to be a goal. The rule *a* ?D? *b* implements that if *a* is derived as a goal, then the agent desires that *b* is a goal. In general, if *a* is derived as a goal and it is not inconsistent to derive *b*, then there is reason to believe, be obliged, intend, or desire that also *b* is a goal. The rules provide necessary but not sufficient justification for making the inference from *a* to *b*. There might be other rules blocking the inference, such as for example T ?X? ? *b*, in which T stands for any tautology like *a* ? ? *a*.

A set of BOID rules is not used to specify the resulting agent reasoning behavior, and they should therefore not be read as declarative specifications. A declarative specification formula like *B*(*a* ? ?*b*) means that the agent believes that the implication relation holds between the propositions *a* and *b*. The implication ? in this formula obeys different logical properties than ? ? , such as contraposition and reasoning by cases. See (Makinson & van der Torre, 2000) for a discussion on such logical properties. In other words, the operational reading of the rules implies that the symbol ? ? cannot be interpreted as a material implication.

BOID goal generation

Goal generation is formalized as the derivation of so-called extensions in default logic (Reiter, 1980). Default logic extends the inference rule modus ponens with two new mechanisms. First, there is a consistency constraint on the inference process, such that rules are only applied if they do not lead to an inconsistency. Second, the application of defeasible rules may result in conflicting outputs and thus in conflicting goal sets. They lead to alternative sets of logic formulas. The representation of conflicting mental attitudes by multiple consistent sets of formulas was proposed (for unconditional obligations) in the seventies by van Fraassen (1973) and was first related to default logic by Horty (1994). The representation of candidate goal sets by extensions is originally proposed by Thomason (2000). A motivation to use default logic is that extensions are constructed before conflicts are resolved. This can be contrasted with an approach in which a conflict is resolved as one is encountered. A conflict may be defined as a minimal set, in the sense that if two sets are conflicting then one of the sets cannot be a strict subset of the other one (Reiter, 1987). However, to resolve the conflict we have to consider the whole extension, because agents should consider the effects of goals before they commit to them.

In our approach, goal generation is based on *prioritized* default logic. The representation of a goal generation component in Definition 1 - our instantiation of a default theory - therefore not only contains a set of facts and a set of rules, but also a priority function ? on the rules. To keep the formal details

in this paper to a minimum we assume that individual extensions do not contain disjunctive information, that is, we assume that extensions are sets of positive or negated atomic formulas called literals. More advanced implementations of prioritized default logics are now being studied in the area of answer set programming, see e.g. (Brewka & Eiter, 1999).

Definition 1 ***(Goal generation component)*** *Let L be a propositional language and let rules be ordered pairs of L written as (a ?? b), with b a conjunction of literals over L. The goal generation component is represented by a tuple ?Obs, B, O, I, D, ?? with Obs a set of literals over L, and B, O, I and D sets of rules, and ? a function from B ? ?O ? ?I ? ?D to the integers.*

The goal generation procedure in Definition 2 starts with a set of observations *Obs*, which cannot be overridden, and initial sets of default rules for the component: *B*, *O*, *I*, and *D*. Moreover, it assumes an ordering function *?* on the rules of the different components. The procedure then determines a sequence of sets of extensions S_0, S_1, ???. 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 *?*-value are added to the extension separately, to form distinct extensions in S_{i+1}. The operator *Th*(*S*) refers to the closure of *S* under propositional logic consequence, 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 2 ***(Generate goal procedure)*** *Let ? = ?Obs, B, O, I, D, ?? be the representation of the goal generation component for propositional logic L, and let an extension E be a set of L literals (an atom or the negation of an atom). We say that:*

- *a rule (a ?? b) is strictly applicable to an extension E, iff a ? Th(E), b ? Th(E) and ? b ? Th(E);*
- *max(E, ?) ? B ? ?O ? ?I ? ?D is the set of rules (a ?? b) that are strictly applicable to E such that there does not exist a (c ?? d) ? B ? ?O ? ?I ? ?D strictly applicable to E with ? (c ?? d) > ? (a ?? b);*
- *E ? L is an extension for ? iff there exists an n such that E ? S_n and $S_n = S_{n+1}$ for the procedure in Figure* 3.

In some circumstances the goal generation procedure in Definition 2 is efficient, and in other circumstances it is not. For example, it efficiently generates one extension if all rules have a unique priority value, because in each

iteration at most one rule can be selected. However, the number of extensions increases quickly when the number of rules with the same priority value increases. It remains an open problem how to optimize the procedure in the general case.

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

Figure 3: Procedure to calculate extensions

We found it useful to implement a prototype of the goal generation procedure in Prolog, along with some examples. In this way one can try out different priority functions, and see if their behavior on the examples comes out as expected. The source code can be found at http://www.cs.vu.nl/~boid/. One of the examples contains the following mental attitudes. If the agent goes to the conference, then it believes that there are no cheap rooms close to the conference site. If it goes to the conference, then it is obliged to take a cheap room. If it goes to the conference, then it desires to stay close to the conference site. It intends to go to the conference. This example can be represented by the following rules with priority function ? ?:

(go_to_conference ? ?cheap_room) ??B? ? close_to_conf_site	*(?*	=	*5)*
(go_to_conference ? ?close_to_conf_site) ??B? ? ?cheap_room	*(?*	=	*4)*
T ??I? *go_to_conference*	*(?*	=	*3)*
go_to_conference ??O? cheap_room	*(?*	=	*2)*
go_to_conference ??D? close_to_conf_site	*(?*	=	*1)*

Let the input of the agent be empty. Then, following the extension calculation mechanism, we first derive all beliefs and intentions, resulting in extension *{go_to_conference}*. Because it is a social agent, the obligation rule is applied first. This results in the intermediate extension *{go_to_conference,*

cheap_room}. It is returned to the belief component, where it triggers the first belief rule. This produces the following extension, which denotes the situation in which the agent has decided to go to the conference and take a cheap room not close to the conference site. This is social behavior in case the room is paid from the traveling budget of the research group.

{ go_to_conference, cheap_room, ? ?*close_to_conf_site }*

Moreover, consider a ? that assigns 2 to the desire rule and 1 to the obligation rule. This results in the following extension, which represents the situation in which an agent has decided to go to the conference and takes an expensive room close to the conference site. This may be called selfish behavior.

{ go_to_conference, close_to_conf_site,? ?*cheap_room }*

Finally, assume that ? assigns to both the desire and obligation rule the same priority value 2. So ? no longer assigns unique values to the rules, and the goal generation procedure now derives both extensions. The agent has to select one of the candidate goal sets later.

We index all formulas of the extension with the name of the component from which they are derived to keep track of their origin. This information is used in the goal selection and plan generation components, not in the goal generation component. The formulas derived from the belief component should for example be removed from the generated extensions since these formulas are thought to be about the world (informational attitude) and do not form goals (motivational attitude).

Specialized BOID architectures

Agent types are used to distinguish, classify and compare agent behavior, and can be defined in all parts of the BOID architecture. Agent types for goal generation are based on overriding, such that for example, in realistic agents beliefs override other mental attitudes and in social agents obligations override desires. We also illustrate that agent types cannot only be used to classify agent behaviors, but also to compare them. We discuss two ways in which overriding can be made computational, either as constraints on the priority function ?, or hard-wired in the architecture.

Agent types for goal generation

Agent types for goal generation are conflict resolution methods. An agent has a conflict if the goal generation procedure in Definition 2 derives multiple extensions. A conflict is resolved if the priority function is adapted such that no alternative extensions are generated. A mental attitude conflicts with another mental attitude if two rules from the different attitudes are applica-

ble, but applying both leads to an inconsistent set. A rule overrides another rule if it has a higher priority. Agent types for goal generation are formalized in Definition 3 as constraints on the set of available priority functions. When the deliberation loop starts, the selected priority function obeys the constraints corresponding to the agent type, and when in the update procedure another priority function is selected, it has to obey those constraints too. An agent type is called primitive if it contains only one constraint, and complete if it induces a total strict ordering on the components.

Definition 3 *An agent type for goal generation is a consistent set of constraints on priority function ? of the form X Y with X, Y ? {B, O, I, D} defined as follows: ? ?rule_x ? ?X, ? ?rule_y ? ?Y, ? (rule_x) > ? (rule_y)*

A primitive agent type contains a single constraint. A complete agent type is a maximal consistent set of constraints. There are twelve primitive agent types, which are listed in Table 1 together with the corresponding constraints. They are ordered in six pairs, each agent type *X Y* together with its opposite *X Y*.

Constraints				Agent type
B	O	(O	B)	Realistic relative to obligations (dogmatic)
B	I	(I	B)	Realistic relative to intentions (over-committed)
B	D	(D	B)	Realistic relative to desires (wishful thinker)
O	I	(I	O)	(Un)Stable relative to obligations
O	D	(D	O)	Social (selfish)
I	D	(D	I)	(Un)Stable relative to desires

Table 1: Twelve primitive agent types

An agent type is a set of primitive agent types. For example, the realistic agent type is {*B O*, *B I*, *B D*}, the stable agent type is {*I O*, *I D*}, the social stable agent type is {*I O*, *I D*, *O D*}, etc. Moreover, agent types can be derived. For example, since orderings are transitive we can derive that an agent which is unstable relative to obligations (*O I*) and stable relative to intentions (*I D*) is social (*O D*). There are twenty-four complete agent types, of which the six realistic ones are listed in Table 2.

Constraints					Agent type
B	O ; O	I ; I	D		Realistic, unstable-O, stable-D, social
B	O ; O	D ; D	I		Realistic, unstable-O, unstable-D, social
B	I ; I	O ; O	D		Realistic, stable-O, stable-D, social
B	I ; I	D ; D	O		Realistic, stable-O, stable-D, selfish
B	D ; D	O ; O	I		Realistic, unstable-O, unstable-D, selfish

B	D ; D	I ; I	O	Realistic, stable-O, unstable-D, selfish

Table 2: Six complete realistic agent types

The definition of agent types leads to a simple way in which agent types can be compared. Agent type *A* is at least as general as agent type *B* if all the priority functions that respect the constraints of agent type *B*, also respect the constraints of agent type *A*. The generality relation between realistic agent types forms the lattice visualized in Figure 4, if we add a top element.

This figure should be read as follows. The level in this hierarchy indicates the generality of agent types. The bottom of this lattice is the realistic agent type. Each higher layer adds additional constraints resulting in more specific agent types. The top of this lattice is the falsum, which indicates that adding any additional constraint to the agent types at the lower level results in an inconsistent agent type. Just below the falsum are the six complete realistic agent types.

There are also other constraints on priority functions. One of them is the following unique extension property, which says that ? associates with each rule a unique integer. It induces a strict total order on the rules.

Definition 4 ***(Unique goal set)*** *A goal generation component that generates unique goal sets is represented by a tuple ?Obs, B, O, I, D, ? ? with ? a function from B ? ?O ? ?I ? ?D to the integers such that ? (x) = ? (y) implies x = y.*

Another constraint on priority function ? is the following component order property.

Definition 5 ***(Component order)*** *A goal generation component represented by a tuple ?Obs, B, O, I, D, ? ? induces a strict component order when ? is a function from B ? ?O ? ?I ? ?D to the integers such that for all X, Y ? {B, O, I, D} with X ? Y we have either X Y or X Y.*

The only agent types that satisfy the component order property are the complete agent types. Lack of the component order property is illustrated by the realistic agent. It 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.

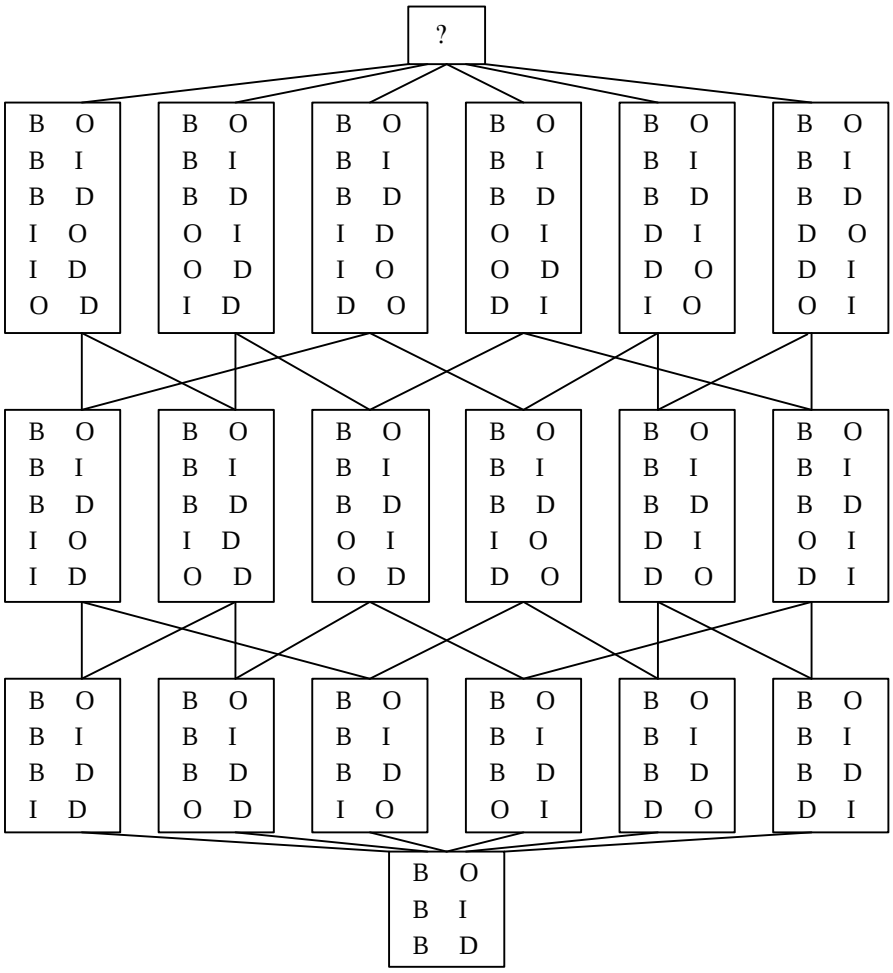

Figure 4: The lattice structure of realistic agent types.

Mapping agent types to agent architectures

A computational agent architecture specifies the components of an agent, how they are related, and how the information flows around. The combination of the goal generation procedure with an agent type can be mapped to a specialized agent architecture. This works as follows. We add an information link from component *X* to component *Y* if *X* *Y* and there is no component *Z* such that *X* *Z* and *Z* *Y*. Moreover, we add links from all components *X* to components *Y* for which there exist no component *Z* such that *Z* *Y*. We give two examples of such mappings and how they must be interpreted.

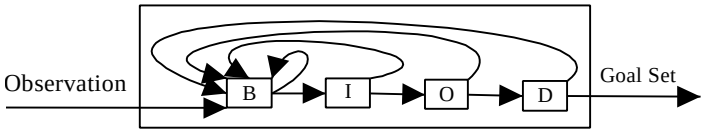

Figure 5: Goal generation component for stable social agents.

First, consider the goal generation component of the realistic social stable agent type in Figure 5 that is specified by the constraints *B I O D*. Belief rules are applied iteratively, indicated by the incremental loop around the *B* component. If no more belief rules are applicable, then the calculated set of extensions is sent to the *I* component. If possible, one intention is applied and the set of extensions is sent back to the *B* component via the incremental loop from *I* to *B*; otherwise the set of extensions goes to the *O* component, etc. The goal generation component visualized in Figure 5 represents an ordering among mental attitudes where the connections between the components fully determine how sets of extensions flow around.

Second, consider the selfish unstable-D agent type that is specified by a partial component ordering. As a consequence of the partiality of the component order, there is a component that is connected to two other components. This agent type is mapped to the goal generation component, as visualized in Figure 6. This goal generation component should be interpreted as above: if the set of output extensions of a component differs from the set of input extensions, it flows back through the incremental loop; otherwise it flows from the *B* component to the *D* component and from the *D* component to either *O* or *I* component.

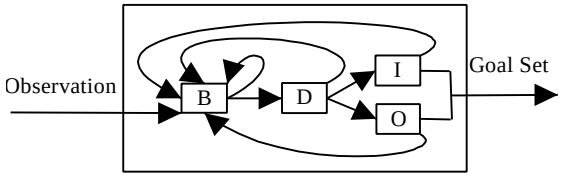

Figure 6: Goal generation component for selfish unstable-D agents.

The specialized architectures suggest several enhancements to the generic goal generation architecture in Definition 1 and 2. For example, in the generic architecture all extensions are calculated at the same time, whereas the specialized architectures suggest that also a non-deterministic choice may be made during extension calculation. Moreover, the generic architecture is based on sets of default rules, whereas the specialized architectures

suggest that components can be implemented otherwise. For example, the belief component may maximize cross entropy or apply AGM belief revision (Alchourrón, Gärdenfors & makinson 1985), and its output may be a probability distribution, a set of them, plausibility measures, a belief set, etc. Such a component needs an interface (called a wrapper) which translates the input and output of the component to a propositional language. Summarizing, a set of rules can be interpreted as an *abstract* input-output description of a component, which can be implemented in a variety of different ways.

Consequences for planning and updating attitudes

Given the set of extensions as candidate sets of goals, a goal set should be selected and a plan should be generated. Thereafter, before new candidate goal sets can be generated, the mental attitudes should be updated. Below we sketch how these notions can be made computational. Planning becomes abduction, and attitude revision becomes a rule update procedure.

Select goal set and generate plan

The goal selection component selects one extension from the set of extensions generated by the goal generation component. Each extension represents one possible future state of affairs. Several selection strategies are possible to select an extension for planning and execution. We can select the smallest extension or use some domain dependent selection strategies, based on additional information about preferences, priorities or costs. Selection can also be based on the compatibility or similarity of the new extension with previously selected extensions or with additional observations. One could for example distinguish a persistent agent type, which typically selects the extension most similar to previously selected extensions, or a conservative agent type, which selects the extension that differs least from the current state of affairs. The selection must be based on the feasibility of the extension. In case the selected extension cannot be translated into a feasible plan, the second best extension is selected for execution.

In the BOID architecture, knowledge about the preconditions and effects of actions and plans is typically stored as situation-action rules in the belief component. Also the structure of plans, the dependencies and preferred order of execution can be stored as defeasible belief rules. The rules encode what would happen, if the action were carried out. Actions that are part of a committed plan are best stored as conditional rules in the intention component. Once we decided on a plan, we intend to continue to carry it out, unless some major reconsideration is called for. The exact details of planning and scheduling are beyond the scope of this paper.

On a conceptual level, our views have been influenced by classical decision theory and planning under uncertainty, in which the effects of actions

and events is seen as the manipulation of variables. We apply the distinction between so called decision variables and parameters (Lang, van der Torre & Weydert, to appear). An agent can directly control the truth value of the decision variables, but not the truth value of the parameters. In multi-agent systems for example, each agent has its own decision variables, corresponding to the actions it can perform (Dastani & van der Torre, 2002). It cannot influence external events, nor change the value of other agents' decision values. It can only try to collaborate. Although such a view of actions is simple, it is sufficient to discuss the role of planning in the BOID architecture.

At first sight, it seems we should restrict goal generation to decision variables only. One may argue that we should not generate or select goals that we cannot see to. However, it may be that we cannot see to it directly, but we can see to it indirectly. For example, consider a decision variable *a*, a parameter *p*, a belief rule *a* ?B? *p* and a desire rule T ?B? *p*. The agent reasons as follows. First, in the goal generation component it generates the goal or extension {*p*}. Second, in the plan component - which also has access to the relevant belief rules - the agent derives from goal {*p*} and belief rule *a* ?B? *p* that it should see to it that *a*, that is: *do(a)*. Hence, the agent believes that a side effect of *do*(*a*) is that *p* will become true. Note that *a* is not derived by applying the belief rule, but by applying it in reverse. This form of reasoning is known as abduction. We believe that planning by abduction is an efficient way to represent means-end reasoning in the BOID architecture.

There is a computational issue which thus far has not been addressed. In decision theory, one way to find a plan is to order all possible plans and select the best one. Compared to this one stage decision making process, the two stage process of first selecting goals and thereafter generating plans to reach goals, may lead to inferior choices. After all, some information may get lost in the process. However, the two stage process can be much more efficient, because the number of possible plans is exponential in the number of decision variables. Moreover, there is a trade-off problem. In some applications it is better first to calculate the complete candidate goal sets and then find actions to achieve them, in other applications it is better to calculate partial goals first, look for ways to achieve them, and then continue to calculate the goal sets (Dastani, Hulstijn & van der Torre, 2001).

Update mental attitudes

The conditional representation of mental attitudes introduces the distinction between updating a conditional attitude and deriving an attitude from an existing rule. The general guideline is that conditional attitudes represent long term attitudes. During goal generation formulas are added to the current goal set. For example, the observation of an advertisement triggers the belief rule that ice cream is available. This triggers a desire to have an ice

cream. The agent then adds the intention of buying the ice cream, which will subsequently generate the obligation to pay. But the agent can also make long term adjustments to the rules that model its attitudes. These adjustments are motivated by some external observation or communication, by a learning process or discovery, or by the process of deliberation which produces intentions to influence future decision making. For example, when agents enter institutions or organizations, their set of conditional obligations is updated with the appropriate norms and obligations. There are also cases which can be formalized either way. For example, by signing a contract, an agent becomes obliged to perform according to terms stated. This can be formalized as an update of the obligation rules as well as an application of existing rules. The guide line here is that it is most efficient to change the rule base as little as possible. If many contracts are signed, then it is better to have a generic obligation rule for signing contracts.

Goal generation and goal selection and planning are related to each other by updates of the set of intention rules. If in the planning component an agent decides to see to it that b, i.e. $do(b)$, then $\top \Rightarrow b$ is added to the intentions used in the next goal generation. Updating intention rules is known as intention reconsideration. Usually there is a trade-off between keeping an intention, thereby maintaining stability in decision making, and reconsidering or abandoning intentions, when the environment has changed so much that the original motivations for adopting the intention no longer make sense (Bratman, 1987). A common commitment strategy abandons intentions in case the intention or its motivation have become fulfilled, or the intention or its motivation can no longer be achieved, or its motivation may be reached by some other means. Intentions therefore do not automatically remain in the extension after each loop, but must be reinforced by the original desire or obligation rule that gave rise to it.

Each update of rule based components can be constructed from primitives $add(a \Rightarrow_X b, ?)$ and $delete(a \Rightarrow_X b, ?)$, where $X \in \{B, O, I, D\}$. We do not require that rules are mutually consistent, since they are only applied when they do not contradict the given extension. However, the belief and intention component must remain internally consistent. Moreover, inconsistencies and multiple applicable rules slow down and complicate the process of extension calculation. Therefore, some form of `bookkeeping' is required to keep the rule base manageable, see e.g. work in truth maintenance systems (Doyle, 1979) and clever implementations of production systems like the RETE algorithm (Forgy, 1982).

Related research

In general, there are two types of approaches to study, model and translate mental attitudes and their balance into a cognitive agent architecture. The

first type of approach starts with analytical (philosophical) or empirical (psychological) studies of mental attitudes, their properties and interactions. Based on these studies, agent logics and agent architectures are developed in which the properties and interactions are translated and formalized. The proposed logics and architectures are then supposed to model the expected behavior of rational agents. An example of this type of approach is the BDI model (Bratman et al., 1988; Rao & Georgeff, 1991). The second type of approach works the other way around. It starts by proposing agent logics and architectures which are easy, intuitive and computationally tractable. The inspiration for proposing these systems originate from computational systems in computer science and artificial intelligence. In these systems the representation of mental attitudes, their properties and interactions are studied and related to the expected agent characteristics such as agent behavior or agent type.

Our approach is inspired by default logic and it is thus of the second type. However, it is also inspired by investigations in philosophical logic which formalize conditional mental attitudes in terms of rules reminiscent of production systems. In particular our proposal is inspired by deontic logic (von Wright, 1999; Alchourrón, 1993; Makinson, 1999) and the so-called input-output logics (Makinson & van der Torre, 2000).

A detailed comparison with other agent architectures is beyond the scope of this paper. We believe that goal generation procedures can be built in most other goal based architectures. The goal generation component based on interaction between mental attitudes can be interpreted as a society of minds, and it can be described with the Russian doll metaphor.

There are many other approaches to conflict resolution in artificial intelligence. Our use of priorities is different from Thomason's ad hoc conflict resolution mechanism, see (Broersen et al., 2001b) for a detailed comparison. Another alternative is to associate losses and gains with the rules. Such a decision-theoretic approach is by itself a line of research with many interesting problems (Dastani et al., 2001). There are also many other approaches to planning, see e.g. (Thomason, 2000).

Concluding Remarks

In this paper we have considered the question how goal generation can be decomposed into interaction between mental attitudes from a conceptual as well as a computational point of view. We obtain the following results. We introduce the BOID architecture in which goals are generated from the conditional mental attitudes beliefs, obligations, intentions and desires. We observe that goal generation architectures can be classified according to agent types based on overriding of mental attitudes. Agent types can be compared in generality, which results in a lattice of agent types. We implement the

BOID architecture by relating conditional mental attitudes to components, goal generation to extension construction, and agent types to constraints on priority functions. We show how agent types can be used to construct specialized BOID architectures. This is possible due to the close correspondence between the conceptual decomposition and the implementation.

One of the motivations for extending the repertoire of belief, desire and intention with obligation is that it allows us to specify normative agents, which can freely select to follow norms or violate them. The obligation component can thus be used to model the social structure of groups of agents. In addition to the autonomous BOID systems of this paper, we therefore intend to investigate multi-BOID systems. In fact, the name `BOID' is partly inspired by Craig Reynolds' generic flocking creatures. Firstly, we would like to extend the BOID approach to joint intentions and collaborative planning. Current formal work on joint planning and action uses modal logics for individual intention (Wooldridge & Jennings, 1999, Dunin-Keplicz & Verbrugge, 2001). However, such models have little to say on the `social glue' that keeps joint plans together. For example, collaboration requires participants to notify the others when they can not perform their part. Such normative conventions can be violated, just like other norms. Secondly, we are interested in the cooperative behavior of groups consisting of agents of different types. Consider for example selfish and social agents who must survive in a prisoner's dilemma setting in which the short term gains of selfish behavior may not outweigh long term benefits from social behavior.

Acknowledgement

Thanks to Zhisheng Huang for important contributions to the BOID project.

References

Alchourrón, C. (1993). Philosophical Foundations of Deontic Logic and the Logic of Defeasible Conditionals. In: J.-J. Meyer & R. Wieringa (Eds.): *Deontic Logic in Computer Science: Normative System Specification*. John Wiley & Sons, pp. 43-84.

Alchourrón, C., Gärdenfors, P. & Makinson, D. (1985). On the logic of theory change: partial meet contraction and revision functions. *Journal of Symbolic Logic* 50, 510-530.

Boella, G. & Lesmo, L. (to appear). A game theoretic approach to norms and agents. *Cognitive Science Quarterly,* this volume.

Bratman, M. E. (1987). *Intention, Plans and Practical reason*. Harvard University Press, Cambridge MA.

Bratman, M. E., Israel, D. J. & Pollack, M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence* 4(4), 349-355.

Brewka, G. & Eiter, T. (1999). Preferred answer sets for extended logic programs. *Artificial Intelligence* 109, 297-356.

Broersen, J., Dastani, M. , Huang, Z. Hulstijn, J. & van der Torre, L. (2001a). The BOID Architecture : Conflicts between beliefs, obligations, intentions and desires. In: *Proceedings of the Fifth International Conference on Autonomous Agents (AA'2001).* pp. 9-16, ACM Press.

Broersen, J., Dastani, M. & van der Torre, L. (2001b). Resolving conflicts between beliefs, obligations, intentions and desires. In: S. Benferhat & P. Besnard (Eds.): *Symbolic and Quantitative Approaches to Reasoning and Uncertainty. Proceedings of ECSQARU'01.* pp. 568-579, Springer Verlag, Berlin. Extended and revised version to appear in Journal of Applied Non-Classical Logics.

Castelfranchi, C., Dignum, F., Jonker, C. & Treur , J. (1999). Deliberate Normative Agents: Principles and Architecture. In: N. Jennings & Y. Lesperance (Eds.): *Intelligent Agents VI: Proceedings of The Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL'99).* pp. 364-378.

Cohen, P. & Levesque, H.(1990). Intention is choice with commitment. *Artificial Intelligence* 42, 213-261.

Dastani, M., Huang, Z. & van der Torre, L. (to appear). Dynamic Desires. In: S. Parsons, P. Gmytrasiewicz, & M. Wooldridge (Eds.): *Game Theory and Decision Theory in Agent-Based Computing.* Kluwer.

Dastani, M., Hulstijn, J. & van der Torre, L. (2001). BDI and QDT: a comparison based on classical decision theory. In: S. Parsons & P. Gmytrasiewicz (Eds.): *Game Theoretic and Decision Theoretic Agents: Papers from the AAAI Spring Symposium (GTDT'01).* pp. 16-26, AAAI Press.

Dastani, M. & van der Torre, L. (2002). What is a joint goal? Games with beliefs and defeasible desires. In: *Proceedings from the Ninth International Workshop on Non-Monotonic Reasoning (NMR'02).*

Dignum, F., Kinny, D. & Sonenberg E. (to appear). From desires, obligations and norms to goals. *Cognitive Science Quaterly*, this volume.

Dignum, F., Morley, D., Sonenberg, E. & Cavedon, L. (2000). Towards socially sophisticated BDI agents. In: E. Durfee (Ed.): *Proceedings of the International Conference on Multi-agent Systems (ICMAS'00).* pp. 111-118, IEEE Press.

Doyle, J. (1979). A Truth Maintenance System. *Artificial Intelligence* 12, 231-272.

Dunin-Keplicz, B. & Verbrugge, L. C. (2001). *Collective Intentions*. Technical Report AI-2001-2, Artificial Intelligence - University of Groningen, The Netherlands.

Forgy, C. L. (1982). RETE: A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem. *Artificial Intelligence* 19, 17-37.

Hansson, B. (1969). An Analysis of Some Deontic Logics. *Nous* 3, 373-398.

Hindriks, K. V., de Boer, F. S., van der Hoek, W. V. & Meyer, J.-J. C. (1999). Agent Programming in 3APL. *Autonomous Agents and Multi-Agent Systems* 2(4), 357-401.

Horty, J. F. (1994). Moral dilemmas and nonmonotonic logic. *Journal of philosophical logic* 23, 35-65.

Katsuno, H. & Mendelzon, A. (1992). On the difference between updating a belief base and revising it. In: P. Gärdenfors (Ed.): *Belief Revision*. Cambridge University Press, pp. 183-203.

Laird, J. E., Newell, A. & Rosenbloom, P. S.. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence* 33(1), 1-64.

Lang, J., van der Torre, L. & Weydert, E. (to appear). Utilitarian Desires. *Autonomous Agents and Multi-Agent Systems*.

Makinson, D. (1999). On a fundamental problem of deontic logic. In: P. McNamara & H. Prakken (Eds.): *Norms, Logics and Information Systems. New Studies on Deontic Logic and Computer Science.* pp. 29-54, IOS Press.

Makinson, D. & van der Torre, L. (2000). Input / output logics. *Journal of Philosophical Logic* 29, 383-408.

Makinson, D. & van der Torre, L. (2001). Constraints for input / output logics. *Journal of Philosophical Logic* 30, 155-185.

Newell, A. (1982). The Knowledge Level. *Artificial Intelligence* 18(1), 87-127.

Rao, A. S. & Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In: J. Allen, R. Fikes, & E. Sandewall (Eds.): *Proceedings of the International Workshop on Knowledge Representation (KR91)*. pp. 473-484, Morgan Kaufmann, San Mateo CA.

Reiter, R. (1980). A logic for default reasoning. *Artificial Intelligence* 13, 81-132.

Reiter, R. (1987). A Theory of Diagnosis From First Principles. *Artificial Intelligence* 32, 57-95.

Thomason, R. (2000). Desires and Defaults: A Framework for Planning with Inferred Goals. In: A. G. Cohn et al. (Eds.): *Proceedings of the 7th International Conference on the Principles of Knowledge Representation and Reasoning (KR'00)*. pp. 702-713, Morgan Kaufmann, San Mateo, CA.

van der Torre, L. & Tan, Y.-H. (1999). Contrary-To-Duty Reasoning with Preference-based Dyadic Obligations. *Annals of Mathematics and Artificial Intelligence* 27, 79-128.

van Fraassen, B. (1973). Values and the Heart Command. *Journal of Philosophy* 70, 5-19.

von Wright, G. (1983). *Practical Reason: Philosophical papers, volume 1*. Basil Blackwell, Oxford.

von Wright, G. (1999). Deontic logic: as I see it. In: P. McNamara & H. Prakken (Eds.): *Norms, Logics and Information Systems. New Studies on Deontic Logic and Computer Science*. IOS Press, pp. 15-25.

Wooldridge, M. J. & Jennings, N. R. (1999). The Cooperative Problem-solving Process. *Journal of Logic and Computation* 9(4), 563-592.