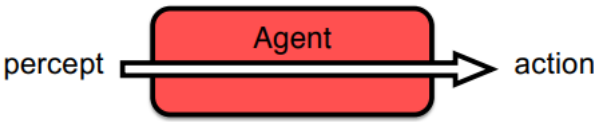
Contents

1 Limitations of simple reactive architecture

no representation of the environment means

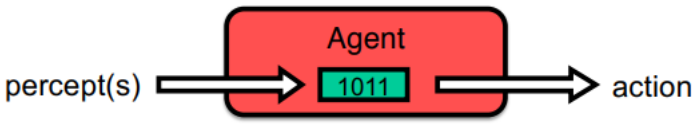
- its knowledge of the world is limited by the **range of its sensors**
- its unable to count (as opposed to recognise number)
- its unable to recover from actions which fail silently and many others

2 Modelling reactive behaviours



- we can model reactive behaviours as condition-action rules
- if the condition matches the agents precepts, it triggers an action `if percept then action`
- a simple reactive agent maintains no internal representation of the state of the world, whether the rule has been fired before etc.

3 Reactive architectures with state



- some rules match against an internal representation of aspects of the environment
- representations can be built using simple **percept-driven rules** (internal actions) which record simple beliefs about the state of the world

4 State

- we have simply taken a condition-action rule which matched against a percept and generated an action and split it in two, with a mediating internal representation
- needs extra machinery to store the state
- requires at least **two computation steps** to choose an action rather than one

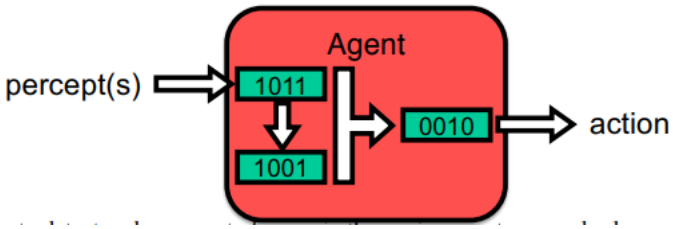
4.1 Actions which modify the internal state

- we can extend this to rules
  - whose conditions match against the agents internal state; and
  - whose actions modify the agents internal state
- again, this appears to make things worse: requires even **more space and more steps** to choose an action

4.2 Importance of representations

- notion of a rule which only responds to and generates internal changes in the agent is a key step
- forms the basis of all derived representations, and of representations which refer to other aspects of the agents internal state
- e.g., allows the agent to respond only to changes in the environment, ignoring features that are constant
- without some representation of the previous state, we cant say what is novel in the current state

5 Detecting change



to detect and represent changes in the environment we need rules

- whose conditions match against representations of the current and previous precepts
- whose action is to remember (copy) the state representing the current percept for use at the next cycle

5.1 Internal representations

- require more space and incur the cost of maintaining the representation
- allow the choice of actions based on sequences of states, e.g.: to react to change or **lack of it**
- given such internal behaviours, much more complex external behaviours are possible

6 Action selection function

- the action selection function for a reactive agent with state looks like
  - $selectAction : EventState \rightarrow ActionState$
- a reactive agent with state (finite-state machine) can respond to regular sequences of events
- we can add more complex state data structures to increase the capabilities of the agent, e.g.,
  - add a stack (pushdown automata) to respond to context-free sequences
  - add a random-access array to get a Turing machine, etc.

Reference section

placeholder