CHAPTER 2: Beliefs, Desires, Intentions (BDI)

The Belief-Desires-Intentions (BDI) approach is one of the major approaches to building agents and multiagent systems, including commercial agent software. This approach is inspired by logics and psychology, and as the name implies, the key here is build agents using symbolic representations of agents' beliefs, desires, and intentions.

In the following, we begin with an informal description of the BDI approach, then provide a more formal, logical description and finally describe practical BDI systems. We will use two types of domains for illustration: (i) A "Mars Rover" domain, which refers to a Mars Rover of the future, that is significantly more autonomous than the current NASA rovers (Spirit and Opportunity); (ii) A university domain, which provides examples from our daily routine for comparison, using ourselves as agents.

This chapter is divided into three main parts. In the first part of the chapter we will provide an informal description of the BDI approach to outline the key ideas underlying this approach. We will next provide a logic-based perspective on BDI. Logic helps us define the BDI approach in a more precise manner. In the final part of this chapter, we will discuss practical BDI agent systems that are based on this logic.

BDI: An informal Description

The key to the BDI approach is that it focuses on reasoning in resource-bounded agents. Resource bounds arise because agents may have limited computational power or limited memory; so agents' reasoning itself takes time, it cannot be infinitely fast. Furthermore, these agents act in environments that are dynamic, i.e. the environment may change even while the agent is engaged in its reasoning. The implication is that: (i) An agent cannot shut off the world around it and keep reasoning: it may take a very long time to come up with a plan, but by the time it has planned everything, the world around it may have changed sufficiently to invalidate the plan. For instance, a Mars-Rover may take a long time to plan its day, but a dust-storm may invalidate its plan even before it begins execution of the plan; (ii) After an agent has generated its plan, if new dangers or opportunities arise, or if the world changes, the agent cannot engage in open-ended reasoning about any and all such changes, to keep coming up with new plans. It will be overwhelmed with replanning. For example, suppose the Mars-Rover has carefully planned its day of scientific observations. Now, if it replans from scratch every time it comes to know a new fact or new possibility for scientific observation, it will spend all its time planning and replanning, and it may never get around to actually doing anything (to acting).

One approach to get around these two difficulties is to avoid any notion of planning or reasoning, and build agents that are completely reactive. In the extreme case, no agent would ever generate any plan; instead it would just react to what it sees. So according to this approach, our Mars Rover will not plan its day. Similarly, no student or professor at a university would plan their day. If they just happen to pass by a coffee shop and they are hungry, they may react by feeding themselves. However, eliminating planning completely is unsatisfactory --- we would require too much effort at system design time to carefully build all of an agent's reactions, and it appears counter-intuitive.

BDI approaches attempt to counter these difficulties by their use of intentions to constrain the amount of reasoning. Let us now go through each of the B, D, and I terms in this approach:

Beliefs are facts representing what an agent believes about the world, i.e. an agent's respresentation of the state of the world. Agents' may obtain such beliefs from sensing their world, as discussed in Chapter 1. For instance, a Mars Rover's video camera may show it is close to an important location called "Husband hill". However, an agent's beliefs need not necessarily be true. Thus, our Mars-Rover agent may believe that a martian dust storm is approaching, but none may be approaching in reality (the rover's camera might be slightly faulty, giving a false impression of an approaching dust storm). In other cases, agents' beliefs may be based on internal inferences. To take an example from our daily routine, an agent may believe that the CS499 class will meet on Sept 4, 2006, when in fact that is the labor day holiday.

Desires are goals or some desired end states. An agent may have multiple desires, which may possibly be in conflict. For example, the Mars-Rover agent may have a desire to dig for water-ice on Mars at location-1, another desire to dig at location-2, and also a desire to protect itself from an approaching dust storm. Similarly, in our university setting, an agent may have a desire to be in the CS499 class on time at 2 pm, but it may also have a desire to play soccer at 2 pm.

Intentions refer both to an agent's commitments to its desires (goals) and its commitment to the plans selected to achieve those goals. Intentions cannot conflict with each other; they have to be consistent. Thus, an agent may intend only a subset of its desires. Examples of intentions include the following: (i) A Mars Rover may intend to dig for water-ice at location-1; (ii) A student may intend to get to class at 2 pm; and thus cannot intend to play soccer at 2 pm.

Since intentions are fundamentally built on the notion of commitments, we will first discuss the notion of commitments in more detail. A commitment implies stability: once

committed to a particular goal (or a plan to achieve that goal), an agent should not easily drop that commitment or easily reconsider that commitment. Such commitments are important because agents are resource bounded: agents have limited time and limited computational power. An agent arrives at a commitment after reasoning about different alternatives. Once it decides on a commitment, this commitments helps constrain an agent's reasoning:

> Commitments inform an agent about what to reason about. That is, commitments set out problems for agents: Agents should try to plan to fulfill their commitments. For example, a Mars-Rover committed to digging at location-1 will need to plan a path to location-1. If a student is committed to getting to a class at 2 pm, then the commitment requires that the student reason about getting to USC by 2 pm.
>
> Commitments also inform an agent about what not to reason about. That is, commitments provide a filter of admissibility: very quickly rejecting opportunities that conflict with commitments, without even having to reason about them in depth. For example, if an agent is committed to attending the CS499 class at 2 pm Monday, then the agent can quickly dismiss a proposal from a friend to meet at 2 pm Monday. No detailed re-evaluation and recomputation of all available classes, schedules need be done to arrive at this decision.

Of course, while an agent should commit to plans for the reasons given above, this does not imply a fanatical level of commitment. Fanatical commitment is also detrimental, e.g. such an agent may never give up its commitment even if it has new beliefs that its commitment is unachievable. For example, suppose a Mars-Rover has a commitment to digging for water-ice at 1ocation-1. However, if it comes to believe that a major dust storm is approaching, it should reconsider this commitment. Similarly, suppose a student is committed to getting to her class at 2 pm. However, if the hospital calls, generating a new belief that a close friend is seriously ill, the student should reconsider her commitment. We will have more to say about reconsidering commitments later.

From commitments to intentions: We have said earlier that intentions are fundamentally based on the notion of commitments. An intention implies that when committed to achieving a goal, an agent must believe it is/was committed to that goal. That is, the agent must be aware that it has the commitment so that it can reason about it --- whether to keep it or remove it. However, researchers often do not emphasize this difference between intentions and commitments. Accordingly in general in the rest of the notes, we will not emphasize this difference between commitments and intentions; we will treat them both as the same.

Commitments are also related to another important concept: *satisficing*. Some of you may be familiar with the concept of optimizing. Optimizing is the idea of coming up with the best

possible, e.g. it may involve removing absolutely all inefficiencies from a plan to make sure it is completely optimal. No further improvements are possible. Or, optimizing may involve ensuring that we get maximum possible usage of some equipment. In contrast, the notion of satisficing is being good enough. A satisficing solution may not be the best solution. A satisficing course of action may not be the best course of action, but it is good enough. For example, if the Mars-Rover commits to a plan and does not reconsider it given a new opportunity, it essentially engages in satisficing, giving up on optimality of the plan --- a more optimal plan would have taken the new fact into consideration. However, the amount of time spent planning and replanning may be simply enormous. Thus, the reason an agent chooses to satisfice rather than optimize is that coming up with an optimal course of action may take too long to plan. In essence, BDI systems choose to satisfice rather than optimize in order to ensure that they are not overwhelmed with planning; that they do get around to acting. (Satisficing as a concept was one of the main ideas of Prof. Herb Simon, one of the founders of Artificial Intelligence).

So far, we have taken a quick informal tour of the concepts involved in BDI. Why use a BDI approach in building agents? Computer science has attempted to continually raise the level of abstraction with which to program or instruct computers. While originally we programmed computers using machine code or assembly code, higher level languages and structured programming followed. Object-oriented programming was the next step. In BDI systems, the goal is to take this level of abstraction one level further by using beliefs, desires and intentions, which are abstract concepts (certainly, more abstract than a "goto" statement!). The key is to provide higher and higher levels of instructions to program agents, and let the underlying system/computer do more of the work. And even if don't program our agents so much in BDI, it may be at least useful as a way of describing how these agents operate. That is, rather than describing them as executing "goto" or executing "while-do" loops etc, it is better to describe them in terms of their beliefs, desires, intentions.

BDI Logic

Now to the logic. We will try to express the notion of a commitment more formally (more precisely). BDI logic can be used in one of three ways: (i) to provide high-level insight or ideas; (ii) to provide a detailed design specification; (iii) to provide an actual operational program, where we use the logic as a way to program and agents reason by proving logical theorems. In these lecture notes, we focus on (ii).

This set of notes is based on the highly influential Cohen and Levesque logic

This is one popular logic for beliefs, desires, intentions; but by no means the only one

It will also be used later in teamwork chapter

Logical Formulae and Notation

Key formulae used in this logic, where in the following x is an agent:

(Bel x P): x has P as a belief
(Goal x P): x has P as a goal
(Eventually P): P is true now or somewhere in the future
(Until P Q): Q is TRUE until P TRUE
(Always P): P is always true
(Never P): P is never true

Examples:

- *(Bel Jon 499-meets-Monday-Wednesday) means that agent Jon believes that the 499 class meets every Monday and Wednesday.*
- *(Bel David (Never 499-meets-Tuesdays)) means that agent David believes that the 499 class never meets on Tuesdays.*
- *(Bel David (Bel Jon 499-meets-Tuesdays)) means that agent David believes that agent Jon believes that the 499 class meetings on Tuesdays. The key point is that an agent may have beliefs about beliefs.*

- We will use "AND" and "OR as connectives, in the same way as you would have seen them in CS101 in programming in C. So for instance, we can construct a new formula such as: (Always (P OR Q)). Similarly, we can define (Know x P) as:

  (Know x P) definition: P AND (Bel x P)

  The point is that we can see that knowing requires that P be actually true in the real-world.

  "NOT" will be used as a prefix to a formula to negate it, e.g. NOT(Q).

  We will use P→Q to indicate that we can infer Q from P. For example, a key axiom that we will assume regarding beliefs is that:

  ((Bel x P) and (Bel x (P → Q)))
  → (Bel x Q)

- Lets consider how we can express some of the terms in terms of one another:

  (Always P): (NOT (Eventually NOT(P)))
  (Never P): (Always (NOT(P))

We may denote ALWAYS by a box (□) and EVENTUALLY by a diamond (◇). Not is denoted by: Ø. As would be expected, Ø(ØP) is the same as P. There is also a way to tie the terms ALWAYS and EVENTUALLY as follows:

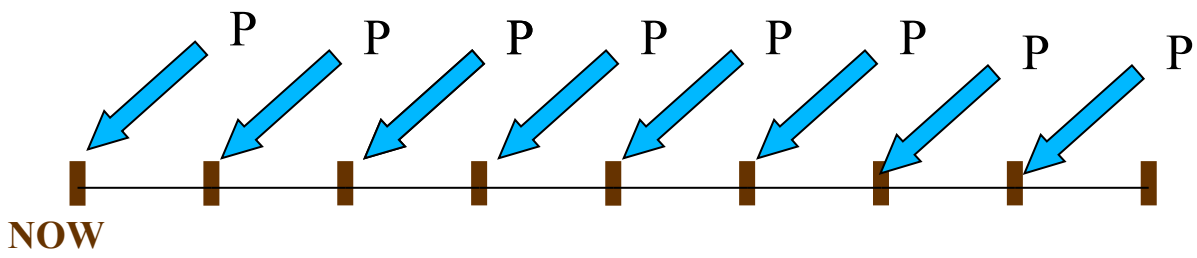
□P: Ø(◇ (ØP))

To test whether we understand these concepts, lets quickly try to answer the following question. Are the following two assertions the same:

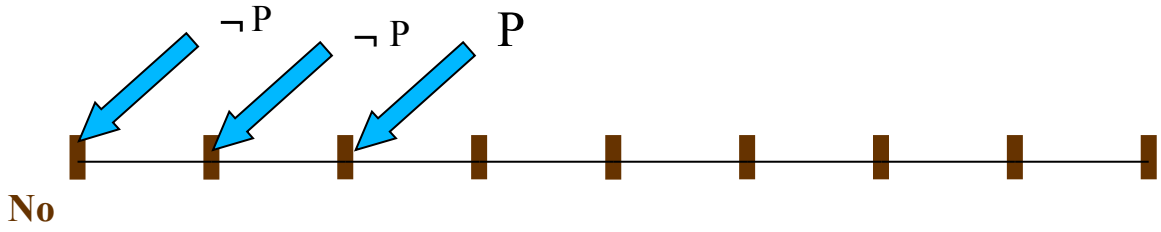
Ø(□P) ?? □(ØP)

No. The first statement says it is not the case that P is always true. So sometime in the future P is false. The second statement states that it is always the case that P is false, i.e. at all points in the future P is false. So the first allows for P to be true now. The second disallows such a possibility.

In fact, the way to understand a formula is to imagine a series of time points extending from now infinitely into the future. When we say □P it means that P is true at all time points from now and at all points in the future. Thus, we can provide the following diagrammatic depiction.



◇ (P) implies that P is true now or some time in the future, which could be seen to be something like this.



Note that we can draw logical inferences based on the above formulae. For example, if we are given: ◇ (P OR Q) AND □(ØP) then we may infer ◇ Q. That is:

(◇ (P OR Q) AND □(ØP))
→ ◇ Q

Informally, we are given that either P or Q are true sometime in the future: ◇ (P OR Q) . And we are given that it is always the case that P is false: □(ØP). Then sometime in the future, Q is true. Similarly, we can state whether rules of inference always hold or hold sometime in the future. That is, □(P → Q) means that it is always the case that we infer Q from P. Whereas ◇ (P → Q) will mean that sometime in the future we can infer Q from P, but it may not always be the case.

Note: There is a difference between NOT(bel Jon P) and (bel Jon NOT(P)). The latter clearly says that agent Jon believes P is False. The former keeps things more ambiguous: it says that it is not the case that agent Jon believes P is true. However, it does not say that agent Jon believes P is false. In other words, the former models uncertainty in agent Jon's beliefs about P.

Formalizing Commitments

We will focus on formalizing commitments to achievement goals. It is useful to note that *achievement goals differ from maintenance goals*. Achievement goals are one-shot goals. An agent flying a NASA probe may have an achievement goal of landing on the surface of planet Mercury. Maintenance goals are on-going goals. An agent flying a plane may have a maintenance goal to stay at an altitude of 35000 feet.

- Thus, if "x" is the agent involved, then achievement its goal implies:
  (Goal x (Eventually P)) AND (Bel x NOT(P))
  - *That is, the agent x currently believes the goal P to be false and wants the goal to be eventually true*

- Note that beliefs and goals must be consistent
  So for instance, if (Goal x (Eventually P)) then NOT(Bel x (Never P))
  - *This is because (Bel x (Never P))= (Bel x (NOT(Eventually P)) contradicts our goal (Goal x (Eventually P))*
  - *That is, if the Mars-rover has the goal to eventually dig for water-ice at location-1, it must not be the case that the Mars-Rover believes that digging for water-ice at location-1 is impossible.*

- We are now ready to formalize the notion of commitments. In the Cohen and Levesque logic, a commitment is formalized as a PGOAL. As seen below, a PGOAL builds on a Goal, but a

PGOAL is more than just a goal. Note that from now on, we will use the two terms (commitments and PGOALS) as synonyms:

- Commitment: If P a commitment, then don't drop P as a goal easily
  - Agent x will keep P, I.e., (goal x P) UNTIL
    - Agent x believes P is Achieved, i.e. (Bel x P)
    - Agent x believes P is Unachievable, i.e. (Bel x (Never P))
    - Agent x believes P is Irrelevant, i.e. (Bel x NOT(Q)): *Q is possibly the reason for P*

  There are no actual restrictions on what "Q" ought to be. We will have more to say about Q shortly.

- Thus, (PGOAL x P Q) is defined as:

$$
\begin{aligned}
&(Bel\ x\ NOT(P))\ AND\ (Goal\ x\ (Eventually\ P)) \\
&AND \\
&\quad (UNTIL\ ((Bel\ x\ P)\ OR \\
&\qquad\qquad ((Bel\ x\ (Never\ P))\ OR \\
&\qquad\qquad (Bel\ x\ NOT(Q))) \\
&\quad (Goal\ x\ (Eventually\ P)))
\end{aligned}
$$

The formula above introduces the notion of a PGOAL or persistent goal, which is a commitment. A commitment to a goal P relative to Q is one where an agent takes up the commitment just in case it believes P is not achieved and has the goal to eventually achieve P, and until such time as P is achieved, or unachievable (NEVER P) or somehow irrelevant (NOT Q), keeps the goal to achieve P. So agents will keep their commitments until they are achieved, unachievable or irrlevant. So agents will be committed to their goals "P", but not fanatically so. We can rewrite this more compactly given earlier notation:

$$
\begin{aligned}
&(Bel\ x\ \not\Diamond P)\ AND\ (Goal\ x\ \Diamond P) \\
&AND \\
&\quad (UNTIL\ ((Bel\ x\ P)\ OR \\
&\qquad\qquad ((Bel\ x\ \Box\ (\not\Diamond P))\ OR \\
&\qquad\qquad (Bel\ x\ \not\Diamond Q)) \\
&\quad (Goal\ x\ \Diamond P))
\end{aligned}
$$

Going back to our example of a Mars Rover looking for ice-water, the agent's commitment to finding water-ice will mean that it will not drop this goal until it has found water-ice, it believes that finding water-ice is unachievable or if somehow finding

water-ice has become irrelevant (because for some reason the current mission was cancelled). That is:

- Agent Mars-rover will keep (goal Mars-Rover Find-water-ice) UNTIL
  - The goal of Find-water-ice is believed to have been Achieved, i.e. (Bel Mars-Rover Find-water-ice)
  - The goal of Find-water-ice is believed to be Unachievable, i.e. (Bel Mars-Rover (Never Find-water-ice))
  - The goal of Find-water-ice is now Irrelevant, i.e. (Bel Mars-Rover NOT(Mission-to-find-water-on-Mars)): *The mission to find water was why the Mars-Rover had established the commitment to find water-ice.*

So we define (PGOAL Mars-Rover Find-Water-ice Mission-to-find-water-on-Mars) as:

(Bel Mars-Rover NOT(Find-Water-ice)) AND (Goal Mars-Rover (Eventually Find-Water-ice))
AND
(UNTIL ((Bel Mars-Rover Find-Water-ice) OR
((Bel Mars-Rover (Never Find-Water-ice)) OR
(Bel Mars-Rover NOT(Mission-to-find-water-on-Mars)))
(Goal Mars-Rover (Eventually Find-Water-ice)))

Note in this example, Q ("mission to find water on mars") is defined to be the reason why P ("Find-water-ice") came about.

Practical BDI agent architectures

Having gone through an informal description of B, D, I and then the BDI logic, let us now try to understand how these concepts are put into practice in practical BDI agent systems. There are three main concepts that we will introduce here: (i) Agent architectures; (ii) Reactive plans that are used to program agent architectures; (iii) Commitment strategies for practical agents.

Agent Architectures

An agent architecture refers to the fixed software organization or structure inside an agent. When we go from one domain to the next, we write different BDI programs, but the architecture remains fixed. Indeed, when one writes a BDI program, the architecture determines how it will actually get executed. There are a large number of agent architectures that have been defined, including several commercial agent architectures (we provide pointers to some of them at the end of this section). Many of them are based on the same common design principles, sketched out below. In particular, the architectures have the following components:

Belief store: Stores the agent's current beliefs. In some architectures, it is also called "working memory". Sensors provide inputs to this belief store in terms of new beliefs. One example belief shown in the figure is that "(Bel Mars-Rover Location-1-indicates-water-ice)", which denotes that the Mars-Rover agent believes that location-1 has some indication that it contains water-ice. This type of belief may come about because the sensor may have built in an infra-red test that is used to test the presence of water-ice.

Goal-store: Contains the agent's current goals. An example goal is shown: (Goal Mars-Rover *find-water-ice)*.

Plan-library: Rather than requiring planning from scratch, many architectures allow us to pre-define a set of plans and store in a plan library. These pre-defined set of plans can be typically characterized as "*reactive plans*". A plan library may contains many such reactive plans, possibly 100s or 1000s. In the example below, the plan library shows one example plan named "plan-1" to be used in fulfilling the goal of finding-water-ice. A reactive plan consists of the following components:

- *Preconditions:* This is the set of conditions that if matched by the current beliefs can cause a plan to get activated, i.e. a new desire is generated. The preconditions in this case are that there is an existing intention *: mission-find-water-on-Mars* and that there is a belief*: (bel Mars-Rover Location-z-indicates-water-ice)*. The presence of a variable "z" instead of a number in the precondition, as in *"Location-z"* is to indicate that the location is a variable, i.e. at the time of defining this plan, we do not know exactly what location we believe will indicate the presence of water-ice. At run-time, the Mars-Rover agent may come to believe in a specific location that that has water-ice, such as Location-1.

- *Body:* The plan body may define either an abstract plan or a concrete set of actions in the plan. In the case of the example shown, the body of the plan indicates that the Mars-Rover fulfill the mission to find water on Mars by looking for water-ice (find-water-ice). This body will in effect lead to a new goal to be generated. In our simple case, since this is the only goal, it will be adopted as the intention, as shown in the figure. In general though the agent may need to select among conflicting goals. Note also that since this is an abstract plan, it will not directly lead to action. Instead, this abstract plan, when adopted as an intention will require further detailed intentions to fulfill it, as shown in the figure below.

- *Termination conditions:* These define conditions under which the agent may reconsider its current intention. Given the PGOAL definition outlined above, a plan once adopted as an intention must be terminated only when it is achieved, or it is unachievable or it is irrelevant. Thus, we need to define conditions that make a plan achieved, unachievable or irrelevant. In the Mars-Rover example, we defined these conditions in the previous section as follows:

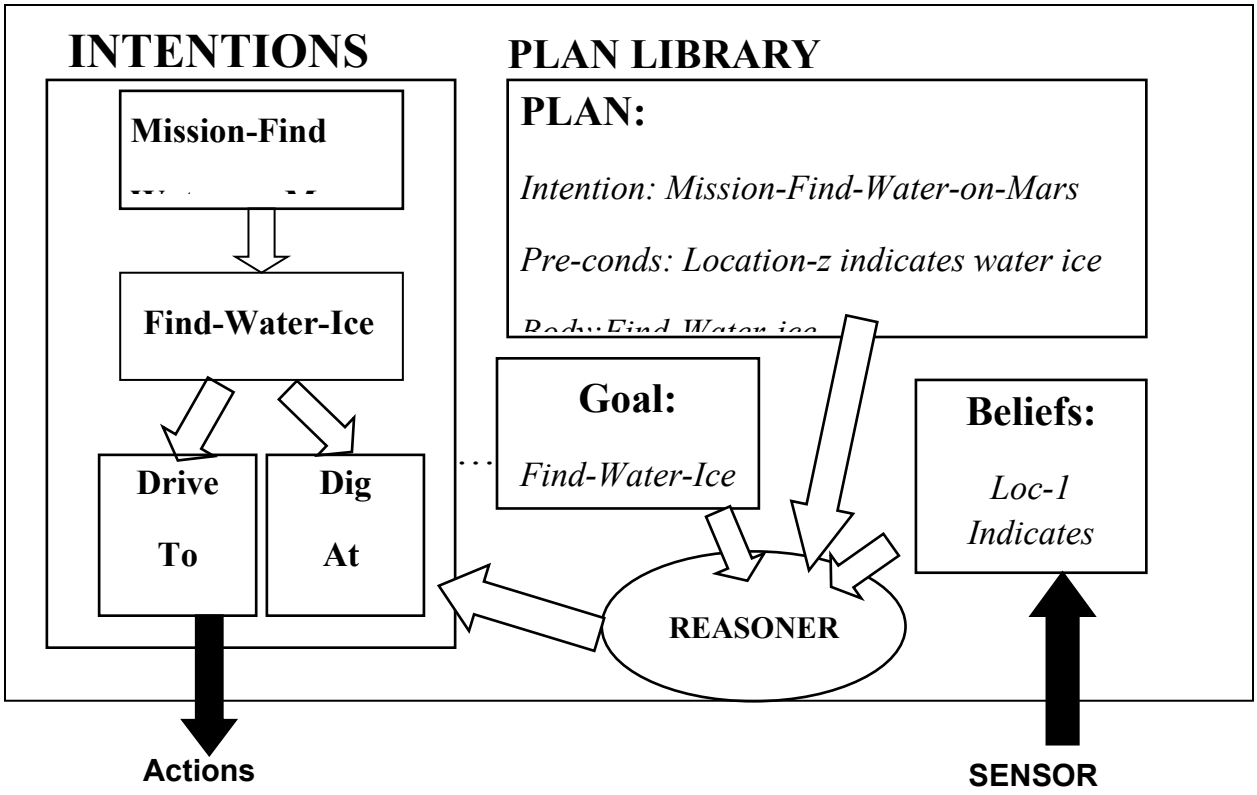
((Bel Mars-Rover Find-Water-ice) OR
((Bel Mars-Rover (Never Find-Water-ice)) OR
(Bel Mars-Rover NOT(Mission-to-find-water-on-Mars)))

When an agent obtains new beliefs, it will match these beliefs with these termination conditions to determine if the current intention should be terminated.

Intention hierarchy: Based on the agent's current beliefs, goals and its plan library, the agent elects to commit to some of the goals, which are its intentions. In this case, the agent intends to find-water-on-Mars. Given this intention, the agent can fulfill it by using plan-1 from the plan library. This plan requires that the agent commit to finding water ice. Finding water ice requires that the agent establish its intention to drive and then dig (two separate steps).

While in this case, we have a three stage hierarchy of intentions, in general, a whole hierarchy of intentions may come about. In particular, if driving to location-1 itself is not a primitive action, but requires several smaller steps to be taken, then the agent will take these smaller steps. Given a hierarchy of intentions, it is the most primitive of these intentions that will lead to action.

While an intention hierarchy is what we see in this case, this hierarchy only arises because the plan library has a corresponding to plan hierarchy. Indeed, typically, the plan library is made of a large plan hierarchy, possibly of 100s of plans. At any point, one part of the hierarchy may be in place as the intention hierarchy.



Commitment Strategies for Practical Agents

The decision process associated with appropriately reconsidering intentions is a key aspect of these agent architectures. Commitment strategies refer to the strategies agents adopt wrt reconsidering itentions. As discussed earlier, the agents should not be fanatically commited. On the other hand, the key point of commitments was to provide stability and to limit an agent's reasoning. Furthermore, intention reconsideration involves re-evaluating options --- an expensive process. Thus, the agent should not reconsider its intention willy-nilly given every change in its beliefs.

Termination conditions associated with reactive plans can define particular commitment strategies. Our earlier definition of PGOALS suggested one particular commitment strategy: that of continuing with the current commitment until it is achieved, unachievabile or irrelevant. Irrelevance is suggested by the "NOT(Q)" condition in our definition of PGOALs.

However, in general, there are no restrictions on Q. Indeed, practical agents may require that they reconsider their commitments even if not achieved or not proven unachievable, e.g. if there are higher priority goals that suddenly come up. Q could be defined to be other higher priority goals that can cause us to abandon P. For example, Q could be defined to be ("Absence of dust storms" AND "Mission-to-find-water-on-Mars"). So if either the mission is cancelled or if a

much feared Martian dust storm rises, we will abandon our goal to "Find-water-ice". In this case, we will require that irrelevance condition becomes:

(Bel Mars-Rover NOT((Mission-to-find-water-on-Mars) AND (Absence-of-dust-storms)))
i.e.

(Bel Mars-Rover NOT(Mission-to-find-water-on-Mars)) OR
(Bel Mars-Rover NOT(Absence-of-dust-storms)))

And thus the termination conditions become:

((Bel Mars-Rover Find-Water-ice) OR
((Bel Mars-Rover (Never Find-Water-ice)) OR
((Bel Mars-Rover NOT(Mission-to-find-water-on-Mars)) OR
((Bel Mars-Rover NOT(Absence-of-dust-storms))))

Given such termination conditions, typically associated with a commitment is a two-stage process of deciding whether to reconsider it:

1. *Filtering:* Given a new belief or desire, this process quickly decides whether it is worth reconsidering the current commitment(s). This process may filter things out, i.e. it may dismiss the new belief or desire as not worthy of requiring reconsideration of current commitments. Or it may filter things in, i.e. it may suggest that the current belief or desire requires that current commitment be reconsidered. This process may not always give the right answer, but the key point is that it is a computationally cheap process.
2. *Reconsideration*: Once past the filtering stage, a new belief or desire requires that the agent reconsider and re-deliberate the current process. This is a computationally expensive process. It may lead to actually throwing away the current commitment and taking on a new one, or it may actually end up re-establishing the current commitment.

To understand this two-stage process, consider the following example. Our Mars-Rover has committed to digging for water-ice at location-1. Furthermore, it has planned a path to location-1, and it has committed to this plan. We will consider two different events and understand their impact on the two stage process outlined above:

1. *Filtering out:* As the Mars-Rover gets close to location-1, it sees another location, location-3 far in the distance. It may thus obtain a new belief that there is a location, location-3, which is also a promising location to dig for water-ice. The filtering process dismisses location-3 because the Mars-Rover is already committed to digging for water-ice at location-1. So the Mars-Rover will not reconsider its commitment to dig for water-ice at location-1. There are two situations possible:
    a. *Filtering out was right:* Had the Mars-Rover actually reconsidered its commitment, it would have ultimately continued with its commitment to location-1 anyway, because location-3 was too far away.

b. *Filtering out was wrong:* Had the Mars-Rover actually reconsidered its commitment, it would have changed its commitment because closer scrutiny of geological factors, and distance to home-base etc would have led the Mars-Rover to decide in favor of location-3 and abandon digging at location-1.

2. *Filtering in:* As the Mars-Rover gets close to location-1, it sees a dust-storm approaching. The filtering process requires that the current commitment to dig at location-1 be reconsidered. Again there are two situations possible:
   a. *Filtering in was right:* Upon reconsideration, the Mars-Rover adopts a new commitment to safely drive back to home-base, protecting itself from the Martian dust-storm.
   b. *Filtering in was wrong:* Upon reconsideration, the Mars-Rover re-establishes its commitment to dig at location-1 because the dust-storm will pass by harmlessly.

Of course, we wish to minimize situations 1b and 2b. However, the point is that we may not have perfection. The agent may occassionally stumble into cases 1b and 2b because globally reasoning perfectly about each situation may not be viable for a resource-bounded agent.

Commercial BDI Agent Software Architectures

We outline below three commercial BDI agent software architectures that are used around the world to support products. This is only a sampling of available architectures on the market.

JACK Agent Software:
- http://www.agent-software.com

Agentis software:
- http://www.agentis.com

Soartech:
http://www.soartech.com

Example Problems:

: How will you write this in our logic: David knows Paul believes that the robot will eventually get ice.

(BEL Paul ◇ (Robot-gets-ice)) AND
(BEL David (BEL Paul ◇ (Robot-gets-ice)))

State True or False: If “x” is the agent involved, then the fact that it has an achievement goal can be written as:

(Goal x □P) AND (Bel x ØP)

FALSE

References:

: Martha Pollack, 1992. "The uses of plans" In Artificial Intelligence, 57(1):43-69
Levesque, H. J., Cohen, P. R. & Nunes, J. T. H. (1990) On acting together. Proceedings of AAAI-90.
Georgeff, M., Pell, B., Pollack, M., Tambe, M. and Wooldridge, M., 1999 The Belief-Desire-Intention model of agency Proceedings of Agents, Theories, Architectures and Languages (ATAL’99)