

## Intelligent Autonomous Agents ICS 606 / EE 606 Fall 2011

Nancy E. Reed

nreed@hawaii.edu

## Lecture #4B Practical Reasoning Agents

- Simple agent control loop
- Deliberation
- Commitment
- Commitment strategies
- Intention reconsideration
- BDI theory (beliefs, desires, intentions)
- Temporal logics

## Implementing Practical Reasoning Agents

- A simple practical reasoning agent:

Agent Control Loop Version 1

1. **while** true
2. **get next percept**  $p$ ; observe the world;
3.  $B := brf(B,p)$ ; update internal world model;
4.  $I := deliberate(B)$ ; deliberate about what intention to achieve next;
5.  $\pi := plan(B,I)$ ; use means-ends reasoning to get a plan for the intention;
6. **execute**( $\pi$ ); execute the plan
7. **end while**

- Problem: **deliberation** and **means-ends reasoning** processes are not instantaneous. They have **a time cost**.
- Suppose the agent *starts deliberating* at  $t_0$ , *begins means-ends reasoning* at  $t_1$ , and *begins executing the plan* at time  $t_2$ .

- Time to deliberate is

$$t_{deliberate} = t_1 - t_0$$

- and time for means-ends reasoning is

$$t_{me} = t_2 - t_1$$

- Further suppose that deliberation is **optimal** in that if it selects some intention to achieve, then this is the best thing for the agent. (Maximizes expected utility.)
- So at time  $t_1$ , the agent has selected an intention to achieve that **would have been optimal if it had been achieved at  $t_0$** .  
But **unless  $t_{deliberate}$  is vanishingly small**, then the agent runs the *risk that the intention selected is no longer optimal* by the time the agent has fixed upon it.
- This is **calculative rationality**.
- Deliberation is only half of the problem (**what**): the agent still has to determine **how** to achieve the intention.

- So, this agent will have **overall optimal behavior in the following circumstances**:

1. When *deliberation and means-ends reasoning take a vanishingly small amount of time*; or
2. When *the world is guaranteed to remain static while the agent is deliberating* and performing means-ends reasoning, so that the assumptions upon which the choice of intention to achieve and plan to achieve the intention remain valid until the agent has completed deliberation and means-ends reasoning; or
3. When *an intention that is optimal when achieved at time  $t_0$  (the time at which the world is observed) is guaranteed to remain optimal until time  $t_2$  (the time at which the agent has found a course of action to achieve the intention)*.

## Practical Reasoning Agents: A More Formal Algorithm (v. 2)

Agent Control Loop Version 2

1.  $B := B_0$ ; /\* initial beliefs \*/
2. **while true do**
3.   **get next percept**  $\rho$ ; observe the world;
4.    $B := brf(B, \rho)$ ; update internal world model;
5.    $I := deliberate(B)$ ; deliberate about what intention to achieve next;
6.    $\pi := plan(B, I)$ ; use means-ends reasoning to get a plan for the intention;
7.   **execute**( $\pi$ ); execute the plan
8. **end while**

## Utility-Based Agent Program (AIMA)

```

function Utility-Agent (percept)
    current world state
static: rules, a set of condition-action rules

    state      <- UPDATE-STATE (state, percept)
do { rule  <- RULE-MATCH (state, rules)
      action <- RULE-ACTION [rule]
      state  <- UPDATE-STATE (state, action)
    } while ((not (goals? <- state)) || (not (good quality?)))

return action
  
```

## Logical Agent Action Function

```

function action (DB : D) : A
begin
  for each  $a \in A$  do           ; each action
    if DB can prove Do( $a$ ) then
      return  $a$                  ; return
    end-if
  end-for
  for each  $a \in A$  do
    if DB not inconsistent with NOT Do( $a$ ) then
      return  $a$ 
    end-if
  end-for
  return null
end function action
  
```

## How Does an Agent Deliberate?

1. understand what *options* are available
2. *choose between them*, and
3. *commit* to some options

Chosen options are then **intentions**

## The Deliberate Function

Can be decomposed into two distinct functional components:

1. **option generation** – generate a set of possible alternatives, using an *options* function
  - Takes the agent's
    - current beliefs and current intentions,
    - and from them
  - Determines a **set of options** (= *desires*)
2. **Filtering** - choose between competing alternatives, using a *filter* function
  - Set of **desires** generated above are given to *filter* and from them determines which to pursue using **commitments**
  - **Commit** to achieving the selected *desires*

## Agent Control Loop Version 3

```

1
2  $B := B_0$ ; /* Initial beliefs */
3  $I = I_0$ 
4 while true do
5   get next percept  $\rho$  observe the world;
6    $B := brf(B, \rho)$ ; update internal world;
7    $D := options(B, I)$ ; generate options
8    $I := filter(B, D, I)$ ; deliberate about intention
9    $\pi := plan(B, I)$ ; means-ends plan for I;
10  execute( $\pi$ ); execute the plan
11 end while
  
```

### Commitment Strategy Tale (1)

“Some time in the not-so-distant future, you are having trouble with your **new household robot**.

You say “*Willie, bring me a beer.*”

The robot replies “OK boss.”

Twenty minutes later, you screech “*Willie, why didn't you bring me that beer?*”

It answers “Well, I intended to get you the beer, but I decided to do something else.”

Miffed, you send the wise guy back to the manufacturer, complaining about a **lack of commitment**.

### Commitment (2)

After retrofitting, Willie is returned, marked “Model C: The Committed Assistant.”

Again, you ask Willie to bring you a beer.

Again, it accedes, replying “Sure thing.”

Then you ask: “*What kind of beer did you buy?*”

It answers: “Genessee.”

You say “*Never mind.*”

One minute later, Willie trundles over with a Genessee in its gripper.

This time, you angrily return Willie for **over commitment**.

### Commitment (3)

After still more tinkering, the manufacturer sends Willie back, promising no more problems with its commitments.

So, being a somewhat trusting customer, you accept the rascal back into your household, but as a test, you ask it to bring you your last beer.

Willie again accedes, saying “Yes, Sir.”

(Its attitude problem seems to have been fixed.)

### Commitment (4)

The robot gets the beer and starts towards you. As it approaches, it lifts its arm, wheels around, deliberately smashes the bottle, and trundles off.

Back at the plant, when interrogated by customer service as to why it had abandoned its commitments, the robot replies that **according to its specifications, it kept its commitments as long as required** — commitments must be dropped when fulfilled or impossible to achieve.

**By smashing the bottle, the commitment became unachievable.**”

### Commitment Strategies

- The following are commonly discussed in the literature of rational agents:

- *Blind commitment*

A blindly committed agent will continue to maintain an intention until it believes the intention has actually been achieved. Blind commitment is also sometimes referred to as *fanatical* commitment.

- *Single-minded commitment*

A single-minded agent will continue to maintain an intention until it believes that either the intention has been achieved, or else that it is no longer possible to achieve the intention.

- *Open-minded commitment*

An open-minded agent will maintain an intention as long as it is still believed possible.

### Commitment Strategies

- An agent has commitment both to **ends** (i.e., the wishes to bring about), **and means** (i.e., the mechanism via which the agent wishes to achieve the state of affairs)
- Currently, our agent control loop is overcommitted, both to means and ends  
Modification: **replan** if ever a plan goes wrong

```

Agent Control Loop Version 4
1
2 B := B0; /* initial beliefs */
3 I := I0
4 while true do
5   get next percept p;
6   B := belief(p);
7   O := options(B,I);
8   I := filter(O,I);
9   x := plan(O);
10  while not empty(x) do
11    a := head(x);
12    execute(a);
13    x := tail(x);
14    get next percept p;
15    B := belief(p);
16    if not succeed(x,I,B) then
17      x := plan(O);
18    End if
19  End while
20 end while
    
```

- ### Still Overcommitted To Intentions
- Never stops to consider whether or not its intentions are appropriate
  - Modification:** stop to determine whether intentions have succeeded or whether they are impossible:  
*(Single-minded commitment)*

```

Agent Control Loop version 5
1
2 B := B0; /* initial beliefs */
3 I := I0
4 while true do
5   get next percept p;
6   B := belief(p);
7   O := options(B,I);
8   I := filter(O,I);
9   x := plan(O);
10  while not empty(x)
11    or succeeded(I,B)
12    or impossible(I,B) do
13    a := head(x);
14    execute(a);
15    x := tail(x);
16    get next percept p;
17    B := belief(p);
18    if not succeed(x,I,B) then
19      x := plan(O);
20    End if
21  End while
22 end while
    
```

- ### Intention Reconsideration
- Our agent gets to reconsider its intentions once every time around the outer control loop, i.e., when:
    - it has completely executed a plan to achieve its current intentions; or
    - it believes it has achieved its current intentions; or
    - it believes its current intentions are no longer possible.
  - This is limited in the way that it permits an agent to **reconsider** its intentions
  - Modification:** Reconsider intentions after executing every action

```

Agent Control Loop Version 6
1
2 B := B0; /* initial beliefs */
3 I := I0
4 while true do
5   get next percept p;
6   B := belief(p);
7   O := options(B,I);
8   I := filter(O,I);
9   x := plan(O);
10  while not empty(x)
11    or succeeded(I,B)
12    or impossible(I,B) do
13    a := head(x);
14    execute(a);
15    x := tail(x);
16    get next percept p;
17    B := belief(p);
18    O := options(B,I);
19    I := filter(O,I);
20    if not succeed(x,I,B) then
21      x := plan(O);
22    End if
23  End while
24 end while
    
```

- ### Intention Reconsideration
- But intention reconsideration is **costly!**  
A dilemma:
    - an agent that does not stop to reconsider its intentions sufficiently often *will continue attempting to achieve its intentions even after it is clear that they cannot be achieved, or that there is no longer any reason for achieving them*
    - an agent that **constantly** reconsiders its attentions may spend insufficient time actually working to achieve them, and hence runs the risk of never actually achieving them
  - Solution: incorporate an explicit **meta-level control** component, that decides whether or not to reconsider

```

Agent Control Loop Version 7
1
2 if  $\gamma > \gamma_0$  then  $\gamma := \gamma_0$ 
3 if  $\gamma > \gamma_0$ 
4 then  $\gamma := \gamma_0$ 
5
6 if  $\gamma > \gamma_0$ 
7 then  $\gamma := \gamma_0$ 
8
9 if  $\gamma > \gamma_0$ 
10 then  $\gamma := \gamma_0$ 
11
12 if  $\gamma > \gamma_0$ 
13 then  $\gamma := \gamma_0$ 
14
15 if  $\gamma > \gamma_0$ 
16 then  $\gamma := \gamma_0$ 
17
18 if  $\gamma > \gamma_0$ 
19 then  $\gamma := \gamma_0$ 
20
21 if  $\gamma > \gamma_0$ 
22 then  $\gamma := \gamma_0$ 
23
24 if  $\gamma > \gamma_0$ 
25 then  $\gamma := \gamma_0$ 
26
27 if  $\gamma > \gamma_0$ 
28 then  $\gamma := \gamma_0$ 
29
30 if  $\gamma > \gamma_0$ 
31 then  $\gamma := \gamma_0$ 
32
33 if  $\gamma > \gamma_0$ 
34 then  $\gamma := \gamma_0$ 
35
36 if  $\gamma > \gamma_0$ 
37 then  $\gamma := \gamma_0$ 
38
39 if  $\gamma > \gamma_0$ 
40 then  $\gamma := \gamma_0$ 
41
42 if  $\gamma > \gamma_0$ 
43 then  $\gamma := \gamma_0$ 
44
45 if  $\gamma > \gamma_0$ 
46 then  $\gamma := \gamma_0$ 
47
48 if  $\gamma > \gamma_0$ 
49 then  $\gamma := \gamma_0$ 
50
51 if  $\gamma > \gamma_0$ 
52 then  $\gamma := \gamma_0$ 
53
54 if  $\gamma > \gamma_0$ 
55 then  $\gamma := \gamma_0$ 
56
57 if  $\gamma > \gamma_0$ 
58 then  $\gamma := \gamma_0$ 
59
60 if  $\gamma > \gamma_0$ 
61 then  $\gamma := \gamma_0$ 
62
63 if  $\gamma > \gamma_0$ 
64 then  $\gamma := \gamma_0$ 
65
66 if  $\gamma > \gamma_0$ 
67 then  $\gamma := \gamma_0$ 
68
69 if  $\gamma > \gamma_0$ 
70 then  $\gamma := \gamma_0$ 
71
72 if  $\gamma > \gamma_0$ 
73 then  $\gamma := \gamma_0$ 
74
75 if  $\gamma > \gamma_0$ 
76 then  $\gamma := \gamma_0$ 
77
78 if  $\gamma > \gamma_0$ 
79 then  $\gamma := \gamma_0$ 
80
81 if  $\gamma > \gamma_0$ 
82 then  $\gamma := \gamma_0$ 
83
84 if  $\gamma > \gamma_0$ 
85 then  $\gamma := \gamma_0$ 
86
87 if  $\gamma > \gamma_0$ 
88 then  $\gamma := \gamma_0$ 
89
90 if  $\gamma > \gamma_0$ 
91 then  $\gamma := \gamma_0$ 
92
93 if  $\gamma > \gamma_0$ 
94 then  $\gamma := \gamma_0$ 
95
96 if  $\gamma > \gamma_0$ 
97 then  $\gamma := \gamma_0$ 
98
99 if  $\gamma > \gamma_0$ 
100 then  $\gamma := \gamma_0$ 

```

### Possible Interactions

- The possible interactions between meta-level control and deliberation are:

Situation number	Chose to deliberate?	Changed intentions?	Would have changed intentions?	reconsider(...) optimal?
1	No	—	No	Yes
2	No	—	Yes	No
3	Yes	No	—	No
4	Yes	Yes	—	Yes

### Intention Reconsideration

- In situation (1), the agent did not choose to deliberate, and as consequence, did not choose to change intentions. Moreover, if it had chosen to deliberate, it would not have changed intentions. In this situation, the *reconsider(...)* function is behaving optimally.
- In situation (2), the agent did not choose to deliberate, but if it had done so, it would have changed intentions. In this situation, the *reconsider(...)* function is not behaving optimally.
- In situation (3), the agent chose to deliberate, but did not change intentions. In this situation, the *reconsider(...)* function is not behaving optimally.
- In situation (4), the agent chose to deliberate, and did change intentions. In this situation, the *reconsider(...)* function is behaving optimally.
- An important assumption: cost of *reconsider(...)* is much less than the cost of the deliberation process itself.

### Optimal Intention Reconsideration

- Kinny and Georgeff's experimentally investigated the effectiveness of intention reconsideration strategies
- Two different types of reconsideration strategy were used:
  - bold** agents never pause to reconsider intentions, and
  - cautious** agents stop to reconsider after every action
- Dynamism** in the environment is represented by the *rate of world change,  $\gamma$*

$\gamma = \text{lambda}$

### Optimal Intention Reconsideration

Results (not surprising):

- If  $\gamma$  is low (i.e., the environment does not change quickly), then **bold agents do well** compared to cautious ones. This is because cautious ones waste time reconsidering their commitments while bold agents are busy working towards — and achieving — their intentions.
- If  $\gamma$  is high (i.e., the environment changes frequently), then **cautious agents tend to outperform bold agents**. This is because they are able to recognize when intentions are doomed, and also to take advantage of serendipitous situations and new opportunities when they arise.

### BDI Theory and Practice

- We now consider the **semantics** of BDI architectures: to what extent does a BDI agent satisfy a **theory of agency**
- In order to give a semantics to BDI architectures, Rao & Georgeff have developed **BDI logics**: non-classical logics with modal connectives for representing beliefs, desires, and intentions
- The 'basic BDI logic' of Rao and Georgeff is a quantified extension of the *expressive branching time logic CTL\**
- Underlying semantic structure is a **labeled branching time** framework

31

### BDI Logic

- From classical logic:  $\wedge, \vee, \neg, \dots$
- The CTL\* *path quantifiers*:
  - $A\phi$  'on all paths,  $\phi$ '
  - $E\phi$  'on some paths,  $\phi$ '
- The BDI connectives:
  - $(Bel\ i\ \phi)$  *i believes*  $\phi$
  - $(Des\ i\ \phi)$  *i desires*  $\phi$
  - $(Int\ i\ \phi)$  *i intends*  $\phi$

$\phi = \text{phi}$

32

### BDI Logic

- Semantics of BDI components are given via accessibility relations over 'worlds', where each world is itself a branching time structure
- Properties required of accessibility relations ensure belief logic KD45, desire logic KD, intention logic KD (Plus interrelationships. . .)

33

### Axioms of KD45

- (1)  $Bel(p \rightarrow q) \rightarrow (Bel\ p \rightarrow Bel\ q)$  (K)  
If you believe that p implies q then if you believe p then you believe q
- (2)  $Bel\ p \rightarrow \neg Bel\ \neg p$  (D)  
This is the consistency axiom, stating that if you believe p then you do not believe that p is false
- (3)  $Bel\ p \rightarrow Bel\ Bel\ p$  (4)  
If you believe p then you believe that you believe p
- (4)  $\neg Bel\ p \rightarrow Bel\ \neg Bel\ p$  (5)  
If you do not believe p then you believe that you do not believe that p is true

34

### Axioms of KD45

It also entails the two inference rules of *modus ponens* and necessitation:

- (5) if p, and  $p \rightarrow q$ , then q (MP)
- (6) if p is a theorem of KD45 then so is  $Bel\ p$  (Nec)

This last rule just states that you believe all theorems implied by the logic

35

### CTL Temporal Logic (from David Garlan's slides, CMU)

- Branching time logic views a computation as a (possibly infinite) tree or DAG of states connected by atomic events
- At each state the outgoing arcs represent the actions leading to the possible next states in some execution
- Example:  
 $P = (a \rightarrow P) \parallel (b \rightarrow P)$

36

### CTL\* Notation

- Variant of branching time logic that we look at is called CTL\*, for **Computational Tree Logic (star)**
- In this logic
  - A = "for every path"
  - E = "there exists a path"
  - G = "globally" (similar to  $\square$ )
  - F = "future" (similar to  $\diamond$ )

### Paths versus States 37

- A and E refer to paths
  - A requires that all paths have some property
  - E requires that **at least some path** has the property
- G and F refer to states on a path
  - G requires that **all states on the given path** have some property
  - F requires that **at least one state on the path** has the property

### CTL\* Examples 38

- AG p
  - For every computation (i.e., path from the root), in every state, p is true
  - Hence, means the same as  $\forall p$
- EG p
  - There exists a computation (path) for which p is always true

### CTL\* Examples continued 39

- AF p
  - For every path, **eventually state p is true**
  - Hence, means the same as  $\diamond p$
  - Therefore, p is *inevitable*
- EF p
  - There is **some path** for which p is eventually true
  - I.e., p is “reachable”
  - Therefore, p will hold *potentially*

### Some Useful CTL\* Equalities 40

- From linear temporal logic:
  - $\Box P \equiv \sim \diamond \sim P$
  - $\diamond P \equiv \sim \Box \sim P$
- In CTL\* we can say:
  - $AG p \equiv \sim EF \sim p$
  - $EG p \equiv \sim AF \sim p$
- We can rewrite  $AG p \equiv \sim EF \sim p$  as  $EF p \equiv \sim AG \sim p$

### BDI Logic 41

- Let us now look at some possible axioms of BDI logic, and see to what extent the BDI architecture could be said to satisfy these axioms
- In what follows, let
  - $\alpha$  be an *O-formula*, i.e., one which contains no positive occurrences of A
  - $\phi$  be an arbitrary formula

### BDI Logic 42

- *Belief goal compatibility:*
  - $(Des \alpha) \rightarrow (Bel \alpha)$
  - States that if the agent has a goal to optionally achieve something, this thing must be an option. This axiom is operationalized in the function *options*: an option should not be produced if it is not believed possible.
- *Goal-intention compatibility:*
  - $(Int \alpha) \rightarrow (Des \alpha)$
  - States that having an intention to optionally achieve something implies having it as a goal (i.e., there are no intentions that are not goals). Operationalized in the *deliberate* function.

## BDI Logic

43

- *Volitional commitment:*

$$(\text{Int } \textit{does}(a)) \rightarrow \textit{does}(a)$$

If you intend to perform some action  $a$  next, then you do  $a$  next.

Operationalized in the *execute* function.

- *Awareness of goals & intentions:*

$$(\text{Des } \phi) \rightarrow (\text{Bel } (\text{Des } \phi))$$

$$(\text{Int } \phi) \rightarrow (\text{Bel } (\text{Int } \phi))$$

Requires that new intentions and goals be posted as events.

## BDI Logic

44

- *No unconscious actions:*

$$\textit{done}(a) \rightarrow \text{Bel}(\textit{done}(a))$$

If an agent does some action, then it is aware that it has done the action.

Operationalized in the *execute* function.

A stronger requirement would be for the success or failure of the action to be posted.

- *No infinite deferral:*

$$(\text{Int } \phi) \rightarrow A \diamond (\neg(\text{Int } \phi))$$

An agent will eventually either act for an intention, or else drop it.

## Summary

45

- Simple agent control loop
- Deliberation
- Commitment
- Commitment strategies
- Intention reconsideration
- BDI theory (beliefs, desires, intentions)
- Temporal logics
- Reference
  - Wooldridge MAS, Ch. 4