Incorporating Learning in BDI Agents

Stéphane Airiau\textsuperscript{1}  Lin Padgham \textsuperscript{2}  Sebastian Sardina \textsuperscript{2}  Sandip Sen \textsuperscript{3}

\textsuperscript{1}ILLC - University of Amsterdam
\textsuperscript{2}RMIT University, Melbourne, Australia
\textsuperscript{3}University of Tulsa, OK, USA

ALAMS+ALAg 2008
Workshop at AAMAS
Estoril, Portugal, 2008
Belief, Desire, Intentions

- **Belief:** knowledge about the world and its own internal state
- **Desires (or goals):** what the agent has decided to work towards achieving
- **Intentions:** how the agent has decided to tackle these goals.
- **No planning from first principles:** agents use a plan library (library of partially instantiated plans to be used to achieve the goals)

Practical reasoning agents: quickly reason and react to asynchronous events.
**Definition (Plan)**

\[ e : \psi \leftarrow P \] where

- \( e \) is an event that triggers the plan
- \( \psi \) is the context for which the plan can be applied
- \( P \) is the body of the plan (succession of actions and/or subgoals)

**Goal-Plan tree**

- \( P_i \): plan
- \( G_i \): goals
- \( SG_i \): subgoals

**Failure recovery**

when a step fails, causing a plan to fail, an alternative plan is tried.

ex: if both \( P_1 \) and \( P_2 \) are applicable, when \( P_4 \) fails, \( P_2 \) can be tried
**BDI execution algorithm**

1. Take the next event (internal/external)
2. Modify any goals, beliefs, intentions (new event may cause an update of the belief, causing a modification of the goals and/or intentions)
3. Select an applicable plan to respond to this event
4. Place this plan in the intention base;
5. Take the next step on a selected intention (may execute an action, generate a new event)
BDI execution algorithm

1. Take the next event (internal/external)
2. Modify any goals, beliefs, intentions (new event may cause an update of the belief, causing a modification of the goals and/or intentions)
3. Select an applicable plan to respond to this event
4. Place this plan in the intention base;
5. Take the next step on a selected intention (may execute an action, generate a new event)

BDI agents are well suited for complex applications with soft real-time reasoning and control requirements.

Airiau, Padgham, Sardina, Sen
Incorporating Learning in BDI Agent
**BDI execution algorithm**

1. Take the next event (internal/external)
2. Modify any goals, beliefs, intentions (new event may cause an update of the belief, causing a modification of the goals and/or intentions)
3. **Select an applicable plan to respond to this event**
4. Place this plan in the intention base;
5. Take the next step on a selected intention (may execute an action, generate a new event)

**BDI agents are well suited for complex applications with soft real-time reasoning and control requirements.**
Issues

- BDI agents lack learning capabilities to modify their behavior (e.g. in case of frequent failures)
- Plans and context conditions are programmed by a user. In a complex environment, context conditions may be hard to capture precisely
  - too loose: plan is applicable when it is not → failures
  - too tight: plan is not applicable when it actually is → a goal may not appear achievable when it is
Issues

- BDI agents lack learning capabilities to modify their behavior (e.g. in case of frequent failures)
- Plans and context conditions are programmed by a user. In a complex environment, context conditions may be hard to capture precisely
  - too loose: plan is applicable when it is not → failures
  - too tight: plan is not applicable when it actually is → a goal may not appear achievable when it is

Research goal

Add learning capabilities to adapt and precise context conditions of plans
Issues

- BDI agents lack learning capabilities to modify their behavior (e.g. in case of frequent failures)
- Plans and context conditions are programmed by a user. In a complex environment, context conditions may be hard to capture precisely
  - too loose: plan is applicable when it is not → failures
  - too tight: plan is not applicable when it actually is
    → a goal may not appear achievable when it is

Research goal

Add learning capabilities to adapt and precise context conditions of plans

A first step

Use a decision tree (DT) in addition to the context condition
Each plan has a decision tree telling whether it is applicable
Example of a DT

the environment is described by three boolean attributes $a$, $b$ and $c$

Context condition converted from the decision tree:
$$(a \land b) \lor (a \land \neg b \land c) \lor (a \land \neg b \land \neg c).$$
Learning Issues

- When to collect data?
  In case of failure,
  - did the failure occur because the current plan was not applicable?
  - did it fail because other plans below were mistakenly chosen?

- When to start to use the decision tree?
Learning Issues

- When to collect data?
  - In case of failure,
    - did the failure occur because the current plan was not applicable? → Correct data
    - did it fail because other plans below were mistakenly chosen?

- When to start to use the decision tree?
Learning Issues

- When to collect data?
  In case of failure,
  - did the failure occur because the current plan was not applicable? $\rightarrow$ Correct data
  - did it fail because other plans below were mistakenly chosen? $\rightarrow$ Incorrect data

$P_i$: plan
$G_i$: goals
$SG_i$: sub-goals

- When to start to use the decision tree?
Initial Experiments

Three mechanisms for plan selection

**CL:** all trees are learnt at the same time, all data is used

**BU:** Bottom Up learning: DT higher in the hierarchy wait for DT below to be formed

**PS:** Probabilistic selection: plans are selected according to the frequency of success provided by the decision tree

Use the DT
- after \(k\) instances have been observed for CL and BU (\(k\) large),
- after few instances for PS (5-10 to have an initial DT).
Initial Results

Setup: 17 plans, world state is defined by six boolean attributes, depth of goal-plan tree is 4. All context conditions are set to true. $k = 100$

Non Deterministic World: action may fail with a probability of 0.1

![Graph showing frequency of success vs. number of instances for different probabilistic selection methods: BU, CL. The graph illustrates the performance of the methods over a range of instances, with BU and CL showing distinct trends.]
Conclusion

- Though theoretically, need to wait for DTs below to be accurate before collecting data for DT higher, DTs handle the spurious data as noise
- Using PS, the context conditions are learnt faster and are accurate

Future Work

- Test on larger goal-plan trees
- Try better criteria for starting using the DTs
Contacts

Stéphane Airiau: stephane@illc.uva.nl

Lin Padgham: lin.padgham@rmit.edu.au

Sebastian Sardina: sebastian.sardina@rmit.edu.au

Sandip Sen: sandip@utulsa.edu