Motivations and declarative goals as cornerstones of autonomy

Felipe Meneguzzi

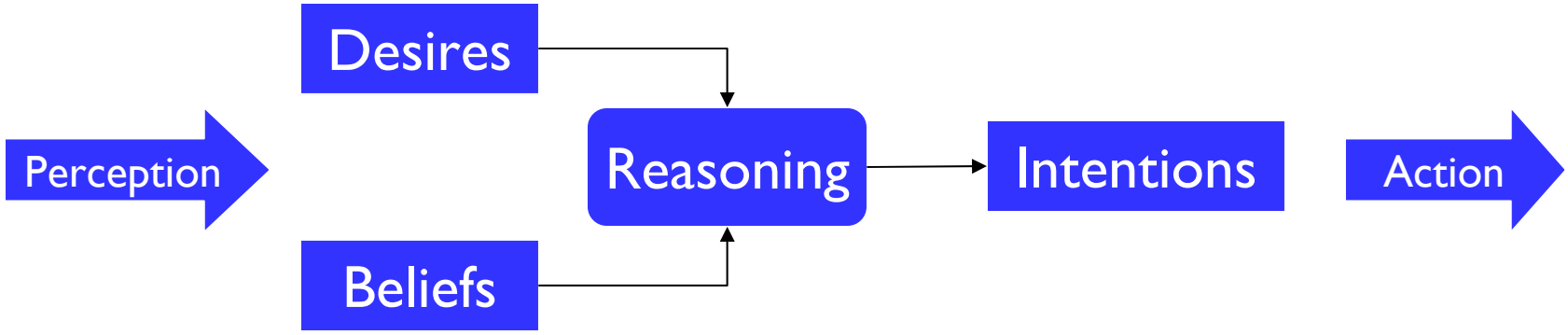
University of Southampton

Outline

- Background: Goal types
- AgentSpeak and Planning AgentSpeak
- Example: Production Cell
- Issues: Execution and Control
- Related Work: Motivations and Decl. Goals
- Future Work: Motivations and Planning

Background

- BDI Agents → Procedural vs. Declarative
- Procedural → Efficient, yet inflexible
  - Predefined encapsulated behaviours
  - Designer must foresee relevant plans
- Declarative →Expressive, not trivial
  - Desired world states
  - Requires a more complex reasoning mechanism



AgentSpeak

- Based on *Procedural* Reasoning System
- Agent is described in terms of a plan library
- Plans are defined by:
  - A *trigger* condition and a *context*
  - A *body* containing the plan itself
- *Events* drive the adoption of plans

Example AgentSpeak Plan

- Event is generated
- First plan with a matching trigger condition and a valid context is adopted
- If the plan fails to finish, the goal has failed

```
+!trigger : context
 <-!subgoal1; //calls plan
   action1;  //does smth
   !subgoal2;
   action2;
   +belief1; //updates bel.
   -belief2.
```

Planning AgentSpeak

- Prototype developed using Jason
- Allows declarative goals to drive plan adoption
  - Goals are satisfied using planning
- Allows dynamic plan generation
  - Supported by a planning component

```
+!des(Goals) : true
  ← plan(Goals).
```

- Where Goals is a list representing a conjunction of goals

Issues of Execution

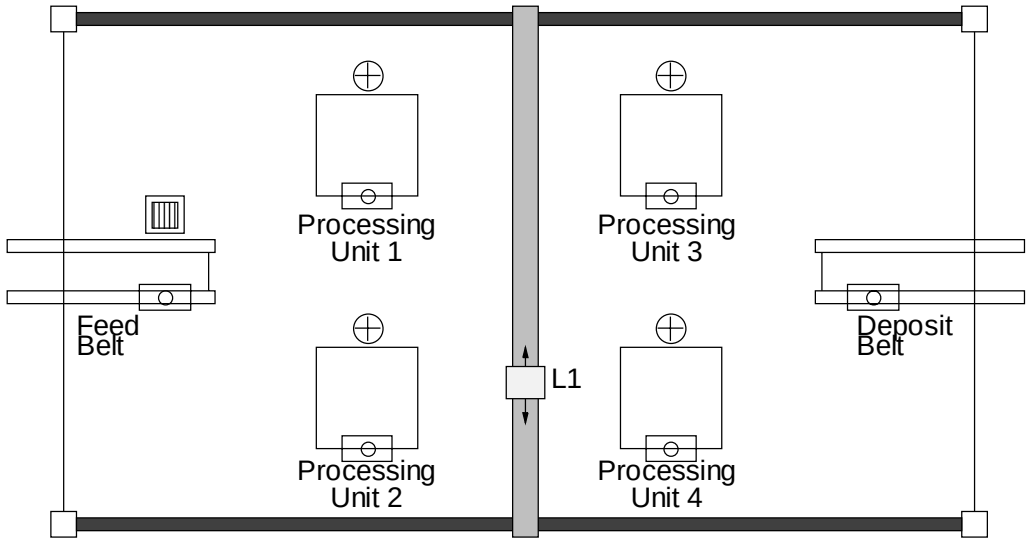
- Failure handling
  - Not integral to procedural plans
- Lookahead
  - Agent selects and executes plans without looking at the outcome
  - Bottlenecks may cause unnecessary failures
- Description size
  - Increases significantly to allow flexibility

Issues of Control

- Choosing dynamic behaviours over predefined ones
  - Currently, these “decisions” are hard-coded
- Controlling the amount of time spent on planning
  - Placing reasonable bounds on planning effort

Example

- Production Cell
  - 4 Processing Units
  - Parts come in from the Feed Belt
  - Must be processed by certain processing units



Declarative vs Procedural

```
+!finish(Block) : Block = block1
<- +des([processed(Block,procUnit1),
     processed(Block,procUnit2),
     processed(Block,procUnit3), finished(Block)]).

+!finish(Block) : Block = block2
<- +des([processed(Block,procUnit2),
     processed(Block,procUnit4),
     finished(Block)]).

+!finish(Block) : Block = block3
<- +des([processed(Block,procUnit1),
     processed(Block,procUnit3),
     finished(Block)]).

+!process(Block, ProcUnit) : over(Block, ProcUnit)
<- +processed(Block, ProcUnit).

+!consume(Block) : over(Block,depositBelt)
<- -over(Block, depositBelt); +empty(depositBelt);
     +finished(Block).

+!move(Block, Device1, Device2) :
     over(Block,Device1) & empty(Device2)
<- +over(Block, Device2); -over(Block, Device1);
   -empty(Device2); +empty(Device1).
```

```
+over(Block, feedBelt) : true
<-!finish(Block).

+!finish(Block) : Block = block1
<- !process(Block,procUnit1); !process(Block,procUnit2);
     !process(Block,procUnit3);!move(Block,procUnit3,dep
     ositBelt); !consume(Block).

+!finish(Block) : Block = block2
<- !process(Block,procUnit2); !process(Block,procUnit4);
     !move(Block,procUnit4,depositBelt);
     !consume(Block).

+!finish(Block) : Block = block3
<- !process(Block,procUnit1); !process(Block,procUnit3);
     !move(Block,procUnit3,depositBelt);
     !consume(Block).

+!process(Block, ProcUnit) : not over(Block,ProcUnit) &
     empty(ProcUnit) & over(Block,Device)
<- !move(Block,Device,ProcUnit);
     !process(Block,ProcUnit).

+!process(Block,ProcUnit) : over(Block,ProcUnit)
<-  +processed(Block,ProcUnit).

+!consume(Block) : over(Block,depositBelt)
<- -over(Block,depositBelt); +empty(depositBelt);
     +finished(Block).

+!move(Block, Device1, Device2) : over(Block,Device1) &
     empty(Device2)
<- +over(Block,Device2); -over(Block,Device1);
   -empty(Device2); +empty(Device1).
```

Related Work

- Motivations
  - Often used by biological systems
  - Provide a plan selection mechanism
- Declarative Goals
  - Decouple goal achievement from actions
  - Means-ends reasoning link current state to desired goal

Future Work

- Planning as an enabler for declarative goals
  - Balance dynamic and static behaviours
  - Multiagent planning
- Motivations as a control mechanism for
  - Planning effort
  - Agent interaction

Motivations and declarative goals as cornerstones of autonomy

Felipe Rech Meneguzzi

University of Southampton